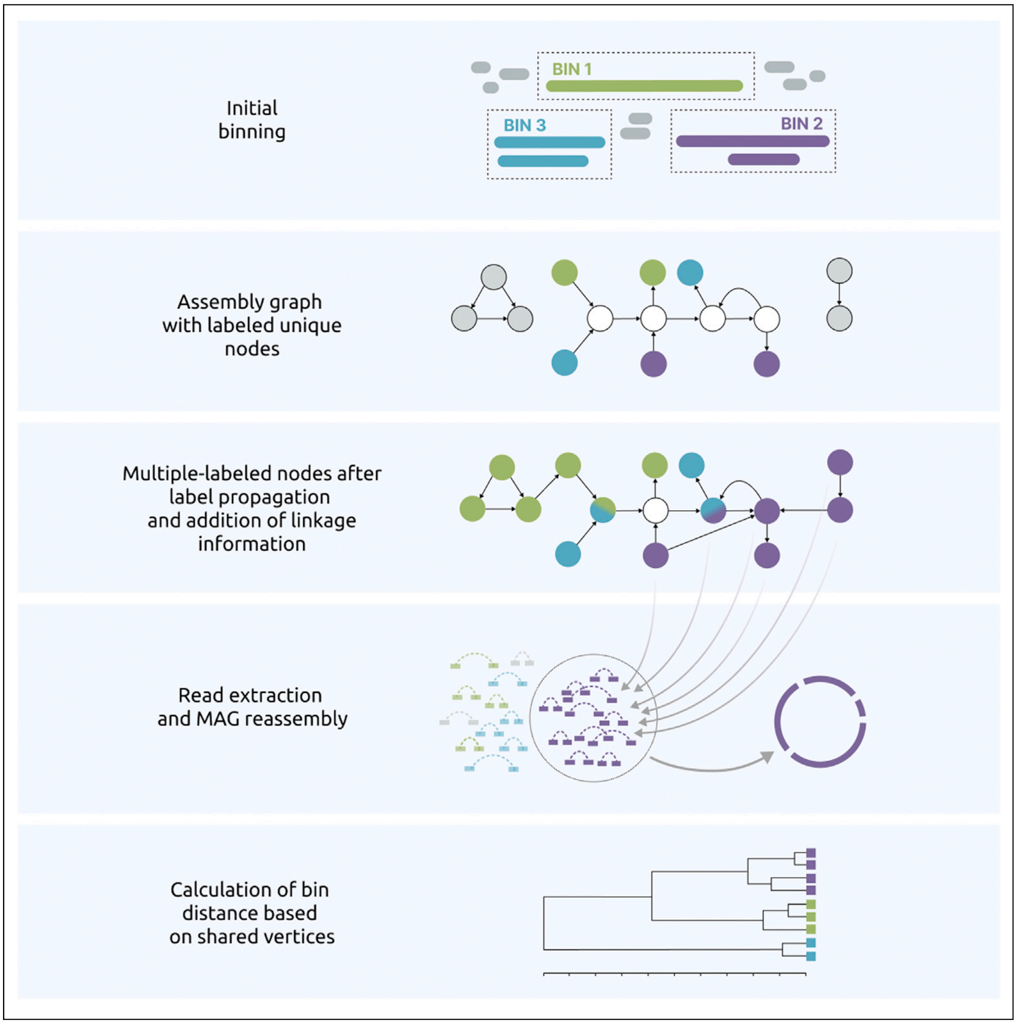
iScience

CellPress
OPEN ACCESS

Article

BinSPreader: Refine binning results for fuller MAG reconstruction


Initial binning
BIN 1
BIN 3
BIN 2
Assembly graph with labeled unique nodes
Multiple-labeled nodes after label propagation and addition of linkage information
Read extraction and MAG reassembly
Calculation of bin distance based on shared vertices

Ivan Tolstoganov, Yuri Kamenev, Roman Kruglikov, Sofia Ochkalova, Anton Korobeynikov

a.korobeynikov@spbu.ru

Highlights

We propose a novel method to refine the binning using the assembly graph connectivity

