Genome Biology

METHOD Open Access

STRONG: metagenomics strain resolution on assembly graphs

Christopher Quince[1,2,3*], Sergey Nurk[4*], Sebastien Raguideau[1,3], Robert James[2], Orkun S. Soyer[5], J. Kimberly Summers[3], Antoine Limasset[6], A. Murat Eren[7,8], Rayan Chikhi[9] and Aaron E. Darling[10]

*Correspondence:
christopher.quince@earlham.ac.uk;
sergey.nurk@nih.gov
[1]Organisms and Ecosystems, Earlham Institute, Norwich NR4 7UZ, UK
[4]Genome Informatics Section, Computational and Statistical Genomics Branch, National Human Genome Research Institute, National Institutes of Health, Bethesda 20892, MD, USA
Full list of author information is available at the end of the article

Abstract

We introduce STrain Resolution ON assembly Graphs (STRONG), which identifies strains de novo, from multiple metagenome samples. STRONG performs coassembly, and binning into metagenome assembled genomes (MAGs), and stores the coassembly graph prior to variant simplification. This enables the subgraphs and their unitig per-sample coverages, for individual single-copy core genes (SCGs) in each MAG, to be extracted. A Bayesian algorithm, BayesPaths, determines the number of strains present, their haplotypes or sequences on the SCGs, and abundances. STRONG is validated using synthetic communities and for a real anaerobic digestor time series generates haplotypes that match those observed from long Nanopore reads.

Keywords: Microbiome, Metagenome, Strains, Bayesian, Microbial community, Assembly graph

Introduction

There is a growing realisation that to fully understand microbial communities it is necessary to resolve them to the level of individual strains [42]. The strain is for many species the fundamental unit of microbiological diversity. This is because two strains of the same species can have very different functional roles. The classic example is *E. coli*, where one strain can be a dangerous pathogen and another a harmless commensal [28]. The best definition of a strain, and the only one that avoids ambiguity, is a set of clonal descendants of a single cell [14, 46], but strain genomes by this definition can only reliably be determined by sequencing cultured isolates or single cells [37]. The former is not representative of the community and the latter is still too expensive and low-throughput for many applications as well as producing only fragmentary genomes. For these reasons, there is a practical need for efficient methods that can profile microbial communities at high genomic resolution.

In contrast to 16S rRNA gene sequencing, shotgun metagenomics has the potential to resolve microbial communities to the strain level. This is because it generates reads

© The Author(s). 2021 **Open Access** This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/. The Creative Commons Public Domain Dedication waiver (http://creativecommons.org/publicdomain/zero/1.0/) applies to the data made available in this article, unless otherwise stated in a credit line to the data.

from throughout the genomes of all the community members. It also has the additional advantages of reduced levels of bias and the capability to reconstruct genomes. There are many methods for reference-based strain resolution from metagenome data [1, 42, 49], but they are, and will continue to be, limited by the challenge of comprehensively isolating and sequencing the genomes of diverse microbial strains. Comprehensive reference genome databases may be possible for a few slowly evolving species or particularly well studied pathogens but for the entirety of a complex community it is unlikely to ever be tractable. For example, in a recent de novo large-scale binning study of the relatively well-studied human gut microbiome, it was found that 77% of the species recovered did not have a reference genome in public databases [38]. This suggests that even less of the strain-level diversity in those samples would be represented in a genome database. These observations motivate the need for de novo methods of metagenomic strain resolution.

In the metagenomics context, we adopt the definition of a 'metagenome strain' as a clonal subpopulation with sufficiently low levels of recombination with other strains, that it can be distinguished genetically from them. This does not require that recombination between strains does not occur, rather that either because of physical separation or selection, it has not been sufficiently strong relative to the rate of mutation [47], to generate a continuum of diversity throughout the genome. This means members of a 'metagenome strain' may differ substantially from each other particularly in rapidly evolving accessory regions and the subpopulation as a whole may descend from multiple cells but with a core genome that has descended from a single cell in the recent past. This is equivalent to the definition of 'lineage' in [35]. For ease, in the discussion below we will refer to strain in the metagenome context when properly we mean this looser definition of a strain as a genetically distinct subpopulation.

De novo assembly of genomes from short read metagenome sequences remains very challenging. Assemblies become fragmented for two reasons: firstly, low coverage genomes will fragment through chance occurrences where sequence coverage drops out, following Lander and Waterman statistics [18], secondly, if either intra or inter-genomic repeats are present then the assembly graphs used to represent possible sequence overlaps become very complex, and it is unclear which paths correspond to true genomes. Both of these issues are particularly problematic for metagenomes, where there can be a wide range of species abundances, and in a complex community a significant fraction of the species may be at low coverage. The first challenge can be addressed by sequencing more deeply. More difficult to address is the problem of repeats. Just as they do in isolate genome sequencing, intra-genomic repeats such as the 16S rRNA operon will lead to uncertainty in metagenomic assemblies, but if multiple closely related strains from the same species are present then they will possess potentially large regions of shared sequence. If the strain genomes are of comparable divergence to the reciprocal of the read length then very complex graphs will result, for typical short read sequencing (75-150bp) this would be strains at around 98–99.5% sequence identity. The result is that it is not possible to find long paths in the graph that can be unambiguously assembled into long contiguous sequence or contigs. For this reason metagenome assemblies for strain-diverse communities can comprise millions of contigs when made from short read data, with the added drawback that in the metagenomics context we do not even know which contig derives from which species. For species that contain multiple very similar strains ($>$ 99.9%), then we expect better assemblies but the variants are then too far apart to be linked or phased by Illumina reads. In that case we may resolve the large-scale genome structure but not the sequences of the individual strains, which we will refer to as their haplotypes.

Metagenomic contig binning methods attempt to mitigate the problem introduced by standard metagenome sample processing approaches, wherein the origin of each sequence read is unknown. Contig binning works because contigs deriving from the same or similar genomes will share features that can be learnt without prior knowledge. These features can be sequence composition, but it is also possible to use per-sample coverage depths of contigs as a more powerful feature, if multiple samples are available from the same (or very similar) communities [2]. There are now numerous algorithms capable of using both coverage across samples and composition to automatically cluster contigs and determine from single-copy core gene (SCG) frequencies where the resulting bins are good quality metagenome assembled genomes (MAGs) [3, 24]. These tools enable genome bins to be extracted de novo from metagenomes, and are becoming crucial for studying unculturable organisms, contributing to many exciting discoveries, such as the description of the Candidate Phyla Radiation [9] or an improved understanding of the diversity of nitrogen fixers in the open ocean [13].

The resolution of genome binning though, is limited by the resolution of the assembler, with a typical maximum kmer length of around 100, the best case is that we can resolve to about 1% sequence divergence, so that bins correspond to something between a species and a strain. In the presence of strain diversity, those contigs that are shared across strains will become a consensus of the strains present, in the ideal situation their sequence would be that of the most abundant strain, but even this is not guaranteed. Contigs that are part of the accessory genome and present in a subset of strains may be successfully binned with the core genome, but they may not if they are too short or divergent in coverage. Consequently, if multiple strains are present in the assembly the MAGs that result from binning will be an imperfect composite of multiple strains.

Strains in a metagenome can exhibit variation in shared genes, such as insertions/deletions and single-nucleotide variants or SNVs, as well as in their accessory gene complements. Methods for finding haplotypes on shared genes mostly fall into one of two types, either they operate on single samples independently [31], or they assume that only a single strain is present in each sample and then examine the sharing of those haplotypes across samples [36, 45]. The former will be limited by the challenge of resolving haplotypes purely from overlaps at least for short reads, the latter will break down if significant strain diversity exists within a sample. Recently, we introduced DESMAN [39] to resolve subpopulations in MAGs using variant frequencies on contigs when multiple samples from a community are available. This is similar to contig binning using coverage but it can be viewed as a relaxed form of clustering closer to non-negative matrix factorisation, because each variant can appear in more than one subpopulation haplotype. Similar strategies had been proposed prior to DESMAN but using variant frequencies on reference genomes e.g. Lineages [35] and Constrains [32]. DESMAN and other earlier methods are all ‘linear mapping-based methods’ where metagenomic reads are mapped onto a linear sequence, either a reference or consensus contig and the variant positions identified. This has multiple drawbacks: firstly, the type of variant that can be represented is limited to changes at a single base; secondly, mapping onto a linear sequence can be challenging when there is variation present yielding unreliable results [21]; thirdly, it treats every variant as independent ignoring the co-occurrence of variants in reads, which is a powerful extra source of information when strain divergence is greater than the inverse of read length, when we would expect most reads to contain more than one variant. The last issue can be addressed by keeping track of which variants appear in which reads but that requires extra bookkeeping [20].

To address these limitations, we introduce a new method, STRONG (Strain Resolution ON Graphs), for analysing metagenome series when multiple samples are available either from the same microbial community e.g. longitudinal time-series or cross-sectional studies where the communities are similar enough to share a significant fraction of strains. STRONG can determine the number of 'metagenome strains' in a MAG formed from binning of a coassembly of all the samples, together with their sequences across multiple single-copy core genes, which we refer to as *strain haplotypes*, and the coverages of each strain in each sample. STRONG avoids the limitations of the variant-based approaches by resolving haplotypes directly on assembly graphs using a novel variational Bayesian algorithm, BayesPaths.

This graph-based approach allows more complex variant structure and incorporates read information. The usefulness of graphs for understanding microbial strains has been noted before, and efficient algorithms developed for querying complex graphs and extracting more complete representatives of MAGs in the presence of strain diversity [10]. STRONG, however, is the first time that graphs have been used in an automated workflow to actually decompose that strain diversity into haplotypes across multiple genes using multiple samples. We compare STRONG to both the current state of the art, DESMAN, and a recent single-sample method mixtureS [31] on synthetic microbial communities and a real metagenome time series from an anaerobic digester. In the former case we validate using the known genome sequences, and for the latter we compare abundant MAGs with haplotypes derived independently from Oxford Nanopore MinION long reads.

Results

STRONG pipeline

The detailed pipeline is described in the Methods but the key steps are summarised in Fig. 1 and reiterated here. We start from multiple samples of the same community and jointly coassemble them with metaSPAdes, we save a high resolution graph (HRG) early in the assembly process that preserves all the variant information in the coassembly. The metaSPAdes assembly process then proceeds as normal and the resulting contigs are binned either with CONCOCT, used for all the results generated here, or alternatively Metabat2 [25]. We annotate the single-copy core genes in the contigs, allowing us to identify a subset of bins as MAGs. A novel algorithm was then developed to map these SCG ORFs onto the HRG and extract the complete assembly subgraphs corresponding to the genes of interest ("Methods - Relevant subgraph extraction" section). We obtained per sample unitig coverages on these subgraphs by threading reads directly onto them. These subgraphs were simplified with a noise filtering algorithm that used the MAG coverage depths, calculated as the length weighted average of the contigs assigned to that MAG. The simplified subgraphs contain all the information required for the BayesPaths algorithm ("Methods - BayesPaths" section), that simultaneously solves for the number of strains present, their coverage in each sample, and their sequences on the SCGs. SCGs from the same MAG are linked through the binning process and jointly solved in the

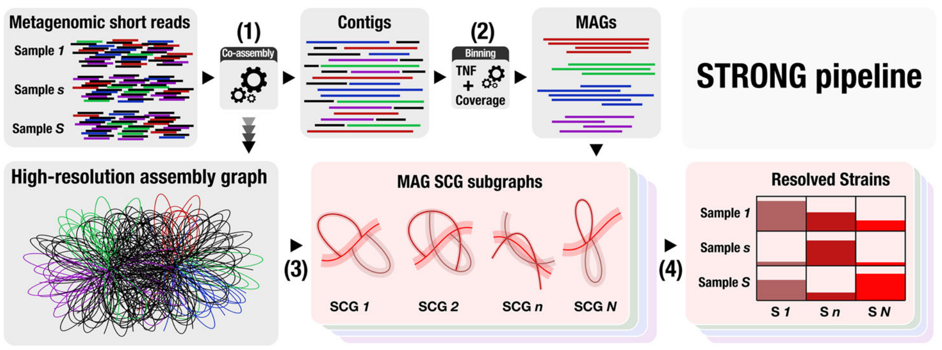

Fig. 1 STRONG pipeline. This figure illustrates the principal steps in the STRONG pipeline (see "Methods - STRONG pipeline" section). Step 1) Co-assembly with metaSPAdes and storage of a high-resolution graph (HRG). Step 2) Contig binning with CONCOCT or Metabat2 and annotation of single-copy core genes (SCGs). Step 3) Mapping of SCGs onto the HRG and extraction of individual SCG assembly graphs together with per-sample unitig coverages. Step 4) Joint solution of SCG assembly graphs from each MAG with BayesPaths to determine strain number, haplotypes and per-sample coverages

strain resolution procedure to generate linked strain resolved sequences for each SCG. We will refer below to the SCG sequences for a given strain as its haplotype. The pipeline also applies DESMAN [39], to the same MAGs for comparative purposes, and will perform benchmarking if known genomes are available. It is important to note that some SCGs will be filtered during the BayesPaths procedure, see "Methods" section, so that sequence inference is only performed on a subset in the final output.

The primary output of the STRONG pipeline comprises a collections of MAGs complete with strain number estimates, strain coverages across samples, and strain haplotype sequences on the filtered subset of SCGs. It does not perform assignment of accessory genes to strains. It will additionally perform taxonomic assignment of MAGs using GTDB [11] and generate phylogenetic trees for strains within MAGs and if evaluation genomes are available generate metrics of strain quality.

Synthetic data sets

In order to provide an example metagenome data set with a known strain configuration for each species, we created a synthetic community comprised of 100 strains, with known genomes deriving from 45 species, with 20 species represented by a single strain, 10 with two strains, 5 with three, 5 with four and 5 species with five strains. We then generated four data sets from this community with the same total number of reads (150 million 2X150 bp) but increasing sample numbers (3, 5, 10 and 15 samples). This configuration, where most species have a single strain, might be an appropriate approximation to the human gut microbiome [45]. We denote these data sets Synth_S03, Synth_S05, Synth_S10 and Synth_S15. For each sample number, random species abundances were generated from a log-normal distribution, with strain proportions from a Dirichlet with unit variance. Full details of the synthetic sequence generation are given in the Methods.

The STRONG pipeline was applied to each of these data sets in turn generating MAGs for which strains were then resolved. In Fig. 2 we illustrate the STRONG output for a single gene, COG0532 'Translation initiation factor IF-2' [44], from one MAG, Bin_55 of the ten sample synthetic data set, giving the resulting decomposition of the assembly subgraph into three strains. Noting that the strains were resolved in this MAG over 22

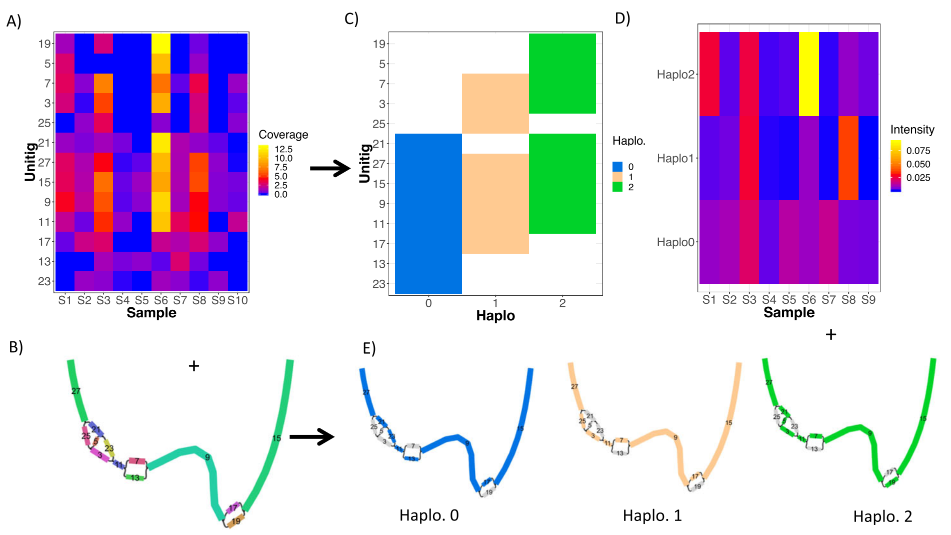

Fig. 2 BayesPaths algorithm. This illustrates the BayesPaths algorithm for a single COG0532 from one MAG, Bin_55 of the ten sample synthetic data set. The algorithm predicted 3 strains. We show the input to the algorithm: **A** the unitig coverages across samples plus **B** the unitig graph without strain assignments. The outputs of the algorithm are shown in **C** the assignments of haplotypes to each unitig, **D** the strain intensities across samples, effectively coverage divided by read length (see "Methods - BayesPaths" section), and **E** unitig graphs for each haplotype with their most likely paths. This algorithm is explained in detail in the "Methods - BayesPaths" section

single-copy core genes simultaneously, and that for this 3.4 kbp gene the haplotypes were found without errors.

To provide a comparison to STRONG we ran two other algorithms, the single sample algorithm mixtureS [31], and our existing SNV based multi-sample algorithm DESMAN [39]. STRONG and DESMAN utilising variant graphs and read-mapping respectively are completely independent algorithms. We ran mixtureS with default parameters combining the individual sample data sets so that the overall strain coverages were the same across algorithms. In Fig. 3 we compare the algorithms in terms of their ability to correctly reconstruct the number of strains in a MAG. By far the most reliable predictions were generated by STRONG, although there was a tendency to be conservative and slightly underestimate strain number. DESMAN did less well although it still outperformed mixtureS, which exhibited no significant association between true and predicted strain number.

For each of the four synthetic data sets we considered only MAGs which were assigned to species (see "Methods" section) with at least two strains - 20, 21, 25 and 23 MAGs, from the Synth_S03, Synth_S05, Synth_S10 and Synth_S15 data sets respectively. For each MAG we mapped the predicted haplotypes for the optimal strain decomposition for the STRONG pipeline, DESMAN algorithm and mixtureS onto the known reference strains. We then assigned each haplotype prediction to its best matching reference. The best such match was denoted 'Found'. If more than one predicted haplotype was matched to the same reference, all but the best matching 'Found' strain were denoted as 'Repeated'. If a reference had no haplotype prediction that matched to it better than the other references, it was denoted as 'Not found'. For the aggregate across these MAGs we show the total number of such strains for each of the four data sets in Fig. 4.

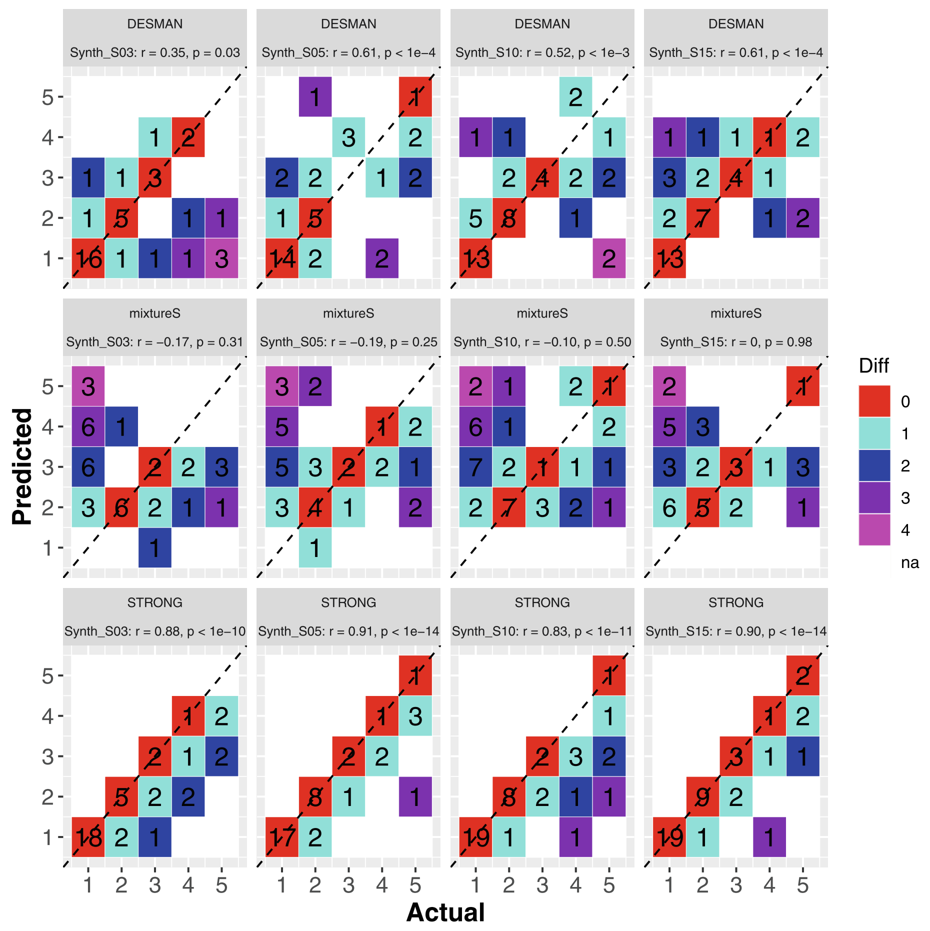

Fig. 3 Actual versus predicted strain number for the synthetic community data sets. For each MAG we compare the actual number of strains against the predicted number. The number in each tile gives the total no. of MAGs observed with those values. The colour of a tile the divergence between true and predicted strain numbers. Results are shown for all four data sets Synth_S03, Synth_S05, Synth_S10 and Synth_S15 with increasing sample number and three algorithms, DESMAN, STRONG, and mixtureS. The results of Pearson's correlations are given in the title texts

STRONG consistently performs better than DESMAN and mixtureS in terms of number of strains in the 'Found' category, in total across all four samples it resolved 216 strains vs. 208 for DESMAN i.e. a 3.8% increase and 40% more than mixtureS which resolved just 154. It also had fewer 'Repeated' strains, 6 vs. 30 for DESMAN: a reduction of 80% and as compared to 64 for mixtureS. The strains 'Found' were also reconstructed more accurately, the per base error rate for the BayesPaths reconstructions averaged across all MAGs and all data sets was just 0.052%, three times lower than that for DESMAN, 0.176%, and nearly ten times more accurate than mixtureS at 48.7%. This improvement was observed for all four data sets (see Table 1 and Fig. 5).

STRONG was also better at predicting the strain relative abundances. Regressing true abundance against predicted abundance gave an adjusted R^2 of 0.85 averaged across sample numbers for STRONG vs. 0.81 for DESMAN. When this was restricted to MAGs where the number of strains was correctly predicted, then both algorithms did better

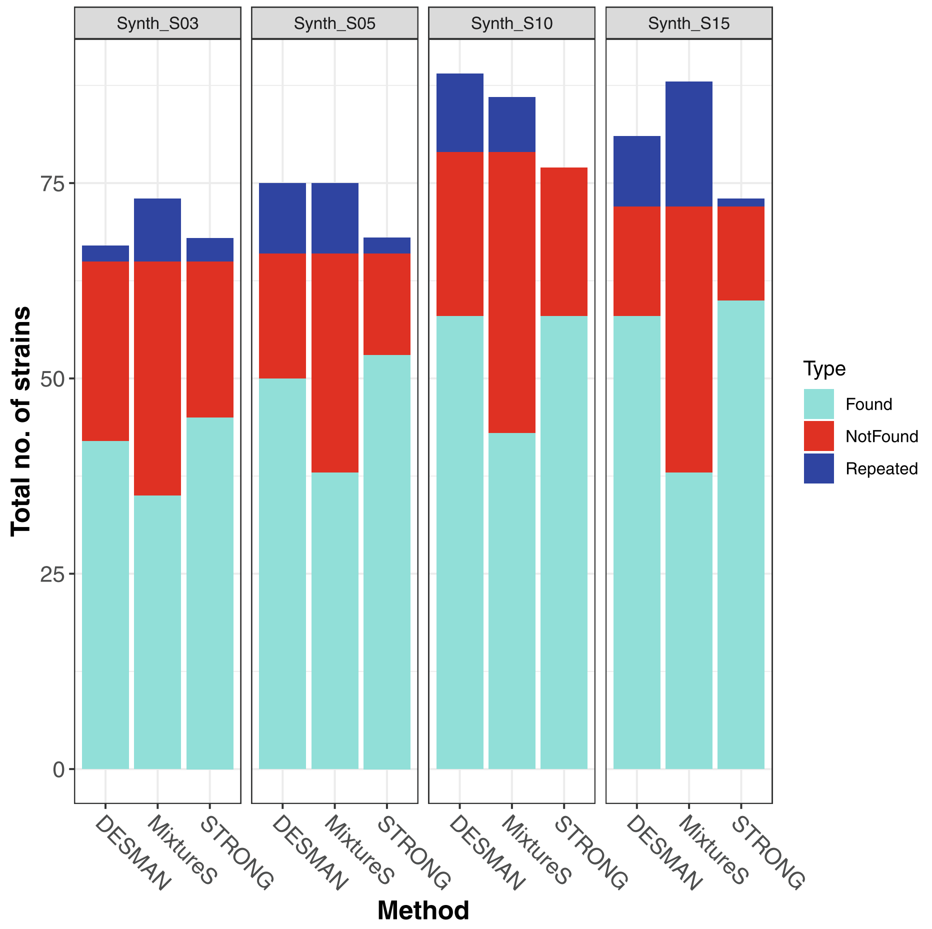

Fig. 4 No. of strains resolved by STRONG, DESMAN and mixtureS algorithms in the synthetic community data sets. For MAGs with two or more strains we mapped haplotypes to the references and assigned each predicted haplotype to its best matching reference. The best such match was denoted 'Found'. If multiple haplotypes matched to the same reference all but the best matching were denoted as 'Repeated'. If a reference had no predicted haplotypes matched to it, it was denoted as 'Not found'. The bars give the total numbers in each category summed over MAGs for the three methods (DESMAN, STRONG and mixtureS) and the panels results for the four different data sets with increasing number of samples (Synth_S03, Synth_S05, Synth_S10 and Synth_S15)

but STRONG still out performed DESMAN, with a mean R^2 of 0.98 compared to 0.95. STRONG filtered roughly 1/3 of the SCGs as outliers although the exact number varied across the four data sets (see Table 1). Outlier SCGs were identified by coverage error rates that exceeded those expected based on analysis of the median absolute deviations, a procedure designed to identify contaminant SCGs introduced by incorrect contig binning (see "Methods" section). The method mixtureS does not predict abundances across multiple samples so was not compared here.

The STRONG pipeline outperforms DESMAN, but it still misses strains that are present. In total across all MAGs and data sets, 64/280 i.e. 22.8%, of strains were missed by STRONG. Some of these, 7 out of 64, were below the minimum coverage of detected strains (5.68), but most were not, suggesting that either they were not sufficiently divergent in terms of nucleotides or coverage profiles to be detected. Examination of

Table 1 Comparison of STRONG, DESMAN and mixtureS for strain reconstruction in the synthetic community data sets

Method	Data set	MAGs	#SCGs	#fSCGs	Found	Not F.	Rep.	Err	R^2	f^G
STRONG	Synth_S03	20	33.18	19.71	45	20	3	0.069	0.81(0.99)	26/38 = 0.68
DESMAN					42	23	2	0.125	0.81(1.00)	26/38 = 0.68
mixtureS					35	30	8	0.706	—	8/38 = 0.21
STRONG	Synth_S05	21	32.13	22.71	53	13	2	0.062	0.87(0.99)	29/38 = 0.76
DESMAN					50	16	9	0.222	0.83(0.96)	20/38 = 0.53
mixtureS					38	28	9	0.590	—	7/38 = 0.18
STRONG	Synth_S10	25	32.05	22.72	58	19	0	0.036	0.84(0.99)	30/43 = 0.70
DESMAN					58	21	10	0.206	0.79(0.92)	25/44 = 0.57
mixtureS					43	36	7	0.279	—	9/44 = 0.20
STRONG	Synth_S15	23	32.17	24.19	60	12	1	0.046	0.87(0.97)	34/42 = 0.81
DESMAN					58	14	9	0.144	0.81(0.92)	25/42 = 0.60
mixtureS					38	34	16	0.321	—	9/33 = 0.21

Data set: Results are shown for the four different sample numbers. MAGs: The number of MAGs reconstructed with more than two reference strains. #SCGs: The average number of SCGs found in each MAG. #fSCGs The average number of SCGs after filtering in STRONG. Found: Number of strains matched to a reference strain. Not F.: Number of reference strains that had no predicted strain with a closest match to it. Rep.: Number of strains matching to a reference that already has a better match. Err: The average error rate of the 'Found' strains in percentage base pairs. R^2: Correlation between predicted and actual strain relative proportions given as adjusted R^2, the figure in parentheses is when restricted to MAGs where the number of strains was correctly predicted.f^G: the fraction of MAGs where the number of strains was correctly inferred

phylogenetic trees for the haplotypes and reference genomes constructed using the SCGs revealed that in many cases 'Not found' strains had identical SCG haplotypes to those that were resolved.

The BayesPaths algorithm used to resolve strains in STRONG uses variational inference (see "Methods - BayesPaths" section), an approximate Bayesian strategy [7]. This has the advantage of providing estimates of uncertainty in the inference of both the strain haplotypes and their abundances. The algorithm predicts the marginal probabilities that a given strain passes through a particular unitig. To provide a single sequence for the evaluation above and applications below we output the most likely path and hence sequence for each strain. However, we also calculate an estimate of path uncertainty by sampling many possible paths (default 100) consistent with the marginal distributions and calculate the average number of nodes that deviate from the most likely path, we refer to this as the divergence. For the 'Found' strains this correlates strongly with actual error rate to the reference strain (Pearson's correlation $r = 0.54$, $p < 2.2e-16$ - see Additional file 1: Figure S1). Thus the divergence is a useful prediction of uncertainty in the haplotype sequence inference, enabling us to estimate error rates in real data sets in the absence of known reference sequences. Roughly speaking, the expected per base error rate is 0.01 times the divergence, so that a strain divergence of 0.1 predicts a 0.1% error rate. In real data sets, the uncertainty estimates in the abundances are also useful, placing bounds on the abundance of individual strains in each sample.