Method could also use paired-end reads, Hi-C links, and other sources of connectivity

It improves the completeness of the bins without sacrificing the purity

BinSPreader could predict contigs belonging to several MAGs

Tolstoganov et al., iScience *25*, 104770
August 19, 2022 © 2022 The Author(s).
https://doi.org/10.1016/j.isci.2022.104770

iScience

CellPress OPEN ACCESS

Article

BinSPreader: Refine binning results for fuller MAG reconstruction

Ivan Tolstoganov,[1] Yuri Kamenev,[2] Roman Kruglikov,[3] Sofia Ochkalova,[4] and Anton Korobeynikov[1,5,6,*]

SUMMARY

Despite the recent advances in high-throughput sequencing, metagenome analysis of microbial populations still remains a challenge. In particular, the metagenome-assembled genomes (MAGs) are often fragmented due to interspecies repeats, uneven coverage, and varying strain abundance. MAGs are constructed via a binning process that uses features of input data in order to cluster long contigs presumably belonging to the same species.

In this work, we present BinSPreader—a binning refiner tool that exploits the assembly graph topology and other connectivity information to refine binning, correct binning errors, and propagate binning to shorter contigs. We show that BinSPreader could increase the completeness of the bins without sacrificing the purity and could predict contigs belonging to several MAGs.

BinSPreader is effective in binning shorter contigs that often contain important conservative sequences that might be of great interest to researchers.

INTRODUCTION

The amount of microbial organisms that can be easily cultivated is relatively small in proportion to the Earth's total diversity (Rappé and Giovannoni, 2003); therefore, most of the Earth's microbiota proves difficult for analysis. Whole metagenomic shotgun sequencing, which allows for a comprehensive analysis of microbial DNA from a sample, provides an alternative method for understanding the functional potential and genetic composition of different microorganisms that have not been previously cultured. Metagenomic sequencing libraries are then assembled using metagenomic assemblers, such as metaSPAdes (Nurk et al., 2017) or MEGAHIT (Li et al., 2015) for short read libraries or metaFlye (Kolmogorov et al., 2020) for long read libraries.

To extract useful information from complex metagenomic assemblies, a process called *binning* is used. State-of-the-art binners use all different kinds of information including nucleotide content, observed contig abundance, paired-end read connectivity, and other connectivity (e.g. from Hi-C links (DeMaere and Darling, 2019)) to cluster contigs that might belong to the same species. However, this kind of information could only be considered reliable for long contigs, and therefore, the majority of binners discard contigs that are shorter than several kilobases. Given this, the set of contigs could not be considered the ultimate result of a metagenomic assembly. Indeed, the complete information about the assembly is provided via the assembly graph. Usually, the edges of an assembly graph are the maximal nonbranching genomic sequences (unitigs), and the resulting contigs are paths in this assembly graph obtained after the repeat resolution process. The recent development of such assembly graph-aware alignment tools such as SPAligner (Dvorkina et al., 2020), PathRacer (Shlemov and Korobeynikov, 2019), and GraphAligner (Rautiainen and Marschall, 2020) among the others shows that the proper utilization of the assembly graph could significantly improve the obtained results.

To date, it seems that the connectivity information between the contigs in the assembly graph is ignored by the majority of the common binning tools such as MetaBAT2 (Kang et al., 2019), MetaWrap (Uritskiy et al., 2018), and VAMB (Nissen et al., 2021), potentially reducing the overall precision of the results. Recently developed graph-aware binning refining tools such as METAMVGL (Zhang and Zhang, 2021), MetaCoAG (Mallawaarachchi and Lin, 2022), and Binnacle (Muralidharan et al., 2021) also do not utilize the assembly graph in the usual sense of the term. Instead, they are relying on the so-called *scaffold graph* that only preserves the connectivity information between different scaffolds. However, the original assembly graph contains more information including the multiplicity of edges and the set of edges that comprise a contig. To

[1] Center for Algorithmic Biotechnology, Saint Petersburg State University, Saint Petersburg, 199004, Russia