In Additional file 1: Table S3, we give approximate run times for each component of the STRONG pipeline on the synthetic community data sets, using 64 threads on a standard bioinformatics server. The BayesPaths step is the most time consuming part of the analysis (up to 36 hours), but it is still comparable to the initial coassembly. The only part of the pipeline with substantial memory requirements is the initial coassembly with metaSPAdes, the other steps are CPU limited.

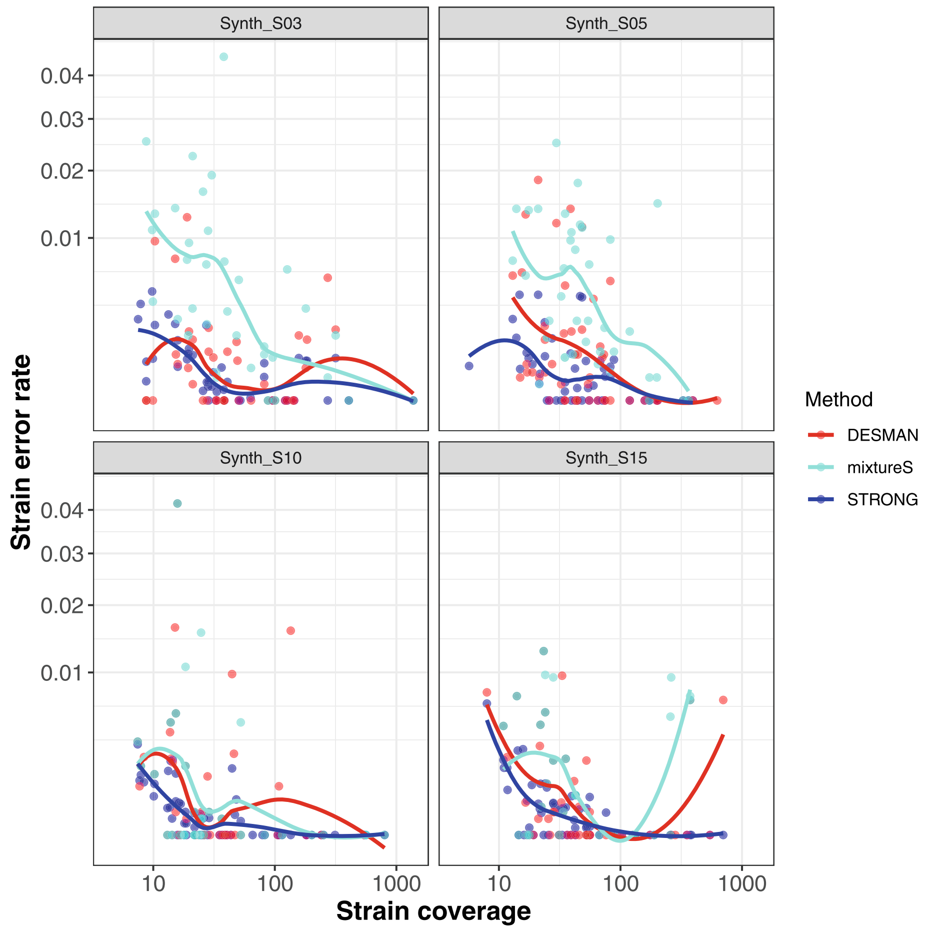

Fig. 5 Error rates for 'Found' strains against coverage depth for STRONG, DESMAN and mixtureS algorithms in the synthetic community data sets. For the 'Found' strains we computed per base error rate to the matched reference, this is shown on the y-axis, against strain total coverage depth summed across samples on the x-axis, both axes are log transformed. The results are separated across methods (DESMAN, STRONG and mixtureS) and sample number in the synthetic community

Limits of strain diversity resolvable by STRONG

The synthetic data sets considered above had at most five strains present for each species. To determine the upper limit on the number of strains resolvable by STRONG we also simulated a community with higher strain numbers. This comprised just twelve species but with strain numbers that increased from five to a maximum of ten, with two species with each strain number, i.e. 90 genomes in total. We generated four data sets using this strain configuration, using the same strategy of increasing sample numbers (3, 5, 10 and 15 samples) as outlined above, but with a higher total read number (250 million 2X150 bp reads) to ensure that most strains had a coverage above the detection limit. We denote these data sets Synth_M10_S03, Synth_M10_S05, Synth_M10_S10 and Synth_M10_S15.

We ran the complete STRONG pipeline on these four high strain diversity sets. We do not present results from running the other two algorithms since they were significantly less effective than STRONG on the lower diversity data sets, and we would not expect that to change at high strain number. The pipeline generated 11 MAGs out of the 12 species possible, for each of the data sets except for when five samples were

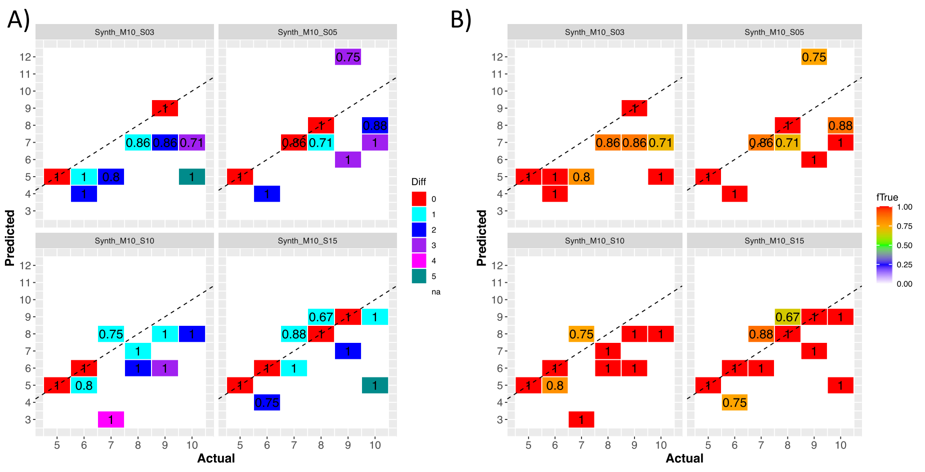

Fig. 6 Strain numbers resolved by STRONG in the high strain diversity synthetic community data sets. **A** For each MAG we compare the actual number of strains against the predicted number. The number in each tile gives the total no. of MAGs observed with those values. The colour of a tile the divergence between true and predicted strain numbers. Results are shown for all four high strain diversity data sets: Synth_M10_S03, Synth_M10_S05, Synth_M10_S10 and Synth_M10_S15 for the STRONG algorithm only. The results of Pearson's correlations between actual and predicted strain number were for Synth_M10_S03 (r = 0.62, p = 0.04), Synth_M10_S05 (r = 0.57, p = 0.11), Synth_M10_S10 (r = 0.59, p = 0.05), and Synth_M10_S15 (r = 0.52, p = 0.10). **B** The same data are shown but now for each tile we give the mean fraction of predicted strains that were 'Found' i.e. mapped uniquely onto a reference strain

used (Synth_M10_S05) when 9 MAGs were obtained. In Fig. 6A, we compare actual vs. predicted strain number for these MAGs across data sets. Overall, we did observe correlations between the two, so that MAGs with more strains generally had more strains predicted, but not the highly significant relationships we found in the less diverse synthetic communities. The correlations also did not noticeably improve with increased sample number. In contrast, the accuracy of the resolved strains did. From Fig. 6B we see that the proportion of predicted strains that match uniquely to reference strains increased for higher sample numbers and for one MAG in the 15 sample data set we predicted nine strains with each uniquely mapping to a true reference. The mean error rate of the 'Found' strains decreased with sample number too, from a mean error rate of 0.31% for the three sample data set Synth_M10_S03, to 0.27%, 0.18% and 0.12% for five, ten and fifteen samples respectively. As for the simpler synthetic communities, error rates decreased with increasing strain coverage for all sample numbers (see Additional file 1: Figure S2).

Impact of variability in strain proportions across samples on reconstruction accuracy

The BayesPaths algorithm depends on differences in strain profiles across samples in order to link unitigs into haplotypes. In some real communities, strain proportions may actually be surprisingly stable, as discussed below. The synthetic communities analysed above generated strain relative abundances in each sample from a Dirichlet distribution with unit variance which is equivalent to a uniform distribution, this implies a high degree of variability between samples. To relax this assumption and provide benchmarks in the case where strains may have more stable proportions we generated an additional set of benchmark examples. We used a Dirichlet distribution to first generate a mean strain profile for each MAG and then generated the individual sample profiles from this, again using a Dirichlet but with a precision, i.e. inverse variance parameter θ, that can be varied across data sets. The result is a hierarchical Dirichlet distribution for strain proportions, that enables us to control the expected variability and model potentially more stable strain proportions (see "Methods" section).

We generated three ten sample data sets all comprising the same ten species, each with five strains, but with increasing precisions for the sample level Dirichlet corresponding to reduced difference in profiles between samples. The first had $\theta = 1$, which corresponds to strain proportions that differ greatly from one sample to another, the second $\theta = 10$, and the third $\theta = 100$, we denoted these Synth_T1_S10, Synth_T10_S10, and Synth_T100_S10 respectively. The results of running STRONG on these three data sets are summarised in Additional file 1: Table S4. We confirmed the expected reduction in strain variability with increasing θ, by computing the average coefficient of variation in strain proportions in each data set, which is simply the standard deviation in strain relative abundance divided by the mean. This decreased from 1.45 in Synth_T1_S10, to 0.65 in Synth_T10_S10, and 0.23 in Synth_T100_S10. This reduction in variability was not accompanied by a clear reduction in the number of strains found, or number of MAGs for which the correct number of strains was predicted (that is five), but there was a decrease in the accuracy with which the strain haplotypes were reconstructed. The error rate increased from pratically zero in Synth_T1_S10, to 0.086% at the intermediate θ value Synth_T10_S10, to 0.38% for the most stable strain configuration, Synth_T100_S10.

Anaerobic digester time series

We next applied the STRONG pipeline to a real metagenomics time series, comprising ten samples taken at approximately 5 weekly intervals, from an industrial anaerobic digestion reactor (see Additional file 1: Table S5 and Methods for details). This provides an evaluation community of intermediate complexity to test the pipeline's capability to resolve strains and reconstruct intraspecies dynamics. Each sample was sequenced on the NovaSeq platform with 2x150 bp reads at a mean depth of 11.63 Gbp. One sample was also run on a Nanopore MinION flow cell producing 43.78 Gbp of reads with a read N50 of 6,727 bp and a maximum length of 108 kbp.

CONCOCT binning produced 905 bins, of which 309 had 75% of SCGs present in single-copy, which we designate MAGs. In total 11 of these MAGs exhibited overlapping SCG graphs and were merged into 6 composite MAGs (see "Methods - STRONG pipeline" section), so that 304 MAGs were actually used in the strain decomposition. We calculated coverage depth per sample for each bin and then normalised by sample size to obtain a community profile at each time point. Overall the reactor exhibited a clear shift in community structure over time, despite consistent operating conditions, with sample time explaining 48% of the variation in community structure ($p = 0.001$ - Additional file 1: Figure S3). Of the MAGs, 110 had an abundance that changed significantly over time (Bonferonni adjusted p-value <0.05 from Pearson's correlation of log transformed normalised abundance) and these were evenly split between those that increased (55) or decreased in abundance (55).

We used the STRONG algorithm to resolve strains in the 304 MAGs. This is a complex data set and running the complete pipeline took over 16 days, of which roughly 60% of the time was spent on the BayesPaths strain resolution (see Additional file 1: Table S3). The number of strains found varied between 1 and 7, with a mean of 1.7, shown as a function of coverage depth in Fig. 7. In total 121 (39.8%) of these MAGs had more than

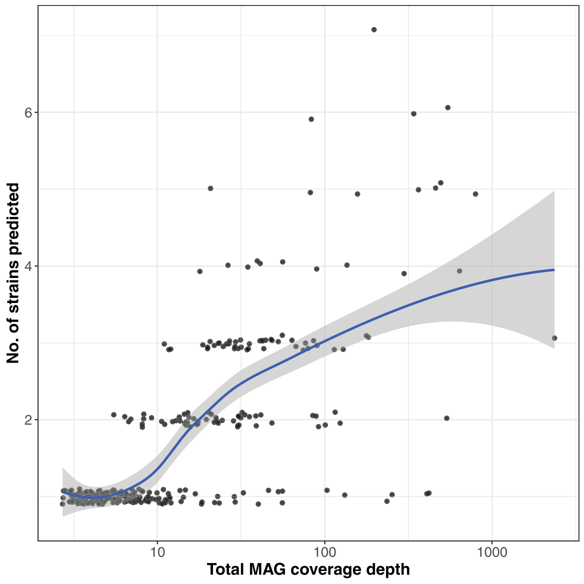

Fig. 7 Number of strains resolved by STRONG against MAG coverage depth for the AD time series. Pearson's correlation between coverage depth and number of strains ($r = 0.36$, $p = 1.004e - 10$). The curve indicates a LOESS smoothing

one strain, and there was a significant positive association between MAG coverage depth and number of strains ($r = 0.36$, $p = 1.004e - 10$), which is expected, as low coverage MAGs will be under-sampled. This correlation disappears though when we restrict to all MAGs with a coverage greater than thirty ($r = 0.19$, $p = 0.1023$). On average 20.9 SCGs were used after filtering for strain haplotype predictions.

For the 108 MAGs that had at least two strains with relative frequencies determined in five or more samples we used permutation ANOVA to determine whether strain proportions depended on sampling time. In total 13 of the MAGs had an adjusted p-value <0.05 i.e. 12.0%. For these same MAGs 37 had a total coverage that changed significantly over time with an adjusted p-value <0.05 i.e. 34.2%. Therefore the intra-species dynamics are more stable than inter-species, with strain proportions remaining fixed as the MAG coverages vary, this was true for 33 of the 37 MAGs that changed significantly in coverage.

In Fig. 8, we use the Anvi'o program [16, 17] to summarise information on phylogeny, taxonomy, normalised coverages in the ten samples, and whether the MAGs changed significantly in total abundance, together with the number of strains resolved by STRONG and if those strain relative proportions changed significantly with time. This was restricted to just those 114 MAGs with an aggregate coverage greater than twenty to simplify the diagram.