[2] ITMO University, Saint Petersburg 197101, Russia

[3] Lomonosov Moscow State University, Moscow, 119991, Russia

[4] Applied Genomics Laboratory, SCAMT Institute, ITMO University, Saint Petersburg 197101, Russia

[5] Department of Statistical Modelling, Saint Petersburg State University, Saint Petersburg, 198504, Russia

[6] Lead contact

*Correspondence: a.korobeynikov@spbu.ru

https://doi.org/10.1016/j.isci.2022.104770

utilize this greater amount of information, we suggest using the original assembly graph instead of the scaffold graph; this brings to us many opportunities such as multiple binning of individual edges, binning correction, and more precise bin label propagation (from edge to edge and not from scaffold to scaffold).

Standard MAG quality assessment tools such as AMBER (Meyer et al., 2018) and CheckM (Parks et al., 2015) do not assess MAGs for the presence of important sequences, such as mobile genetic elements (MGEs), antibiotic resistance genes (AMR), and CRISPR arrays, that have very high agricultural or clinical importance. As such, MAGs with over 80% completeness as reported by AMBER or CheckM may contain less than 45% of genomic islands and less than 30% of plasmid sequences (Maguire et al., 2020). Mobile genetic elements are commonly flanked by direct repeats (Schmidt and Hensel, 2004) and are therefore located on short repetitive edges of the assembly graph and associated with multiple organisms.

Besides MGEs, MAGs often miss contigs containing rRNA genes. Bacterial genomes contain multiple copies of ribosomal genes forming tangled repeat structures that are often poorly assembled. In a metagenome the situation is further complicated by the presence of conservative parts of rRNA genes shared between different species. Such sequences form intra- and interspecies repeats, and therefore, the overall recovery of decent-length rRNA genes sequences from a metagenome assembly is quite low (Meyer et al., 2022). Finally, the contigs containing rRNA genes have different abundance (due to high copy number) and nucleotide content effectively preventing the majority of binning attempts. Therefore, the inclusion of short edges of the assembly graph into MAGs is crucial for detecting MGE and rRNA sequences.

In this work, we show that assembly graph representation provides more accurate binning of short edges in comparison with scaffold graph representation. We present a new software tool, BinSPreader, which can produce refined MAGs from initial binning by combining metagenomic assembly graph and sequencing data. We show that BinSPreader improves upon state-of-the-art binning refining tools with respect to completeness/purity metrics of MAGs and MGE and rRNA recovery and can accurately predict contigs belonging to multiple bins. BinSPreader is available from cab.spbu.ru/software/binspreader.

RESULTS

Datasets

We used several mock metagenomic datasets, simulated metagenomes as well as real metagenomes for the refining evaluation. These metagenomes are derived from different communities exhibiting different microbial compositions, abundance profiles, genome characteristics, and similarity intended to provide a broader scope of binning data features.

MBARC26 (Singer et al., 2016) is composed of 23 bacterial and 3 archaeal strains isolated from heterogeneous soil, aquatic environments as well as human, bovine, and frog microbiota. For 25 of those strains, reference genomes are known. The genomes of these species span a wide range of genome sizes (1.8–6.5 Mbp), GC-contents (28.4%–72.7%), and repeat contents (0%–18.3%).

BMock12 (Sevim et al., 2019) includes DNA from 12 bacterial strains belonging to actinobacterial, flavobacterial, and proteobacterial taxa that also display a large spread of genome properties. For 11 of those strains, reference genomes are known. Apart from this, it includes three bacteria with genomes of high %GC and high average nucleotide identity (ANI), which complicates the assembly and binning.

ZymoBIOMICS Microbial Community Standard (Nicholls et al., 2019) (referred to as **Zymo**) is a mock community consisting of eight bacterial and two fungal strains. These organisms are lysed in varying degrees and significantly differ in terms of the completeness of sample DNA extraction, which is a determining factor for sequencing and downstream analysis.

The benchmarking dataset from Maguire et al. (2020) (referred to as **magsim-MGE**) contains paired-end Illumina sequencing data of 30 bacteria with randomly assigned relative abundance. It is designed to display a high diversity of genetic features, such as plasmids and genomic islands.

We assembled each of these datasets from Illumina shotgun sequencing data using metaSPAdes 3.15.3 and used reference genomes of included bacteria, archaea, and yeasts to construct ground truth binning standards for benchmark studies.

Table 1. Comparison of running times for BinSPreader and other graph-aware binning refiners in the standard and paired-end utilizing modes on Zymo, BMock12, IC9, and Sharon datasets

Method	Zymo	BMock12	IC9	Sharon
Refining without paired-end connectivity data				
BinSPreader	**0m 15s**	**0m 21s**	**0m 54s**	**0m 29s**
MetaCoAG	19m 15s	4m 22s	14m 29s	3m 3s
Refining with paired-end connectivity data				
BinSPreader-PE	**1h 11m 18s**	**2h 9m 40s**	**39m 25s**	8h 24m 29s
METAMVGL	3h 24m 40s	6h 14m 16s	1h 16m 10s	**4h 57m 48s**
Binnacle (+MetaCarvel)	3h 19m 29s	4h 44m 10s	1h 49m 23s	12h 40m 21s

The execution times for the Binnacle and the MetaCarvel scaffolder are summed because they are only intended to be used together. In addition to the time listed, Binnacle and METAMVGL, unlike BinSPreader, that maps reads on the run, require time for read alignment step. For the evaluation we used bins generated with MetaBAT2 and machine Intel(R) Xeon(R) CPU E7-4880 v2 @ 2.50GHz with five cores.

simHC+ simulated dataset (Wu et al., 2014) was derived out of genome assemblies of 100 bacterial species that mimic high-complexity communities lacking dominant strains. As no original reads for this dataset were available, we used metagenomic assembly, abundance profiles, and ground truth binning standard as provided in MetaCoAG paper (Mallawaarachchi and Lin, 2022).

IC9 is a real clinical gut metagenome of a chronically ill patient collected in a critical care unit. The dataset contains both paired-end and Hi-C data that were crucial for better resolution of MAGs (Ivanova et al., 2022). The metagenome is harboring many antibiotic-resistant strains with elevated levels of horizontal gene transfer. The dataset was assembled as described in Ivanova et al. (2022).

Sharon dataset (Sharon et al., 2012) contains the metagenomic sequencing data of preborn infant fecal samples collected across 18 time points. All these sequencing libraries were co-assembled together using metaSPAdes 3.15.3 before binning and refining.

Evaluated approaches

We benchmarked BinSPreader against state-of-the-art graph-aware binning refiners METAMVGL (Zhang and Zhang, 2021), MetaCoAG (Mallawaarachchi and Lin, 2022) and Binnacle (Muralidharan et al., 2021), as well as consensus-based refiner DAS_TOOL (Sieber et al., 2018). Although all five binning refiners require metagenomic assembly, their requirements for other types of input data differ.

MetaCoAG, Binnacle, and BinSPreader require assembly graph in GFA format as an input. METAMVGL utilizes assembly graphs in obsolete FASTG format, which makes it difficult to use on assembly graphs produced by, e.g. metaFlye. METAMVGL, Binnacle, DAS_TOOL, and BinSPreader require initial binning to refine, whereas MetaCoAG produces initial binning internally using provided coverage profiles. Paired-end read library is required for both METAMVGL and Binnacle as a source of connectivity information between scaffolds and for BinSPreader input paired-end library may be provided optionally to supplement assembly graph links.

Binning refining certainly depends on the quality of the initial binning, as no refining procedure could introduce new bins. In order to reduce the variation of the results that might depend on the initial binning, we used three state-of-the-art binners, MetaBAT2 (Kang et al., 2019), MetaWrap (Uritskiy et al., 2018) (which internally bins using MetaBAT2, CONCOCT, and MaxBin2 (Wu et al., 2014) and produces some sort of consensus binning), and VAMB (Nissen et al., 2021) to produce three initial binnings for METAMVGL and BinSPreader. Because Binnacle is compatible with a limited number of binners, we used it with MetaBAT2 only. Unless stated otherwise, an input metagenomic assembly graph was constructed using metaSPAdes 3.15.3 (Nurk et al., 2017). The comparison of running times of used binners are presented in Table 1.