The Nanopore sequencing provides us with a means to directly test the validity of the STRONG haplotype reconstructions, at least for the most abundant MAGs. The most abundant MAG, Bin_72, had an aggregate short read coverage depth of 2364.25, across all the samples. This MAG was assigned to the phylum Cloacimonadota using the GTDB taxonomy [11]. Interestingly, this is an example of a MAG which changes significantly in abundance, decreasing over time, (adjusted $p = 4.9e - 05$) but where the proportions of the three strains predicted varied less dramatically ($R^2 = 0.35$ adjusted $p = 0.089$ - Additional file 1: Figure S4). We will focus on the longest SCG for which strains were resolved,

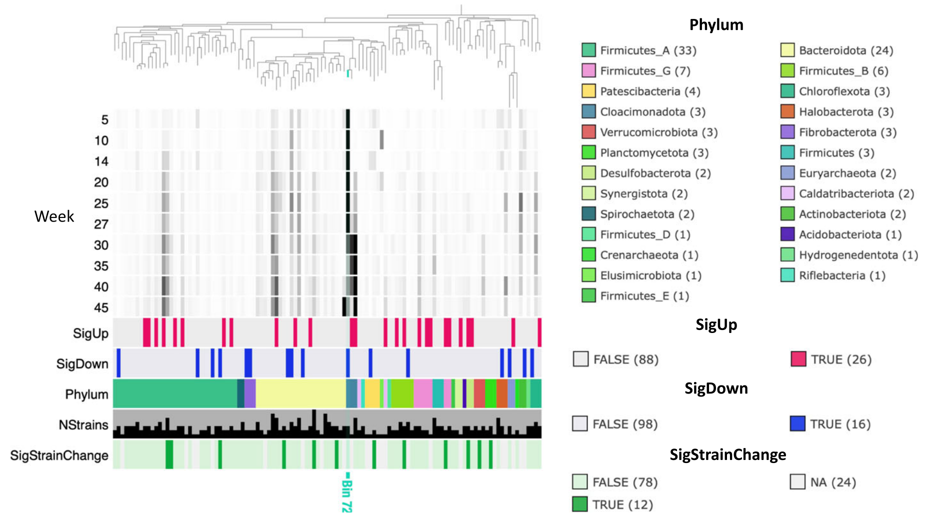

Fig. 8 MAG summary for anaerobic digester time series. For the 114 MAGs with aggregate coverage > 20 we give their phylogeny constructed using concatenated marker genes together with their normalised coverages in the ten samples. We also indicate which MAGs significantly increased (SigUp) or decreased (SigDown) in total abundance (adjusted $p < 0.05$), their GTDB phylum assignment, no. of strains resolved by STRONG and whether the strain abundances changed significantly over time (adjusted $p < 0.05$) using permutation ANOVA (SigStrainChange)

COG0532, where the three strains are present in only two variants, haplotypes 0 and 2 being identical on this core gene. In Additional file 1: Figure S5 we give the short read variant graph for this gene, which in this case is mostly simple bubbles, together with the assigned haplotypes. In fact, across the 18 SCGs used to decompose strains, haplotypes 0 and 1 were most similar with 99.7% nucleotide identity. These two strains had 99.4% and 99.1% identity with haplotype 2 respectively. That this pattern was not observed on COG0532 may suggest some recombination in the evolution of these organisms.

In Fig. 9 we show for COG0532 both the Nanopore reads that map to this gene and the three haplotypes inferred by BayesPaths, as an Non-metric Multi-dimensional Scaling (NMDS) plot using fractional Hamming distances on the short read variant positions. These are defined as the Hamming distance between two reads but only on the intersecting variant positions and ignoring gaps. We then normalise by the number of such non-gap intersecting positions to give a distance between 0 and 1. The Nanopore reads are consistent with the inference of two variants on this gene, as there are two clear clusters observed, and the two modes of those clusters are close to those haplotypes. The variation around the modes is caused by the high error rate of the Nanopore reads.

In order to provide a quantitative comparison of the Nanopore reads and the STRONG predictions, we applied the EM algorithm defined in the Methods (Nanopore Sequence Analysis) on the 1,603 Nanopore reads mapping to this COG (cluster of orthologous groups). Examining the negative log-likelihood as a function of number of strains, it flattens at two strains (see Additional file 1: Figure S6) and the two strains inferred exactly match (100% identity over 2,313 bps) haplotypes 0/2 and 1 respectively. Furthermore, STRONG in this sample predicted frequencies of 28.0% for haplotype 1. This closely matched the Nanopore haplotype frequencies for this strain of 27.6%. We also ran the Nanopore EM algorithm for all 18 filtered COGs in this bin separately. For the 11 COGs

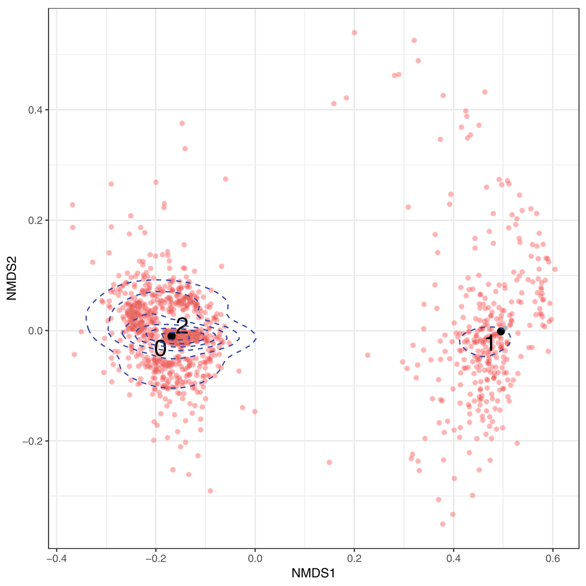

Fig. 9 Comparison of Nanopore reads to STRONG prediction for COG0532 from Bin_72. Non-metric multidimensional scaling of Nanopore reads that mapped to COG0532 from Bin_72 of the anaerobic digester time series (red) together with the three haplotypes reconstructed from short reads by STRONG (black 0, 1 and 2). Haplotypes 0 and 2 were identical for COG0532. Distances were calculated as fractional Hamming distances (see text) on short read variant positions (see "Methods - Nanopore sequence analysis" section). Blue dashed lines indicate read density contours

where more than one strain was predicted from the Nanopore reads, we compared the STRONG and Nanopore predictions. For haplotypes 0, 1 and 2 exact matches were found for 6, 7 and 4 SCGs respectively with average nucleotide identities across all genes of 99.89%, 99.89% and 99.82%.

For lower coverage MAGs we generally obtain a reasonable correspondence between the STRONG haplotypes and Nanopore predictions. In most cases the number of strains is comparable between the two, although the accuracy of matches reduces with decreased Nanopore read counts, as we might expect. As an example, in Additional file 1: Figure S7 we compare Nanopore reads with the five STRONG haplotypes from COG0072 of Bin_846, a Firmicutes MAG in the AD time series. The short read variant graph for this gene is shown in Additional file 1: Figure S8. The most abundant Nanopore mode clearly matches STRONG haplotype 4, the most abundant strain in this sample, and there is also some support for haplotypes 0 and 2. There is less evidence for strains 1 and 3, but these are low abundance in this sample (see Additional file 1: Figure S9). This is confirmed from the EM algorithm applied to the Nanopore reads matching this gene, where we would predict four Nanopore haplotypes (Additional file 1: Figure S10). Comparing these 4 Nanopore strains to the STRONG predictions (see Additional file 1: Figure S11) we find that three closely match: Nanopore haplotype 0 matched best to STRONG strain 4 with 98.8% nucleotide identity, Nanopore haplotype 1 to STRONG 4 with 99.9% identity and Nanopore haplotype 2 to STRONG strain 0 with 99.7% identity. There is also a correspondence in relative abundance, with the most abundant Nanopore haplotype 1 recruiting 82% of the reads vs 74% relative frequency for the corresponding strain haplotype from STRONG.

Discussion

We have demonstrated that on the synthetic data sets with relatively low strain number, less than or equal to five strains per species, the STRONG pipeline and the BayesPaths algorithm are able to accurately infer strain sequences on the SCGs and abundances across samples. Performance does improve with increasing sample number in terms of the number of strains resolved, but reassuringly even when only a small number of samples are available we are still able to accurately predict (with 0.068% per base error rate) strains, and when ten or more samples are available we obtain error rates below 0.05% i.e. 1 error in every 2000 bps from short read data. This is better performance than the read-based DESMAN algorithm illustrating the advantage of a graph-based approach. DESMAN in turn outperformed mixtureS [31], which is also read-based but uses only a single sample, confirming the advantage of deconvolving across multiple data sets when these are available. Strains are resolved more accurately as they increase in coverage (see Fig. 5), and in fact, when coverages exceed twenty fold we can resolve strains very reliably, with just 0.011% error rate averaged across strains in the ten sample low diversity synthetic data set. This is sufficiently accurate for high-resolution phylogenetics.

There is a decrease in strain reconstruction accuracy when high levels of strain diversity are present, in the data sets where every MAG had at least five strains, then the per base error rate increased to 0.18% for ten samples. However, even in this situation correct strain numbers were inferred in some instances, and at the higher sample number we did on a number of occasions correctly determine up to eight or nine strains which probably represents the upper limit for our approach. There was also a substantial decrease in haplotype reconstruction accuracy when strain proportions became more stable across samples, although those proportions were themselves accurately predicted. The latter is important in the context of our observation of stable configurations such as these in the AD time series. However, given these caveats regarding highly diverse or stable MAGs, we believe that this pipeline will be useful whenever high quality de novo strains are required from metagenome short read time series.

We do not present results comparing our approach to the strategy of simply reconstructing strains as the most abundant haplotypes in individual samples, as used in StrainPhlAn [45] or inStrain [36]. The implicit assumption in those methods is that one strain dominates each sample, therefore the strain mixtures that we are testing are outside their scope. Such a strategy will also inevitably fail to resolve any strains that are not the most abundant at least somewhere, and furthermore will not estimate strain proportions. They may be appropriate for some microbes in low strain diversity communities such as the human gut but will not have the range of applicability of our approach.

This is to our knowledge the first algorithm capable of constructing strains from metagenomes using assembly graphs from multi-sample coassemblies. Graph-based haplotype resolution has been applied to viruses [4] and for eukaryotic transcripts [5, 6], but ours is the first algorithm to resolve strains across multiple gene subgraphs connected through a contig binning procedure. The BayesPaths algorithm is also a substantial algorithmic advance enabling coverage across multiple samples to be incorporated into a rigorous Bayesian procedure that gives uncertainties in both the paths (i.e. the sequences) and the strain abundances.

In addition, to the new strain resolution algorithm, BayesPaths, STRONG incorporates a number of useful tools for large-scale variant graph processing, in particular, the tools for extraction of subgraphs that correspond to individual coding genes and the spades-gsimplifier tool for error correction on those graphs. These can be applied to any graph in the GFA format, and could therefore find applicability outside of the context of our pipeline. This also means that in the future we could add alternative choices for the coaseembly step, for instance replacing metaSPAdes with MEGAHIT [29].

The STRONG pipeline resolves strains in a MAG collection generated either by CONCOCT or Metabat2. It will therefore be most effective if a comprehensive set of MAGs can be generated from the coassembly for a particular community. The strain resolution by BayesPaths does, however, include an additional SCG filtering step so even the low quality MAGs should be usable. Currently, we are restricted to core genes that are single-copy and shared across all strains in a MAG. We can in theory use any such genes, specified by a pipeline configuration file, so if a particular MAG is of interest the pipeline could be run with a larger set of COGs that are SCGs for that MAG. There would be a cost in terms of increased running time, which will increase with more genes and unitigs in a roughly linear fashion.

The limitation to prokaryotes is perhaps more fundamental, the default choice of 36 COGs in STRONG is designed for bacteria and archaea. The pipeline cannot be simply extended to eukaryotes by specifying a set of COGs that are appropriate for eukaryotes, because even if a set of core COGs for eukaryotes could be specified, the ORF calling via prodigal currently implemented, and the SCG subgraph extraction assumes prokaryotic genes. There is no technical reason preventing an adaptation of the pipeline to eukaryotes but it would require substantial further work and validation.

STRONG itself operates on prokaryotic MAG collections, but the strain resolution, BayesPaths, could be applied to any set of SCGs from the same species, these could, if MAG generation fails for a species of interest, be obtained by mapping SCG containing contigs to a reference genome, and subgraphs extracted and processed as in the default STRONG pipeline. This would require some reengineering but it is quite feasible. In fact, even individual genes could be used for strain resolution without any binning at all, to provide a lower resolution but more comprehensive picture of strain diversity in a community. These are both functionalities we plan to add to STRONG in the near future. BayesPaths could also be used in other application areas, for example for finding viral haplotypes.

The analysis of a time series from an anaerobic digestor illustrates the practicality of our pipeline on a realistically sized data set. We should note though that to resolve strains on these 304 MAGs took nearly 10 days using 64 threads on a standard bioinformatics server (see Additional file 1: Table S3). The AD analysis also demonstrates the importance of strain dynamics in a real microbial community with nearly 40% of MAGs exhibiting strain variation, but this variation was relatively stable compared to the MAG dynamics themselves. If strains are functionally redundant to one another we would expect significant neutral fluctuations over time. Therefore this could be evidence for intra-species niche partitioning.

In general, we found a good correspondence between haplotypes inferred from Nanopore reads and the STRONG predictions in the AD data set. For the most abundant MAG, Bin_72, they matched very closely. In addition, the relative abundances of strains were consistent across the two sequencing technologies, despite the use of different DNA extraction protocols, and the different biases inherent in library preparation and sequencing platforms. These technical elements in the data generation process are known to introduce bias at the species level [12], but our findings suggest that intraspecies abundance may generally be robust against such biases, which makes sense in that all the strains of a species will have similar physical properties and genomic traits.

STRONG is an effective strategy to de novo resolve subpopulations at high phylogenetic resolution within MAGs, but as discussed in the Introduction, it is important to add the caveat that the haplotype sequences obtained are not equivalent to those from sequencing cultured isolates, where we can identify the resulting genome with a single organism present in the original community. The metagenome strains, in the best case, will correspond to different modal sequences of the target species, about which substantial unresolved variation may exist. They will correspond to peaks in the probability distribution of all possible sequence configurations, and as such will provide important insights into the naturally occurring variation, but there remains the question of how to identify and quantify the unresolved variation surrounding those peaks. In the worst case, when STRONG is applied to rapidly recombining microbes, such as those found in the oceans [37], the resulting sequences may not even be real in the sense of characterising any true individual.

This is likely to be more of an issue when STRONG is applied to spatial or cross-sectional data sets, rather than time series, for the latter we expect to sample the same or very similar clonal populations multiple times, but for the former local adaptation may lead to variation in accessory genes even between populations that share a recent common ancestor. An additional unaddressed question is how to determine when this has occurred, for now we would simply urge caution when using STRONG in cross-sectional studies of rapidly evolving microbes, and suggest that the term 'metagenome strain' or 'metagenome haplotype' be used when referring to the output sequences. The same caveat does of course apply to any current purely bioinformatics strategy for de novo resolution of genomes from metagenomes. Even if a single sample is used for binning and there are no subpopulations, the resulting MAG is still a composite and not a strain in the traditional microbiological sense [46].