The resulting binnings of mock and simulated samples were analyzed with AMBER (Meyer et al., 2018). AMBER assessment of bin quality is based on the annotation of metagenomic contigs using the reference

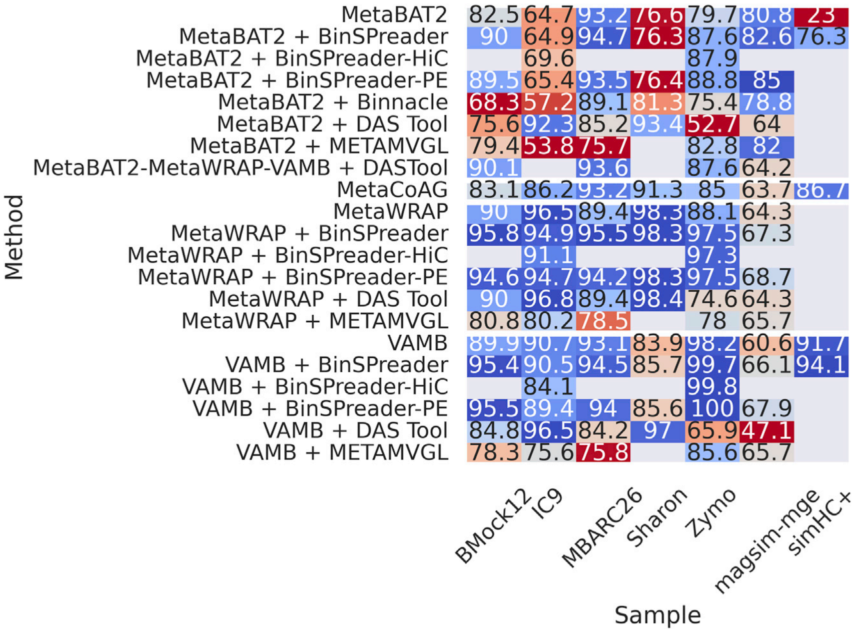


Figure 1. Mean F1 scores across all methods and samples

genomes provided as a "gold standard binning." Contig alignment to reference genomes was performed using metaQUAST (Mikheenko et al., 2015). Evaluations of real metagenomes without references were done via CheckM (Parks et al., 2015). AMR genes were searched using RGI 5.2.1 with CARD database 3.1.4 (McArthur et al., 2013). CRISPRs were detected using MinCED 0.4.2 (Bland et al., 2007). rRNA were annotated with Barrnap 0.9 (Seeman, 2013).

Completeness, contamination, and F1

To benchmark BinSPreader, we analyzed the average (mean) purity, completeness, and F1-score of the binning results calculated by AMBER (at the nucleotide level) for five synthetic datasets. To complement these metrics, we also took into account the number of recovered high-quality genomes with > 90% completeness and < 5% contamination as reported by AMBER. Benchmark metrics on real **IC9** and **Sharon** datasets included mean purity, completeness, and F1-score metrics, which were assessed using CheckM (Parks et al., 2015), as well as total number of bins and the number of high-quality bins with > 90% completeness and < 5% contamination as reported by CheckM.

Mean F1-scores for initial and refined binnings, and the number of recovered high-quality genomes, across all seven datasets are summarized in Figures 1 and 2, respectively. Individual F1-scores for refined bins for **IC9** and **Sharon** can be found in Figures S10 and S11, respectively. Individual F1-scores for refined bins across all datasets can be found in Figures S1–S4.

On **magsim-MGE** dataset, MetaBAT2, VAMB, and MetaWRAP recovered very pure bins with average purity taking values from at least 97.2% for MetaBAT2 to 99.9% for VAMB and MetaWRAP (refer to Table S1 for all AMBER metrics of this dataset). Yet these binnings had very low average completeness with a maximum value of 69.2% for MetaBAT2 and a minimum of 43.5% for VAMB. This poor trade-off between purity and completeness is indicated by the moderate values of the mean F1 score. Best-performing binning tool, MetaBAT2, resulted in an F1 score of 80.8% and recovered 12 high-quality out of 30 total genomes; the worst-performing tool was VAMB with an F1 score of only 60.6% and 8 recovered genomes.

Although refining of initial bins with METAMVGL and BinSPreader led to a minor decrease in average bin purity (no more than 3% for METAMVGL and 1% for BinSPreader across all bins), it significantly reduced the number of unbinned contigs and increased average bin completeness. Bins refined with METAMVGL and BinSPreader had average completeness ranging from 50% for VAMB and MetaWRAP to 72% for MetaBAT2. Refining MetaBAT2 bins using Binnacle did not affect bin purity compared with running MetaBAT2 alone,

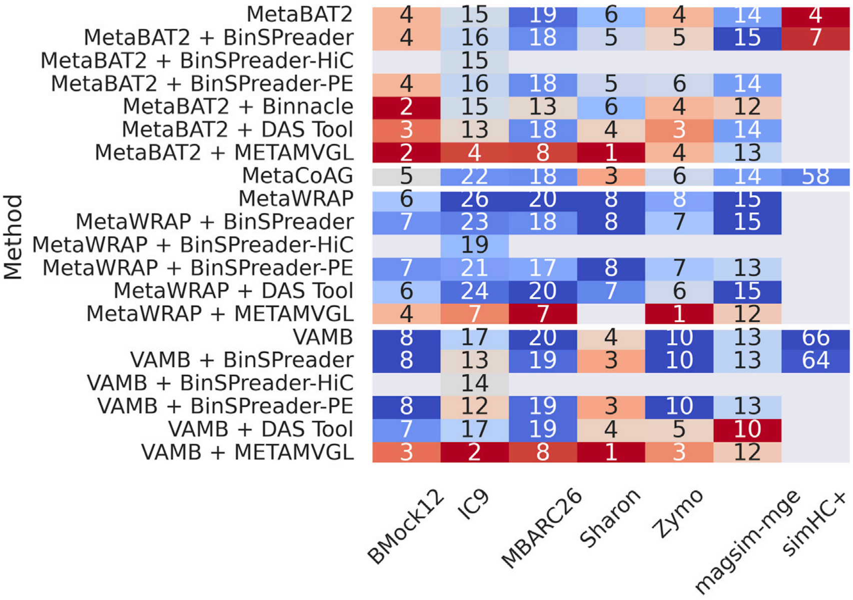


Figure 2. Number of recovered high-quality genomes across all methods and samples

but reduced average completeness. MetaCoAG produced bins with an average purity of 97.5%, average completeness of 47.3%, F1 score of 63.7%, and 10 high-quality MAGs yielding results somewhat worse than several standalone binners.

Of all binning and refining approaches MetaBAT2 bins refined using BinSPreader with paired-end reads showed the best average F1 score of 85.0%, although metaWRAP bins refined using BinSPreader contained more high-quality MAGs (14 for MetaWRAP + BinSPreader vs 12 for MetaBAT2 + BinSPreader).

Available data of **simHC+** dataset allowed benchmarking of the BinSPreader performance against MetaCoAG only (refer to Table S2 for all AMBER metrics), as no original paired-end reads were available in the MetaCoAG paper and therefore one cannot run METAMVGL or Binnacle using only assembly graph and provided abundance profiles. For initial binnings, we used VAMB bins as well as precomputed bins of MaxBin2 and MetaBAT2. The initial bins had the average F1 scores of 23.0%, 84.5%, and 91.7% for MetaBAT2, MaxBin2, and VAMB, respectively. The poor value of the F1 score for MetaBAT2 binning is a result of 13.0% average bin completeness, which is the lowest among all binners. Refining of MetaBAT2 with BinSPreader overall increased bin completeness to 88.4% and F1 score to 76.3% but caused a major drop in average purity of bins. VAMB showed the best balance between precision and sensitivity, although many of the contigs remained unlabeled by VAMB. Refined with BinSPreader VAMB bins showed an increase of the F1 score value to 94.1% and the number of high-quality MAGs increased from 56 to 61. MetaCoAG showed somewhat lower F1 score of 86.7% and captured only 43 high-quality genomes; therefore, BinSPreader + VAMB is the best-performing pair for the **simHC+** dataset.