An obvious extension of our algorithm would be to resolve the accessory genome into strain genomes. This could be done on a per gene basis by relaxing the requirement that every strain passes through every gene, but an approach that incorporates the path structure in the full metagenomic assembly would be more powerful. Use of the full assembly may be possible in an efficient manner by factorising the variational approximation on a per gene basis and allowing the solutions for one gene to depend on the expectations across their neighbours. Or it may be that more computationally tractable versions of the algorithm can be developed that will scale to larger graphs. In any case the issues discussed above of our inferred 'strains' containing unresolved variation will become more pertinent when we extend our algorithm to the full genome, and it will be necessary to consider not just the most likely genome associated with a subpopulation but also its variants.

In the future we also plan to directly incorporate long read information into the strain resolution rather than just using it for validation. It was encouraging therefore to see the correspondence in strain frequencies between the two approaches. We are confident that in the near future, through the combination of long reads with methods similar to those we have introduced in STRONG, that complete metagenome de novo strain resolution, i.e. obtaining the complete genomes of all significant clonal subpopulations within a species, will become a realistic possibility.

Conclusion

We have introduced a complete bioinformatics pipeline, STrain Resolution ON assembly Graphs (STRONG), that is capable of extracting single-copy core gene variant graphs from short read metagenome coassemblies for individual metagenome assembled genomes (MAGs). We demonstrated how these graphs and associated per-sample unitig coverages can be used in a novel Bayesian algorithm, BayesPaths, to find MAG strain number, haplotypes and abundances. This approach achieves superior accuracy to variant based methods on synthetic communities and predictions on real data that match those from long Nanopore long reads.

Methods

Synthetic data set generation

The in silico synthetic communities were generated by first downloading a list of complete bacterial genomes from the NCBI and selecting species with multiple strains present. Genomes were restricted to those that were full genome projects, possessed at least 35 of 36 single-copy core genes (SCGs) identified in [3], and with relatively few contigs (< 5) in the assemblies. Communities were created by specifying species from this list and the number of strains desired. The strains selected were then chosen at random from the candidates for each species, with the extra restrictions that all strains in a species were at least 0.05% and no more than 5% nucleotide divergent on the SCGs from any other strain in the species. This corresponds to a minimum divergence of approximately 15 nucleotides over the roughly 30 kbp region formed by summing the SCGs. The genomes used are given in Additional file 1: Tables S1 and S2 and as a csv text file in Additional file 2.

Each species indexed i was then given an abundance, $y_{i,s}$, in each sample, $s = 1, \ldots, S$, which was drawn from a lognormal distribution with a species dependent mean and standard deviation, themselves drawn from a normal and gamma distribution respectively:

$$\log(y_{i,s}) \sim N(\mu_i, \sigma_i)$$

where:

$$\mu_i \sim N(\mu_p, \sigma_p)$$

and:

$$\sigma_i \sim \mathrm{Gamma}(k_p, \theta_p).$$

For all the synthetic communities we used $\mu_p = 1$, $\sigma_p = 0.125$, $k_p = 1$ and $\theta_p = 1$. The species abundances were then normalised to one ($y'_{i,s} = y_{i,s} / \sum_i y_{i,s}$). For each strain within a species its proportion in each sample was then drawn from a Dirichlet:

$$\rho_{g,s} \sim \mathrm{Dirichlet}(\alpha) \tag{1}$$

with $\alpha = 1$ and $g = 1, \ldots, G$ indexing G strains per species. This was the procedure used for all the data sets except Synth_T1_S10, Synth_T10_S10, and Synth_T100_S10 where we used a hierarchical Dirichlet first generating for each MAG an expected strain profile for all samples:

$$\mu_{i,g} \sim \text{Dirichlet}(\alpha) \tag{2}$$

with $\alpha = 1$ but then in each sample the strain proportions were generated as:

$$\rho_{i,s,g} \sim \text{Dirichlet}(\mu_i \theta), \tag{3}$$

so that the mean strain proportions were μ_i with an adjustable precision θ.

This allowed us to specify a copy number for each genome g in species i in each sample as $y'_{i,s}\rho_{g,s}$. We then generated 150 million paired-end 2x150 bp reads in total across all samples with Illumina HiSeq error distributions using the ART read simulator [23]. The code for the synthetic community generation is available from https://github.com/chrisquince/STRONG_Sim.

Synthetic data set evaluation

We can determine which contig derived from which reference genome by considering the simulated reads that map onto it. We know which reference each of these came from, enabling us to assign a contig to a genome as that which a majority of its reads derive from. We can then assign each MAG generated by STRONG to a reference species as the one which the majority of its contig's derive from weighted by the contig length.

Anaerobic digester sampling and sequencing

AD sample collection

We obtained ten samples from a farm anaerobic digestion bioreactor across a period of approximately one year. The sampling times, metadata and accession numbers are given in Additional file 1: Table S5. The reactor was fed on a mixture of slurry, whey and crop residues, and operated between 35-40°C, with mechanical stirring. Biomass samples were taken directly from the AD reactor by the facility operators and shipped in ice-cooled containers to the University of Warwick. Upon receipt, they were stored at 4°C and then sampled into several 1-5mL aliquots within a few days. DNA was usually extracted from these aliquots immediately but some were first stored in a -80°C freezer until subsequent thawing and extraction.

AD short read sequencing

DNA extraction was performed using the Qiagen Powersoil extraction kit following the manufacturer's protocol. DNA samples were sequenced by Novogene using the NovaSeq platform with 2x150 bp reads at a mean depth of 11.63 Gbp.

AD long read sequencing

Anaerobic digester samples were stored in 1.8 mL Cryovials at -80°C. Samples were defrosted at 4°C overnight prior to DNA extraction. DNA was extracted from a starting mass of 250 mg of anaerobic digester sludge using the MP Biomedical$^{\text{TM}}$ FastDNA$^{\text{TM}}$ SPIN Kit for Soil (cat no: 116560200) and a modified manufacturers protocol. Defrosted samples were homogenised by pipetting and then transferred to a MP bio$^{\text{TM}}$ lysing matrix E tube (cat no: 116914050-CF). Samples were resuspended in 938 μL of Sodium phosphate buffer (cat no: 116560205).

Preliminary cell lysis was undertaken using lysozyme at a final concentration of 200 $ng/\mu L$ and 20 μL of Molzyme Bug Lysis^{TM} solution. Samples were mixed by inversion and incubated at 37°C for 30 min on a shaking incubator (<100 rpm). Lysozyme was inactivated by adding 122 μL of MP bio MT buffer and mixing by inversion. Samples were then mechanically lysed in a VelociRuptor V2 bead beating machine (cat no: SLS1401) at 5 m/s for 20 seconds then placed on ice for five minutes.

Samples were centrifuged at 14000 g for five minutes to pellet the particulate matter and the supernatant was transferred to a new 1.5 mL microfuge tube. Proteins were precipitated from the crude lysate by adding 250 μL of PPS^{TM} (cat no: 116560203) and then mixing by inversion. Precipitated proteins were pelleted for five minutes at 14000 g and the supernatant was transferred to 1000 μL of pre mixed DNA binding matrix solution (cat no: 116540408). Samples were mixed by inversion for two minutes.

DNA binding matrix beads were recovered using the MP bio^{TM} spin filter (cat no: 116560210) and manufacturer based spin protocol. The binding matrix was washed of impurities by complete resuspension in 500 μL of SEWS-M solution (cat no: 116540405) and centrifuged at 14000 g for five minutes. The DNA binding matrix was then washed for a second time by resuspension in 500 μL of 80% EtOH followed by centrifugation at 14000 g for five minutes. Flow though was discarded and centrifuged at 14000 g for two minutes to remove residual EtOH. The binding matrix was left to air dry for 2 minutes then DNA was eluted using 100 μL of DES elution buffer at 56°C. Elute was collected by centrifugation at 14000 g for 5 minutes and stored at 4°C prior to library preparation. Eluted DNA concentration was estimated using a Qubit 4^{TM} fluorometer with the dsDNA Broad Range sensitivity assay kit (cat no: Q32853). 260:280 and 260:230 purity ratios were quantified using a Nanodrop^{TM} 2000.

A 1x SPRI clean up procedure was undertaken prior to library construction to further reduce contaminant carry over. Input DNA was standardised to 1.2 μg in 48 μL of H_2O using a qubit 4^{TM} fluorometer and dsDNA 1x High Sensitivity assay kit (cat no: Q33231). Library preparation was undertaken using the Oxford Nanopore© Ligation Sequencing Kit (SQK-LSK109) with minor modifications to the manufacturer protocol. The FFPE/End repair incubation step was extended to 30 min at 20°C and 30 min at 65°C, while DNA was eluted from SPRI beads at 37°C for 30 min with gentle agitation. The SQK-LSK109 long fragment buffer was used to ensure removal of non-ligated adaptor units and reduce short fragment carryover into the final sequencing library. The final library DNA concentration was standardised to 250 ng in 12 μL of EB using a qubit 4^{TM} fluorometer and dsDNA 1 x High Sensitivity assay kit.

Sequencing was undertaken for 72 hours on an Oxford Nanopore© R 9.4.1 (FLO-MIN106) flow cell with 1489 active pores. DNA was left to tether for 1 hour prior to commencing sequencing. The flow cell and sequencing reaction was controlled by a MinION^{TM} MKII device and the GUI MinKNOW V. 19.12.5. ATP refuelling was undertaken every 18 hours with 75 μL of flush buffer (FB). Post Hoc basecalling was undertaken using Guppy V. 3.5.1 and the high accuracy configuration (HAC) mode.

STRONG pipeline

STRONG processes co-assembly graph regions for multiple metagenomic datasets in order to simultaneously infer the composition of closely related strains for a particular MAG and their core gene sequences. Here, we provide an overview of STRONG. We start from a series of S related metagenomic samples, e.g. samples of the same (or highly similar) microbial community taken at different time points or from different locations.

The Snakemake based pipeline begins with the recovery of metagenome-assembled genomes (MAGs) [26]. We perform co-assembly of all available data with the metaSPAdes assembler [33], and then bin the contigs obtained by composition and coverage profiles across all available samples with CONCOCT [3] or alternatively Metabat2 [25]. Each bin is then analyzed for completeness and contamination based on single-copy core genes, and poor quality bins are discarded. The default criterion is that a MAG requires greater than or equal to 75% of the SCGs in a single copy. While we currently focus on this combination of software tools, in principle we could use any other software or pipeline for MAG recovery, e.g. we could use MEGAHIT as the primary assembler [29]. For each MAG we then extract the full or partial sequences of the core genes that we further refer to as single-copy core gene (SCG) sequences.

The final coassembly graph produced by metaSPAdes cannot be used for strain resolution because, as with other modern assembly pipelines, variants between closely related strains will be removed during the graph simplification process. Instead, we output the initial graph for the final K-mer length used in the (potentially) iterative assembly following processing by a custom executable — spades-gsimplifier based on the SPAdes codebase — to remove likely erroneous edges using a 'mild' coverage threshold and a tip clipping procedure. We refer to the resulting graph as a high-resolution assembly graph or HRAG.

The graph edges are then annotated with their corresponding sequence coverage profiles across all available samples. As is typical in de Bruijn graph analysis, the coverage values are given in terms of the k-mer rather than nucleotide coverage. Profile computation is performed by a second tool for aligning reads onto the HRAG: unitig-coverage. The potential advantage of this approach in comparison to estimation based on k-mer multiplicity, is that it can correctly handle the results of any bubble removal procedure that we might want to add to the preliminary simplification phase in future.

For each detected SCG sequence (across all MAGs) we next try to identify the subgraph of the HRAG encoding the complete sequences of all variants of the gene across all strains represented by the MAG. The procedure is described in more detail in the next section. During testing we faced two types of problems here: (1) related strains might end up in different MAGs and (2) some subgraphs might consist of fragments corresponding to several different species. We take several steps to mitigate those problems. Firstly, we compare SCG graphs between all bins, not just MAGs. If an SCG graph shares unitigs between bins, then it is flagged as overlapping. If multiple SCG graphs between MAGs (> 10) overlap then we merge those MAGs, combining all graphs and processing them for strains together. Following merging any MAG SCG graphs with overlaps remaining are filtered out and not used in the strain resolution.

After MAG merging and COG subgraph filtering we process the remaining MAGs one by one. Before the core 'decomposition' procedure is launched on the set of SCG subgraphs corresponding to a particular MAG, they are subjected to a second round of simplification, parameterised by the mean coverage of the MAG, to filter nodes that are likely to be noise again by the spades-gsimplifier program. This module is described in more detail below. The resulting set of simplified SCGs of the HRAG for a MAG are then passed to the core graph decomposition procedure, which uses the graph structure constraints, along with coverage profiles associated with unitig nodes, to simultaneously predict: the number of strains making up the population represented by the MAG; their coverage depths across the samples; paths corresponding to each strain within every subgraph (each path encodes a sequence of the particular SCG instance).

A fraction of the SCGs in a MAG may properly derive from other organisms due to the possibility of incorrect binning *i.e contamination*. In fact, the default 75% single-copy criterion allows up to 25% contamination. In addition, the subgraph extraction is not always perfect. We therefore add an extra level of filtering to the BayesPaths algorithm, iteratively running the program for all SCGs, but then filtering those with mean error rates, defined by Eq. 20, that exceed a default of 2.5 times the median deviation from the median gene error. Filtering on the median deviation is in general a robust strategy for identifying outliers. As a result of this filtering the pipeline only infers strain sequences on a subset of the input SCGs. We have found, however, that the number of SCGs for which strain haplotypes are inferred is sufficient for phylogenetics.

Relevant subgraph extraction

Provided with the predicted (partial) gene sequence, $\bar{T}$, and the upper bound on the length of the coding sequence, L, defined as $3\alpha\langle\bar{U}_n\rangle$ where $\langle\bar{U}_n\rangle$ is the average length in amino acids of that SCG in the COG database, and $\alpha = 1.5$ by default. The procedure for relevant HRAG subgraph extraction involves the following steps. First, the sequence $\bar{T}$ is split into two halves, $\bar{T}'$ and $\bar{T}''$, keeping the correct frame (both $\bar{T}'$ and $\bar{T}''$ are forced to divide by 3). $\bar{T}'$ and $\bar{T}''$ are then processed independently. Without loss of generality we describe the processing of $\bar{T}'$:

1. Identify the path $\mathcal{P}$ corresponding to $\bar{T}'$ in the HRAG. We denote its length as $L_{\mathcal{P}}$.
2. Launch a graph search of the stop codons to the right (left) of the rightmost (leftmost) position of $\bar{T}'$ ($\bar{T}''$). The stop codon search is frame aware and is performed by a depth-first search (DFS) on the graph in which each vertex corresponds to a pair of the HRAG position and the partial sequence of the last traversed codon. Due to the properties of the procedure and the fact that it deals with DBGs, the actual implementation encodes frame state as an integer [0,2] rather than the string of last partially traversed codon. Vertices of this 'state graph' are naturally connected following the HRAG constraints. The search is cut off whenever a vertex with a frame state encoding a stop codon sequence is identified. Several stop codons can be identified within the same HRAG edge sequence in 'different frames', moreover the procedure correctly identifies all stop codons even if the graph contains cycles (although such subgraphs may be ignored in later stages of the pipeline). This codon search procedure was originally implemented for https://github.com/ablab/orf-search and used in [15].
3. The 'backward' search of the stop codons 'to the left' is actually implemented as a 'forward' search of the complementary sequences from the complementary position in the graph. Note that, as in classic ORF analysis, while the identified positions of the stop codons 'to the right' correspond to putative ends of the coding sequences for some of the variants of the analyzed gene, positions of the stop codons 'to the left' only provide the likely boundary for where the coding sequence can start. In particular, left stop codons are likely to fall within the coding sequence of the neighbouring gene (in a different frame). Actual start codons are thus likely to lie somewhere on the path (with sequence length divisible by 3) between one of

the 'left' stop codons and one of the 'right' stop codons. For reasons of simplicity, further analysis of edges on the paths between left (right) stop codons ignores the constraint of divisibility by 3.

4. After the sets of 'left' and 'right' stop codon positions are identified along with the shortest distances between them and the $\bar{T}'$ path, we attempt to gather the relevant subgraph given by the union of edges lying on some path of a constrained length (see further) between some pair of left and right stop codons. First, for each pair (s, t) of the left and right stop codon positions we compute the maximal length of the paths that we want to consider $L_{s,t}$ as $L_{\mathcal{P}} + \text{minimal distance from s to start of} \mathcal{P} + \text{minimal distance from end of} \mathcal{P} \text{ to t}$. The edge $e = (v, w)$ is considered relevant if there exists a pair of left (right) stop codon positions (s', t') such that the edge e lies on the path of length not exceeding $L_{s,t}$ between s' and t', which is equivalent to checking that $\text{min\_dist}(s', v) + \text{length}(e) + \text{min\_dist}(w, t') < L_{s,t}$. To allow for efficient checks of the shortest distances we precompute them by launching the Dijkstra algorithm from all left (right) stop codon positions in the forward (backward) direction. In practice, the Dijkstra runs are initiated from the ends/starts of corresponding edges and the distances are later corrected.
5. We then exclude from the set of relevant edges the edges that are too far from any putative (right) stop codon to be a part of any COG instance. In particular, we exclude any edge $e = (v, w)$, such that the minimal distance from vertex w to any of the right stop codon positions exceeds L.
6. After the sets of the graph edges potentially encoding the gene sequence are gathered for $\bar{T}'$ and $\bar{T}''$ the union of the two sets, $\mathcal{ER}$, is then taken and augmented by the edges, connecting the 'inner' $\mathcal{ER}$ vertices (vertices which have at least one outgoing and at least one incoming edge in the $\mathcal{ER}$) to the rest of the graph.

Initial splitting of $\bar{T}$ into $\bar{T}'$ and $\bar{T}''$ is required to detect relevant stop codons which are not reachable from the last position of $\bar{T}$ in HRAG (or from which the first position of $\bar{T}$ in HRAG can not be reached). In addition to the resulting component in gfa format, we also store the positions of the putative stop codons, and ids of edges connecting the component to the rest of the graph (further referred to as 'sinks' and 'sources').

Subgraph simplification

While processing SCG subgraphs from a particular MAG we use the available information on the coverage of the MAG in the dataset. In particular, we set up the simplification module to remove tips (a node with either no successors or predecessors) below a certain length (twice the read length) and edges with coverage below a limit that is the larger of 2.0 or one per cent of the MAG coverage for regular edges or 2.5 and two per cent of the MAG coverage for short edges.

While simplifying a SCG subgraph, edges connecting it to the rest of the assembly graph should be handled with care (in particular, they should be excluded from the set of potential tips). This is because in the BayesPaths algorithm they form potential sources and sinks of the possible haplotype paths. Moreover, during the simplification the graph changes, and such edges might become part of longer edges. Since we are interested in which dead-ends of the component do, and do not lead to the rest of the graph, the output contains the up-to-date set of connections of the simplified component to the rest of the graph.

We now briefly describe additional, disabled by default procedures, based on 'relative coverage' criteria. Amongst other procedures for erroneous edge removal SPAdes implements a procedure considering the ratio of the edge coverage to the coverage of edges adjacent to it. We define an edge e as 'predominated' by vertex v incident to it if there is edge e_1 outgoing from v and edge e_2 incoming to v whose coverages exceed the coverage of e at least by a factor of α. Short edges (shorter than $k + \epsilon$) predominated by both vertices incident to them are then removed from the graph. Erroneous graph elements in high genomic coverage graph regions often form subgraphs of three or more erroneous edges. SPAdes implements a procedure for search (and subsequent removal) of subgraphs limited by a set of predominated edges. Starting from a particular edge (v, w) predominated by vertex v, the graph is traversed from vertex w breadth-first without taking into account the edge directions. If the vertex considered at the moment predominates the edge by which it was entered, the edges incident to it are not added to the traversal. The standard limitation of erroneous edge lengths naturally transforms into a condition of maximum length of the path between the vertices of the traversed subgraph. A limit on the maximum total length of its edges is additionally introduced.

BayesPaths

The model

We define an assembly graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ as a collection of unitig sequence vertices $\mathcal{V} = 1, \ldots, V$ and directed edges $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$. Each edge defines an overlap and comprises a pair of vertices and directions $\left(u^d \to v^d\right) \in \mathcal{E}$ where $d \in \{+, -\}$ and indicates whether the overlap occurs between the sequence $(+)$ or its reverse complement $(-)$. We define:

- Counts $x_{v,s}$ for each unitig $v = 1, \ldots, V$ in sample $s = 1, \ldots, S$
- Paths for strain $g = 1, \ldots, G$ defined by $\eta^g_{u,v} \in 0, 1$ indicating whether strain g passes through that edge in the graph
- Flow of strain g through unitig v, $\phi^{g+}_v = \sum_{u \in A(v)} \eta^g_{u,v}$ and $\phi^{g-}_v = \sum_{u \in D(v)} \eta^g_{v,u}$ where $A(v)$ is the set of ancestors of v and $D(v)$ descendants in the assembly graph
- The following is true $\phi^{g+}_v = \phi^{g-}_v = \phi^g_v$
- Strain intensities $\gamma_{g,s}$ as the rate per position that a read is generated from g in sample s
- Unitig lengths L_v
- Unitig bias θ_v is the fractional increase in reads generated from v given factors such as GC content influencing coverage
- Source node s and sink node t such that $\phi^{g+}_s = \phi^{g-}_s = \phi^{g+}_t = \phi^{g-}_t = 1$

Then assume normally distributed counts for each node in each sample giving a joint density for observations and latent variables:

$$P(\mathbf{X}, \boldsymbol{\Gamma}, \mathbf{H}, \boldsymbol{\Theta}) = \prod_{v=1}^{V}\prod_{s=1}^{S} \mathcal{N}\left(x_{v,s} \middle| L_v \theta_v \sum_{h=1}^{G} \phi^h_v \gamma_{h,s}, \tau^{-1}\right) \prod_{h=1}^{G}\prod_{s=1}^{S} P(\gamma_{h,s}|\lambda_h)$$
$$\cdot \prod_{h=1}^{G}\prod_{v=1}^{V} \left[\phi^{h+}_v = \phi^{h-}_v\right]\left[\phi^{h-}_s = 1\right]\left[\phi^{h+}_t = 1\right] P(\tau)$$
$$\cdot \prod_{h=1}^{G} P(\lambda_h|\alpha_0, \beta_0) \prod_{v=1}^{V} P(\theta_v|\mu_0, \tau_0) \qquad (4)$$

where [] indicates the Iverson bracket evaluating to 1 if the condition is true and zero otherwise. We assume an exponential prior for the $\gamma_{g,s}$ with a rate parameter that is strain dependent, such that:

$$P(\gamma_{g,s}|\lambda_g) = \lambda_g \exp(-\gamma_{g,s}\lambda_g) \tag{5}$$

We then place gamma hyper-priors on the λ_g:

$$P(\lambda_g|\alpha_0, \beta_0) = \frac{\beta_0^{\alpha_0}}{\Gamma(\alpha_0)}\lambda_g^{\alpha_0-1} \exp(-\beta_0\lambda_g) \tag{6}$$

This acts as a form of automatic relevance determination (ARD) forcing strains with low intensity across all samples to zero in every sample [8].

We use a standard Gamma prior for the precision:

$$P(\tau|\alpha_\tau, \beta_\tau) = \frac{\beta_\tau^{\alpha_\tau}}{\Gamma(\alpha_\tau)}\tau^{\alpha_\tau-1} \exp(-\beta_\tau\tau) \tag{7}$$

For the biases θ_v we use a truncated normal prior:

$$\begin{aligned} P(\theta_v|\mu_0, \tau_0) &= \frac{\sqrt{\frac{\tau_0}{2\pi}} \exp\left(-\frac{\tau_0}{2}(\theta_v - \mu_0)^2\right)}{1 - \Psi\left(-\mu_0\sqrt{\tau_0}\right)} \quad \theta_v >= 0 \\ &= 0 \quad \theta_v < 0 \end{aligned}$$

where Ψ is the standard normal cumulative distribution. The mean of this is set to one, $\mu_0 = 1$, so that our prior is that the coverage on any given node is unbiased, with a fairly high precision $\tau_0 = 100$, to reflect an assumption that the observed coverage should reflect the summation of the strains. Finally, we assume a uniform prior over the possible discrete values of the $\eta^g_{v,u}$. If the assembly graph is a directed acyclic graph (DAG) then $\eta^g_{v,u} \in 0, 1$. We have found that for most genes and typical kmer lengths this is true, but we do not need to assume it.

Variational Approximation

We use variational inference to obtain an approximate solution to the posterior distribution of this model [7]. Variational inference is an alternative strategy to Markov chain Monte Carlo (MCMC) sampling. Rather than attempting to sample from the posterior distribution, variational inference assumes a tractable approximating distribution for the posterior, and then finds the parameters for that distribution that minimise the Kullback-Leibler divergence between the approximation and the true posterior distribution. Further, in mean-field variational inference the approximation can be factorised into a product over a number of components that each approximate the posterior of a parameter in the true distribution. In practice the Kullback-Leibler divergence is not computable because it depends on the evidence, i.e. the joint distribution marginalised over all latent variables. Instead, inference is carried out by maximising the evidence lower bound (ELBO), which is equal to the negative of the Kullback-Leibler divergence plus a constant, that constant being the evidence. In our case, because all the distributions are conjugate we can employ CAVI, coordinate ascent variational inference, to iteratively maximise the ELBO.

Our starting point is to assume the following factorisation for the variational approximation:

$$q(\mathbf{X}, \boldsymbol{\Gamma}, \mathbf{H}) = \prod_{h=1}^{G} q_h\left(\left\{\eta_{v,u}^{h}\right\}_{u,v\in\mathcal{A}}\right) \prod_{h=1}^{G}\prod_{s=1}^{S} q_h(\gamma_{h,s}) \prod_{h=1}^{G} q_h(\lambda_h) \prod_{v=1}^{V} q_v(\theta_v) q(\tau) \tag{8}$$

where $\mathcal{A}$ is the set of edges in the assembly graph and $\mathcal{V} = 1, \ldots, V$ the set of unitig sequence vertices. Note that we have assumed a fully factorised approximation except for the $\eta_{v,u}^{h}$, the paths for each strain through the graph. There we assume that the path for each strain forms a separate factor allowing strong correlations between the different elements of the path. This is therefore a form of structured variational inference [22].

To obtain the CAVI updates we use the standard procedure of finding the log of the optimal distributions q for each set of factorised variables as the expectation of the log joint distribution Eq. 4 over all the other variables, except the one being optimised. Using an informal notation we will denote these expectations as $\langle \ln P \rangle_{-q_j}$ where q_j is the variable being optimised.

Then the mean field update for each set of $\{\eta_{v,u}^{g}\}_{u,v\in A}$ is derived as:

$$\begin{aligned}
\ln q_g^*\left(\left\{\eta_{v,u}^{g}\right\}_{u,v\in A}\right) &= \langle \ln P \rangle_{-\eta_{v,u}^{g} u,v\in\mathcal{A}} \\
&= \ln\left(\prod_{v=1}^{V} \delta_{\phi_v^{g+},\phi_v^{g-}} \delta_{\phi_s^{g-},1} \delta_{\phi_t^{g+},1}\right) \\
&\quad - \left\langle \sum_{v=1}^{V}\sum_{s=1}^{S} \frac{\tau}{2}\left(x_{v,s} - \theta_v L_v\left[\sum_{h=1}^{G}\phi_v^{h}\gamma_{h,s}\right]\right)^2\right\rangle_{-\eta_{v,u}^{g} u,v\in\mathcal{A}} \\
&\quad + \text{Terms independent of} \eta^{g}
\end{aligned}$$

Consider the second term only:

$$-\frac{\langle\tau\rangle}{2}\left(-\sum_{v=1}^{V}\sum_{s=1}^{S} 2x_{v,s}L_v\langle\theta_v\rangle\langle\gamma_{g,s}\rangle\phi_v^{g} + L_v^2\left\langle{\theta_v}^2\right\rangle\left\langle\left(\sum_{h=1}^{G}\phi_v^{h}\gamma_{h,s}\right)\left(\sum_{i=1}^{G}\phi_v^{i}\gamma_{i,s}\right)\right\rangle\right)$$

This becomes:

$$\begin{aligned}
-\frac{\langle\tau\rangle}{2}\Bigg(\sum_{v=1}^{V}\sum_{s=1}^{S}\Bigg[&- 2x_{v,s}\langle\theta_v\rangle L_v\langle\gamma_{g,s}\rangle\phi_v^{g} + 2L_v^2\left\langle{\theta_v}^2\right\rangle\sum_{h\neq g}^{G}\left\langle\phi_v^{h}\right\rangle\langle\gamma_{h,s}\rangle\langle\gamma_{g,s}\rangle\phi_v^{g} \\
&+ L_v^2\left\langle{\theta_v}^2\right\rangle\left\langle\gamma_{g,s}^2\right\rangle\left(\phi_v^{g}\right)^2\Bigg]\Bigg)
\end{aligned}$$

Which can be reorganised to:

$$\ln q_g^*\left(\left\{\eta_{v,u}^{g}\right\}_{u,v\in A}\right) = \ln\left(\prod_{v=1}^{V}\delta_{\phi_v^{g+},\phi_v^{g-}}\delta_{\phi_s^{g-},1}\delta_{\phi_t^{g+},1}\right) + \sum_{v=1}^{V} c_{1,v}\phi_v^{g} + \sum_{v=1}^{V} c_{2,v}\left(\phi_v^{g}\right)^2 \tag{9}$$

Where:

$$\begin{aligned}
c_{1,v} &= -\frac{\langle\tau\rangle}{2}\sum_{s=1}^{S}\left[-2x_{v,s}\langle\theta_v\rangle L_v\langle\gamma_{g,s}\rangle + 2L_v^2\left\langle{\theta_v}^2\right\rangle\sum_{h\neq g}^{G}\left\langle\phi_v^{h}\right\rangle\langle\gamma_{h,s}\rangle\langle\gamma_{g,s}\rangle\right] \\
c_{2,v} &= -\frac{\langle\tau\rangle}{2}L_v^2\left\langle{\theta_v}^2\right\rangle\left\langle\gamma_{g,s}^2\right\rangle
\end{aligned}$$

It is apparent from Eq. 9 that the $q_g^*\left(\left\{\eta_{v,u}^{g}\right\}_{u,v\in A}\right)$ takes the form of a multivariate discrete distribution with $|u,v \in A|$ dimensions. The first term in Eq. 9 enforces the flow constraints, and does not separate across nodes, the next two terms are effectively coefficients on the total flow through a unitig and its square. The updates for the other variables below, depend on the expected values of the total flow through each of the unitig nodes for the strain g, $\langle \phi_v^g \rangle$, which themselves depend on the $\eta_{v,u}^g$. These expected values can be efficiently obtained for all v by representing Eq. 9 as a factor graph comprising nodes consisting of factors corresponding to both the constraints and the flow probabilities through each node with variables $\eta_{v,u}^g$. We can then find the marginal probabilities for both the $\eta_{v,u}^g$ and the ϕ_v^g using the Junction Tree algorithm [48], from these we can calculate the required expectations.