Binning assessment of **Zymo** mock metagenome showed 100% average purity of MetaBAT2, VAMB, and MetaWRAP bins (refer to Table S3 for more details). Among these, VAMB produced bins with the highest average completeness of 96.5% and the highest value of F1 score of 98.2%. MetaWRAP and MetaBAT2 recovered bins with poorer completeness of 78.8% and 66.2% and moderate F1 scores of 88.1% and 79.7%, respectively. Refining of MetaBAT2 bins with Binnacle decreased the value of average completeness down to 60.6%. Refining with METAMVGL led to a decrease in the purity of bins down to 88.4% for MetaBAT2 and no visible changes in VAMB and MetaWRAP bins. MetaCoAG showed a better trade-off between precision and sensitivity of binning yielding 85.0% F1 score but labeled fewer contigs than BinSPreader. BinSPreader significantly increased bin completeness with negligible effect on purity value that is demonstrated by F1 scores of 87.6% of refined MetaBAT2 bins, 97.5% of MetaWRAP, and 99.7%

of refined VAMB bins. Supplementing BinSPreader with paired-end library allowed the increase of F1 score up to 100% on VAMB bins achieving the best binning result for **Zymo** dataset.

Binning results for the **MBARC26** mock community are described in Table S4. Initial binnings showed balanced precision and sensitivity with an average F1 value of 89.4% for MetaWRAP-produced bins and 93% for VAMB and MetaBAT2. Refined bins produced by METAMVGL had lower quality than the initial binnings of the MetaBAT2, VAMB, and MetaWRAP alone. F1 score of bins recovered with Binnacle and MetaBAT2 dropped from 93.2% down to 89.1%.

MetaCoAG showed better performance with 93.9% average purity, 92.6% average completeness, and an F1 score of 93.2%. F1 scores of BinSPreader refining of MetaBAT2 and VAMB bins were 94.7% and 94.5%, respectively. BinSPreader had a major impact on MetaWRAP binning quality, raising average completeness from 80.0% to 98.9% and decreasing an average purity from 99.8% to 92.3%. This binning approach showed the highest value of F1 score of 95.5% among all tested tools.

Finally, we benchmarked BinSPreader on **BMock12** mock dataset (refer to Table S5 for all AMBER metrics). Bins from initial binning tools had high average purity ranging from 96.5% for MetaWRAP to 98.1% for VAMB and moderate average completeness taking values from 66.9% for MetaBAT2 to 79.3% for MetaWRAP. The F1 scores were in the interval from 79.4% (MetaBAT2) to 87.1% (MetaWRAP). MetaCoAG bins had lower average bin purity of 88.6% and correspondingly lower F1 score of 81.3%. Refining of bins produced with MetaBAT2, VAMB, and MetaWRAP using METAMVGL and refining of MetaBAT2 with Binnacle both led to a considerable decline in all metrics as compared with the original bins. METAMVGL refining of VAMB bins resulted in 9% less average purity and 8% less average completeness compared with the initial VAMB bins. Of all refining tools, only BinSPreader effectively improved the quality of an input binning. Average F1 scores of MetaBAT2, VAMB, and MetaWRAP bins refined using BinSPreader had values of 89.5%, 94.3%, and 94.6%, respectively. MetaWRAP + BinSPreader also retrieved seven high-quality MAGs out of 11 total genomes, more than any other of the tools tested.