Next we consider the mean field update for the $\gamma_{g,s}$:

$$\begin{aligned}\ln q^*(\gamma_{g,s}) &= \langle \ln P \rangle_{-\gamma_{g,s}} \\ &= -\left\langle \sum_{v=1}^{V} \frac{\tau}{2} \left(x_{v,s} - \theta_v L_v \left[\sum_{h=1}^{G} \phi_v^h \gamma_{h,s} \right] \right)^2 \right\rangle_{-\gamma_{g,s}} - \langle \lambda_g \rangle \gamma_{g,s}\end{aligned}$$

$$\begin{aligned}\ln q^*(\gamma_{g,s}) = \\ & - \frac{\langle \tau \rangle}{2} \Bigg(\sum_{v=1}^{V} -2 x_{v,s} \langle \theta_v \rangle L_v \left\langle \phi_v^g \right\rangle \gamma_{g,s} + 2 \left\langle \theta_v^2 \right\rangle L_v^2 \gamma_{g,s} \left\langle \phi_v^g \right\rangle \sum_{h \neq g} \langle \gamma_{h,s} \rangle \left\langle \phi_v^h \right\rangle \\ & + \left\langle \theta_v^2 \right\rangle L_v^2 \gamma_{g,s}^2 \left\langle \left[\phi_v^g \right]^2 \right\rangle \Bigg) \\ & - \langle \lambda_g \rangle \gamma_{g,s}\end{aligned}$$

with the restriction $\gamma_{g,s} > 0$ this gives a normal distribution but truncated to $(0, \inf)$ for $\gamma_{g,s}$, with mean and precision:

$$\mu_{g,s} = \frac{\sum_v x_{v,s} \theta_v L_v \left\langle \phi_v^g \right\rangle - \left\langle \phi_v^g \right\rangle \sum_{h \neq g} \langle \gamma_{h,s} \rangle \left\langle \phi_v^h \right\rangle \left\langle \theta_v^2 \right\rangle L_v^2}{\sum_v L_v^2 \left\langle \left[\phi_v^g \right]^2 \right\rangle} - \frac{\langle \lambda_g \rangle}{\tau_{g,s}} \tag{10}$$

$$\tau_{g,s} = \langle \tau \rangle \sum_v L_v^2 \left\langle \left[\phi_v^g \right]^2 \right\rangle \tag{11}$$

$$\tag{12}$$

Derivations for the other updates follow similarly giving a Gamma posterior for the τ with parameter updates:

$$\alpha_\tau = \alpha_0 + \Omega/2 \tag{13}$$

$$\beta_\tau = \beta_0 + \sum_{v,s} \left\langle (x_{v,s} - \lambda_{v,s})^2 \right\rangle \tag{14}$$

where $\Omega = VS$ and we have used $\lambda_{v,s}$ as a short hand for the predicted count number:

$$\lambda_{v,s} = \theta_v \sum_g \gamma_{g,s} \phi_v^g.$$

Then the τ have the following expectations and log expectations:

$$\langle \tau_{v,s} \rangle = \alpha_\tau / \beta_\tau \tag{15}$$

$$\langle \log \tau_{v,s} \rangle = \psi\left(\alpha_\tau + 1/2\right) - \log\left(\beta_\tau\right) \tag{16}$$

where ψ is the digamma function. The biases θ_v have a truncated normal distribution and their updates can be derived similar to the above.

Evidence lower bound (ELBO)

Iterating the CAVI updates defined above will generate a variational approximation that is optimal in the sense of maximising the evidence lower bound (ELBO) so called because it bounds the log evidence, $log(p(x)) \geq ELBO(q(z))$. It is useful to calculate the ELBO whist performing CAVI updates to verify convergence and the ELBO itself is sometimes used as a Bayesian measure of model fit, although as a bound that may be controversial [7]. The ELBO can be calculated from the relationship:

$$ELBO(q) = \mathbb{E}\left[\log p(x|z)\right] + \mathbb{E}\left[\log p(z)\right] - \mathbb{E}\left[\log q(z)\right] \tag{17}$$

The first term is simply the expected log-likelihood of the data given the latent variables. In our case it is:

$$\mathbb{E}\left[\log p(x|z)\right] = \frac{1}{2}\Omega\left(\langle\log\tau\rangle - \log(2\pi)\right) - \frac{1}{2}\langle\tau\rangle\left\langle\left(x_{v,s} - L_v\theta_v\sum_{h=1}^{G}\phi_v^h\gamma_{h,s}\right)^2\right\rangle \tag{18}$$

where $\Omega = VS$ and the expectations are over the optimised distributions q.

The second term is the expectation under $q(z)$ of the log prior distributions. In our case with standard distributions it is easy to calculate for instance for each of the $\gamma_{g,s}$:

$$\mathbb{E}\left[\log p(\gamma_{g,s})\right] = \frac{1}{2}\log\left(\frac{\tau_{g,s}}{2\pi}\right) - \frac{\tau_{g,s}}{2}\left(\left\langle\gamma_{g,s}^2\right\rangle + \mu_{g,s}^2 - 2\mu_{g,s}\langle\gamma_{g,s}\rangle\right) - \log\left[\frac{1}{2}\mathrm{erf}\left(\mu_{g,s}\sqrt{\frac{\tau_{g,s}}{2}}\right)\right].$$

With the $\mu_{g,s}$ and $\tau_{g,s}$ given by their current values derived from Eq. 12 and the moments of $\gamma_{g,s}$ calculated from a truncated normal distribution with those current parameters. The third terms are simply the negative entropy of the variational approximations and for the standard distributions used here are easily calculated.

Implementation details

One update of the algorithm consists of updating each variable or sets of variables in turn given the current optimal solutions of the other distributions. In practice we update:

- Compute the marginal flows $\{\eta_{v,u}^g\}_{u,v\in A}$ for each strain $g = 1, \ldots, G$ in turn using Eq. 9 and the Junction Tree algorithm. This can be performed for each single copy-core gene independently
- Update the truncated normal strain abundances $q(\gamma_{g,s})$ for each strain in each sample, $s = 1, \ldots, S$ using Eq. 12
- Update the $q(\tau)$
- Update the ARD parameter distributions $q(\lambda_g)$ if used
- Update the nodes biases $q(\theta_v)$
- Check for redundant or low abundance strains and remove (see below)

After a maximum fixed number of iterations or if the ELBO converges we stop iterating. Variational inference can be sensitive to initial conditions as it can only find local maxima of the ELBO, we therefore use a previously published variational Bayesian version of non-negative matrix factorisation [8], to find an initial approximate solution.

Empirical modelling of node variances

For low-coverage MAGs a precision that is identical for all nodes performs satisfactorily, but since the true distribution of read counts is Poisson this overestimates the precision for nodes with high counts $x_{v,s}$. To address we developed an empirical procedure where we first calculate $\langle \log \tau_{v,s} \rangle$ for each node using Eq. 16 as:

$$\langle \log \tau_{v,s} \rangle = \psi\left(\alpha_0 + 1/2\right) - \log\left(\beta_0 X_{v,s} + \frac{1}{2}\left\langle (x_{v,s} - \lambda_{v,s})^2 \right\rangle\right) \tag{19}$$

a quantity which exhibits high variability, so we then smooth this over $\log(x_{v,s})$ using generalised additive models as implemented in pyGAM [43] to give $\langle \log \tau_{v,s} \rangle^*$. The term $\beta_0 X_{v,s}$ gives a prior which is effectively Poisson. We then obtain $\langle \tau_{v,s} \rangle$ as $\exp(\langle \log \tau_{v,s} \rangle^*)$. This procedure has no theoretical justification but gives good results in practice. This approach of modelling a non-Gaussian distribution as a Gaussian with empirical variances is similar to that used in voom for RNASeq [27].

Cross-validation to determine optimum number of strains

The ARD procedure usually converges on the correct number of strains except for high-coverage MAGs where overfitting may occur and too many strains can be predicted. We therefore additionally implemented a cross-validation procedure, splitting the entire data matrix $x_{v,s}$ into test and train folds (default ten folds) and training the model on the train fold and then testing on the held out data. The correct number of strains was then taken to be the one that maximised the log predictive posterior with an empirical variance reflecting the Poisson nature of the data. The exact test statistic being:

$$\sum_{v,s \in E} \frac{1}{2} \log\left(\tau'_{v,s}\right) - \frac{1}{2} \sum_{v,s \in E} \tau'_{v,s} \left\langle (x_{v,s} - \lambda_{v,s})^2 \right\rangle \tag{20}$$

where $\tau'_{v,s} = 1/(0.5 + x_{v,s})$ and E indicates data points in the test set to down-weight high read count nodes reflecting approximately Poisson noise.

Nanopore sequence analysis

Sequence preprocessing

To enable a qualitative comparison between haplotypes obtained from the Nanopore reads and the BayesPaths predictions we developed the following pipeline applied at the level of individual single-copy core genes (SCGs) from MAGs:

1. We mapped all reads using minimap2 [30] against the SCG contig consensus ORF sequence and selected those reads with alignments that spanned at least one third of the gene with a nucleotide identity >80%.
2. We then extracted the aligned portion of each read, reverse complementing if necessary, so that all reads ran in the same direction as the SCG ORF.
3. We then obtained the variant positions on the consensus from the output of the DESMAN pipeline [39]. These are variant positions prior to haplotype calling representing the total unlinked variation observed in the short reads.
4. For each Nanopore fragment we aligned against the SCG ORF using a Needleman-Wunsch global alignment and generated a condensed read comprising bases only from the short read variant positions.

This provided us with a reduced representation of each Nanopore read effectively filtering variation that was not observed in the short reads. These reduced representations were then used to calculate distances, defined as Hamming distances on the variant positions normalised by number of positions observed, both between the reads and between the reads and the predicted COG sequences from BayesPaths. From these we generated NMDS plots indicating sequence diversity, and they provided an input to the hybrid Nanopore strain resolution algorithm below.

EM algorithm for hybrid Nanopore strain resolution

We also developed a simple EM algorithm for direct prediction of paths and their abundances on the short read assembly graph that are consistent with a set of observed long reads. We began by mapping the set of $n = 1, \ldots, N$ Nanopore sequences denoted $\{\mathcal{S}_n\}$ onto the corresponding simplified SCG graph generated by STRONG using GraphAligner [40]. This provided us with N optimal alignment paths as walks in our SCG graph. We denote this graph $\mathcal{G}$ comprising unitig vertices v and edges $e \in \{u, v\}$ defining overlaps.

We assume, as is almost always the case that the graphs contain no unitigs in both forward and reverse configurations, and that there are no cycles, so that each SCG is a directed acyclic graph (DAG) with one copy of each unitig, and we only need to track the direction that each overlap enters and leaves each unitig. Then best alignment walks comprise a sequence of edges, $e^n_1, \ldots, e^n_{W_n}$ where W_n is the number of edges in the walk of read n, that traverse the graph.

Given these observed Nanopore reads we aim to reconstruct G haplotypes comprising paths from a dummy source node s, which has outgoing edges to all the true source nodes in the graph, through the graph to a dummy sink t, which connects all the true sinks. We further assume that each haplotype has relative frequency π_g. Each such haplotype path $\mathcal{P}_g = \left\{s, e^g_1, \ldots, e^g_{W_g}, t\right\}$ will translate into a nucleotide sequence $\mathcal{T}_g$. We assume that these haplotypes generate Nanopore reads with a fixed error rate ϵ which gives a likelihood:

$$P(\{\mathcal{S}_1, \ldots, \mathcal{S}_N\}|\pi, \{\mathcal{T}_1, \ldots, \mathcal{T}_G\}) = \prod_{n=1}^{N} \left(\sum_{g=1}^{G} \pi_g \epsilon^{m_{n,g}} (1-\epsilon)^{M_{n,g}} \right). \tag{21}$$

where $m_{n,g}$ is the number of basepair mismatches between $\mathcal{S}_n$ and $\mathcal{T}_g$ counting insertions, deletions and substitutions equally and $M_{n,g}$ the number of matches.

To maximise this likelihood we used an Expectation-Maximisation algorithm. Iterating the following steps until convergence:

1. E-step: Calculate the responsibility of each haplotype for each sequence as:

$$z_{n,g} = \frac{\pi_g \epsilon^{m_{n,g}} (1-\epsilon)^{M_{n,g}}}{\sum_h \pi_h \epsilon^{m_{n,h}} (1-\epsilon)^{M_{n,h}}} \tag{22}$$

   Alignments of reads against haplotypes were performed using vsearch [41].

2. M-step: We update each haplotype by finding the most likely path on the short read graph given the current expectations. These are calculated by assigning a weight w^g_e to each edge e in the graph as:

$$w^g_e = \sum_{n \in e} z_{n,g} L_e \tag{23}$$

   where $n \in e$ are the set of reads whose optimal alignment contains that edge and L_e is the unique length of the unitig the edge connects to, i.e. ignoring the overlap length. We then find for haplotype g the maximal weight path through this DAG using a topological sort. The error rates are updated as:

$$\epsilon = \frac{\sum_n \sum_g z_{n,g} m_{n,g}}{\sum_n \sum_g z_{n,g} L_{n,g}} \tag{24}$$

where $L_{n,g}$ are the alignment lengths.

As is often the case with EM algorithms convergence depends strongly on initial conditions. Therefore we initialise using a partitioning around medoids clustering using the distances calculated in “Methods - Nanopore sequence analysis” section. We can estimate the number of haplotypes from the negative log-likelihood as a function of haplotype number.

Supplementary Information

The online version contains supplementary material available at https://doi.org/10.1186/s13059-021-02419-7.

Additional file 1: Supplementary tables and figures.

Additional file 2: Genomes used in the synthetic communities (Additional file 1: Tables S1 and S2).

Additional file 3: Review history.

Acknowledgments
We thank Tatiana Dvorkina (PhD student in SPbSU) for developing the graph-based codon finding algorithm and fruitful discussions. We also acknowledge constructive criticisms from two anonymous reviewers.

Peer review information
Andrew Cosgrove was the primary editor of this article and managed its editorial process and peer review in collaboration with the rest of the editorial team.

Review history
The review history is available as Additional file 3.

Authors’ contributions
CQ devised and coded the BayesPaths algorithm, assisted with the creation of the STRONG pipeline, analysed results and wrote the MS. SN devised and coded the graph algorithms, assisted with the creation of the STRONG pipeline, and edited the MS. SR coded the STRONG pipeline. RJ generated and processed the AD Nanopore sequence data. OSS helped devise the AD sequencing study. JKS assisted with the STRONG pipeline and edited the MS. AL contributed to the creation of the algorithms and edited the MS. AME contributed to the creation of the algorithms and assisted with figures. RC tested the STRONG pipeline, contributed to the creation of the algorithms and edited the MS. AED helped plan the STRONG pipeline, assisted the creation of the algorithms and edited the MS. All authors read and approved the final manuscript.

Funding
This work for made possible through the MRC Methodology Grant ‘Strain resolved metagenomics for medical microbiology’ MR/S037195/1. CQ is also funded through MRC fellowship (MR/M50161X/1) as part of the CLoud Infrastructure for Microbial Genomics (CLIMB) consortium (MR/L015080/1). SR is funded through BBSRC ‘EBI Metagenomics - enabling the reconstruction of microbial populations’ (BB/R015171/1). OSS acknowledges funding through the BBSRC (BB/N023285/1 and BB/L502029/1). This research was supported in part by the Intramural Research Program of the National Human Genome Research Institute, National Institutes of Health (SN).

Availability of data and materials
The anaerobic digester time series have been uploaded to the ENA as part of the study PRJEB45779 [19]. STRONG is freely available from https://github.com/chrisquince/STRONG under the MIT License. The BayesPaths algorithm at https://github.com/chrisquince/BayesPaths and code for synthetic community generation at https://github.com/chrisquince/STRONG_Sim. The code in the archived version [34] was used to generate the benchmarking results in this paper.

Declarations

Ethics approval and consent to participate
Not applicable.

Consent for publication
Not applicable.

Competing interests
Aaron Darling is a cofounder of Longas Technologies Pty Ltd, a company that is developing synthetic long read sequencing technologies.

Author details
[1]Organisms and Ecosystems, Earlham Institute, Norwich NR4 7UZ, UK. [2]Gut Microbes and Health, Quadram Institute, Norwich NR4 7UQ, UK. [3]Warwick Medical School, University of Warwick, Coventry CV4 7AL, UK. [4]Genome Informatics Section, Computational and Statistical Genomics Branch, National Human Genome Research Institute, National Institutes of Health, Bethesda 20892, MD, USA. [5]School of Life Sciences, University of Warwick, Coventry CV4 7AL, UK. [6]Univ. Lille, CNRS, Inria, UMR 9189 - CRIStAL, Lille, France. [7]Department of Medicine, University of Chicago, Chicago, Illinois, USA. [8]Josephine Bay Paul Center, Marine Biological Laboratory, Woods Hole, Massachusetts, USA. [9]Department of Computational Biology, Institut Pasteur, C3BI USR 3756 IP CNRS, Paris, France. [10]The iThree institute, University of Technology Sydney, 15 Broadway, Ultimo 2007, NSW, Australia.

Received: 22 August 2020 Accepted: 29 June 2021
Published online: 26 July 2021

References

1. Ahn T-H, Chai J, Pan C. Sigma: strain-level inference of genomes from metagenomic analysis for biosurveillance. Bioinformatics. 2014;31(2):170–7.
2. Albertsen M, Hugenholtz P, Skarshewski A, Nielsen KL, Tyson GW, Nielsen PH. Genome sequences of rare, uncultured bacteria obtained by differential coverage binning of multiple metagenomes. Nat Biotechnol. 2013;31: 533.
3. Alneberg J, Bjarnason BS, de Bruijn I, Schirmer M, Quick J, Ijaz UZ, Lahti L, Loman NJ, Andersson AF, Quince C. Binning metagenomic contigs by coverage and composition. Nat Methods. 2014;11(11):1144–6.
4. Baaijens JA, Van der Roest B, Köster J, Stougie L, Schönhuth A. Full-length de novo viral quasispecies assembly through variation graph construction. Bioinformatics. 2019;35(24):5086–94.
5. Bernard E, Jacob L, Mairal J, Vert J-P. Efficient RNA isoform identification and quantification from RNA-Seq data with network flows. Bioinformatics. 2014;30(17):2447–55.
6. Bernard E, Jacob L, Mairal J, Viara E, Vert J-P. A convex formulation for joint RNA isoform detection and quantification from multiple RNA-seq samples. BMC Bioinformatics. 2015;16(1):262.
7. Blei DM, Kucukelbir A, McAuliffe JD. Variational inference: a review for statisticians. J Am Stat Assoc. 2017;112(518): 859–77.
8. Brouwer T, Frellsen J, Lió P. Comparative study of inference methods for bayesian nonnegative matrix factorisation. In: Ceci M, Hollmén J, Todorovski L, Vens C, Džeroski S, editors. Machine learning and knowledge discovery in databases. Cham: Springer International Publishing; 2017. p. 513–29.
9. Brown CT, Hug LA, Thomas BC, Sharon I, Castelle CJ, Singh A, Wilkins MJ, Wrighton KC, Williams KH, Banfield JF. Unusual biology across a group comprising more than 15% of domain bacteria. Nature. 2015;523:208.
10. Brown CT, Moritz D, O'Brien MP, Reidl F, Reiter T, Sullivan BD. Exploring neighborhoods in large metagenome assembly graphs using spacegraphcats reveals hidden sequence diversity. Genome Biol. 2020;21(1):164.
11. Chaumeil P-A, Mussig AJ, Hugenholtz P, Parks DH. GTDB-Tk: a toolkit to classify genomes with the Genome Taxonomy Database. Bioinformatics. 2019;36(6):1925–7.
12. Costea PI, Zeller G, Sunagawa S, Pelletier E, Alberti A, Levenez F, Tramontano M, Driessen M, Hercog R, Jung F-E, Kultima JR, Hayward MR, Coelho LP, Allen-Vercoe E, Bertrand L, Blaut M, Brown JRM, Carton T, Cools-Portier S, Daigneault M, Derrien M, Druesne A, de Vos WM, Finlay BB, Flint HJ, Guarner F, Hattori M, Heilig H, Luna RA, van Hylckama Vlieg J, Junick J, Klymiuk I, Langella P, Le Chatelier E, Mai V, Manichanh C, Martin JC, Mery C, Morita H, O'Toole PW, Orvain C, Patil KR, Penders J, Persson S, Pons N, Popova M, Salonen A, Saulnier D, Scott KP, Singh B, Slezak K, Veiga P, Versalovic J, Zhao L, Zoetendal EG, Ehrlich SD, Dore J, Bork P. Towards standards for human fecal sample processing in metagenomic studies. Nat Biotechnol. 2017;35(11):1069–76.
13. Delmont TO, Quince C, Shaiber A, Esen ÖC, Lee STM, Rappé MS, McLellan SL, Lücker S, Eren AM. Nitrogen-fixing populations of Planctomycetes and Proteobacteria are abundant in surface ocean metagenomes. Nat Microbiol. 2018;3(7):804–13.
14. Dijkshoorn L, Ursing BMÃ, Ursing JB. Strain, clone and species: comments on three basic concepts of bacteriology. J Med Microbiol. 2000;49:397–401.
15. Dvorkina T, Bankevich A, Sorokin A, et al. ORFograph: search for novel insecticidal protein genes in genomic and metagenomic assembly graphs. Microbiome. 2021;9:149.
16. Eren A, Esen O, Quince C, Vineis J, Morrison H, Sogin M, Delmont T. Anvi'o: an advanced analysis and visualization platform for 'omics data. Peer J. 2015;3:e1319.
17. Eren AM, Kiefl E, Shaiber A, Veseli I, Miller SE, Schechter MS, Fink I, Pan JN, Yousef M, Fogarty EC, Trigodet F, et al. Community-led, integrated, reproducible multi-omics with anvi'o. Nat Microbiol. 2021;6(1):3–6.
18. Lander ES, Waterman MS. Genomic mapping by fingerprinting random clones: a mathematical analysis. Genomics. 1988;2:231–9.
19. Farrell F, James R, Raguideau S, Quince C, Soyer OS. Metagenome sequence from AD reactors used in the STRONG publication. Eur Nucleotide Arch. 2021PRJEB45779. https://www.ebi.ac.uk/ena/browser/view/PRJEB45779.
20. Garrison E, Marth G. arXiv e-prints, arXiv:1207.3907. 2012.
21. Garrison E, Sirén J, Novak AM, Hickey G, Eizenga JM, Dawson ET, Jones W, Garg S, Markello C, Lin MF, Paten B, Durbin R. Variation graph toolkit improves read mapping by representing genetic variation in the reference. Nat Biotechnol. 2018;36(9):875–9.
22. Hoffman M, Blei D. Stochastic structured variational inference. In: Lebanon G, Vishwanathan SVN, editors. Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics, vol. 38 of Proceedings of Machine Learning Research. San Diego: PMLR; 2015. p. 361–369.
23. Huang W, Li L, Myers JR, Marth GT. ART: a next-generation sequencing read simulator. Bioinformatics. 2011;28(4): 593–4.
24. Kang D, Froula J, Egan R, Wang Z. Metabat, an efficient tool for accurately reconstructing single genomes from complex microbial communities. Peer J. 2015;3:e1165.

25. Kang DD, Li F, Kirton E, Thomas A, Egan R, An H, Wang Z. MetaBAT 2: an adaptive binning algorithm for robust and efficient genome reconstruction from metagenome assemblies. Peer J. 2019;7:e7359.
26. Köster J, Rahmann S. Snakemake—a scalable bioinformatics workflow engine. Bioinformatics. 2012;28(19):2520–2.
27. Law CW, Chen Y, Shi W, Smyth GK. voom: precision weights unlock linear model analysis tools for rna-seq read counts. Genome Biol. 2014;15:R29.
28. Leimbach A, Hacker J, Dobrindt U. *E. coli* as an all-rounder: the thin line between commensalism and pathogenicity. Berlin: Springer Berlin Heidelberg; 2013, pp. 3–32.
29. Li D, Liu C-M, Luo R, Sadakane K, Lam T-W. MEGAHIT: an ultra-fast single-node solution for large and complex metagenomics assembly via succinct de Bruijn graph. Bioinformatics. 2015;31(10):1674–6.
30. Li H. Minimap2: pairwise alignment for nucleotide sequences. Bioinformatics. 2018;34(18):3094–100.
31. Li X, Hu H, Li X. mixtureS: a novel tool for bacterial strain genome reconstruction from reads. Bioinformatics. 2020;37(4):575–7.
32. Luo C, Knight R, Siljander H, Knip M, Xavier RJ, Gevers D. Constrains identifies microbial strains in metagenomic datasets. Nat Biotechnol. 2015;33:1045.
33. Nurk S, Meleshko D, Korobeynikov A, Pevzner P. metaspades: a new versatile metagenomic assembler. Genome Res. 2017;27:824–34.
34. Nurk S, Raguideau S, Summers K, Quince C. STRONG - Strain Resolution ON Graphs. GitHub. 2021.
35. O'Brien JD, Didelot X, Iqbal Z, Amenga-Etego L, Ahiska B, Falush D. A bayesian approach to inferring the phylogenetic structure of communities from metagenomic data. Genetics. 2014;197(3):925–37.
36. Olm MR, Crits-Christoph A, Bouma-Gregson K, Firek BA, Morowitz MJ, Banfield JF. inStrain profiles population microdiversity from metagenomic data and sensitively detects shared microbial strains. Nat Biotechnol. 2021;39(6): 727–36.
37. Pachiadaki MG, Brown JM, Brown J, Bezuidt O, Berube PM, Biller SJ, Poulton NJ, Burkart MD, La Clair JJ, Chisholm SW, Stepanauskas R. Charting the complexity of the marine microbiome through single-cell genomics. Cell. 2019;179(7):1623–1635.e11.
38. Pasolli E, Asnicar F, Manara S, Zolfo M, Karcher N, Armanini F, Beghini F, Manghi P, Tett A, Ghensi P, Collado MC, Rice BL, DuLong C, Morgan XC, Golden CD, Quince C, Huttenhower C, Segata N. Extensive unexplored human microbiome diversity revealed by over 150,000 genomes from metagenomes spanning age, geography, and lifestyle. Cell. 2019;176(3):649–662.e20.
39. Quince C, Delmont TO, Raguideau S, Alneberg J, Darling AE, Collins G, Eren AM. Desman: a new tool for de novo extraction of strains from metagenomes. Genome Biol. 2017;18(1):181.
40. Rautiainen M, Mäkinen V, Marschall T. Bit-parallel sequence-to-graph alignment. Bioinformatics. 2019;35(19): 3599–607.
41. Rognes T, Flouri T, Nichols B, Quince C, Mahé F. Vsearch: a versatile open source tool for metagenomics. Peer J. 2016;4:e2584.
42. Segata N. On the road to strain-resolved comparative metagenomics. mSystems. 2018;3(2).
43. Servén D, Brummitt C. pygam: generalized additive models in python. Zenodo. 2018.
44. Tatusov RL, Galperin MY, Natale DA, Koonin EV. The COG database : a tool for genome-scale analysis of protein functions and evolution. Nucl Acid Res. 2000;28(1):33–6.
45. Truong DT, Tett A, Pasolli E, Huttenhower C, Segata N. Microbial strain-level population structure and genetic diversity from metagenomes. Genome Res. 2017;27(4):626–38.
46. Van Rossum T, Ferretti P, Maistrenko OM, Bork P. Diversity within species: interpreting strains in microbiomes. Nat Rev Microbiol. 2020;18(9):491–506.
47. Vos M, Didelot X. A comparison of homologous recombination rates in bacteria and archaea. ISME J. 2009;3(2): 199–208.
48. Wainwright MJ, Jordan MI. Graphical models, exponential families, and variational inference. Found Trends Mach Learn. 2008;1(1-2):1–305.
49. Zhou Z, Luhmann N, Alikhan N-F, Quince C, Achtman M. Accurate reconstruction of microbial strains from metagenomic sequencing using representative reference genomes. In: Raphael BJ, editor. Research in Computational Molecular Biology. Cham: Springer International Publishing; 2018. p. 225–40.

Publisher's Note

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.