Summarizing the results on all datasets, graph-aware refiners, METAMVGL and Binnacle, either yield no noticeable effect (**magsim-MGE**) or impaired the characteristics of the original binning (**MBARC26**, **BMock12**, **Zymo**). MetaCoAG showed a decent ratio of precision to sensitivity but left large portions of contigs unbinned. Exploiting the assembly graph to the fullest extent allowed BinSPreader to augment the bins with unbinned contigs and improve their F1 score with the best trade-off between completeness and contamination. Moreover, it also increased the number of complete MAGs represented with minimal contamination.

We need to outline that the performance of any binning refining tool including BinSPreader depends on the quality of the input bins, as the refiner cannot "invent," e.g. a missed bin. This pitfall is demonstrated on BinSPreader refining of the **simHC+** binning by MetaBAT2. Because of the extremely low completeness of the initial binning, BinSPreader failed to accurately perform contig labeling, causing additional contamination of the bins.

As reported in Tables S7 and S8, MetaWRAP showed the best average F1-score among the initial binners for both **IC9** and **Sharon** datasets (96.5% for IC9 dataset, 98.3% for Sharon). None of the graph-based refiners, namely BinSPreader, METAMVGL, and Binnacle, showed any significant improvement upon initial binnings for both real datasets, with the exceptions of BinSPreader complemented with Hi-C reads for MetaBAT2 on IC9 dataset (64.7% average F1 score for MetaBAT2 against 69.6% average F1 for BinSPreader) and Binnacle-refined MetaBAT2 binning for Sharon dataset (81.3% for Binnacle against 76.6% for MetaBAT2). DAS_TOOL refining demonstrated the best increase in average F1-score for all initial binnings. This, however, could be explained by a consistent decrease in the number of bins after DAS_TOOL refining due to filtering out bins with poor CheckM metrics. As a result, DAS_TOOL recovered less high-quality genomes than BinSPreader (7 instead of 8). Specifically, MetaBAT reported 50 bins for **IC9** dataset, whereas DAS_TOOL reported only 23 refined MetaBAT2 bins.

Negligible increase of CheckM purity and completeness metrics after graph-based refining for real datasets could be explained by limitations in CheckM single-copy gene-based purity and completeness estimation (they are essentially located on long contigs that are likely properly binned and no shorter contigs contribute to these metrics) and by segmentation of metagenomic assembly graphs constructed for these datasets. Indeed, for **Sharon** and **IC9** datasets, the mean number of links outgoing from an assembly graph node (single unitig) are 1.62 and 0.51, respectively, whereas for mock **Zymo** dataset the mean number of outgoing links is 2.71. Also, the bins seem not to cover the whole assembly (30%–60% depending on the binner).

Still, even sparse assembly graphs provide BinSPreader with sufficient information to reconstruct different functional genes more efficiently compared with initial binning as we show below.

Conservative genes recovery

Efficient binning of rRNA still remains one of the greatest challenges in metagenomics, as rRNA gene clusters are hard to assemble due to a high number of intra- and interspecies repeats. Consequently, contigs containing rRNA genes are usually small and belong to multiple genomes. Most of the binners do not support the assignment of one contig to multiple bins making it nearly impossible to recover a sufficiently complete set of rRNA genes for more than one genome, even if rRNA genes were lucky to be assembled completely. We show how BinSPreader's ability to propagate bin labels to small contigs and repeat regions as well as multiple bin assignment could help in rRNA recovery. Beyond that, this approach could also help in genomic islands (GI) recovery that contain regions that are important for clinical applications such as CRISPRs and antimicrobial resistance (AMR) genes.

CRISPRs (Table S9) are not very well assembled in **MBARC26** and **magsim-MGE** datasets, as 18% and 28% of them, respectively, are missing from the assemblies. Nevertheless, BinSPreader shows the best performance recovering all repeat clusters for mock datasets regardless of refining mode. All standalone binners recover nearly equal amounts of CRISPRs, but MetaCoAG manages to greatly surpass them on **MBARC26** (42 recovered CRISPRs against 33 for the best initial binner, MetaWRAP).

However, the most interesting dataset in terms of GI recovery is **magsim-MGE,** as it was specifically designed to showcase this problem (Maguire et al., 2020). Refining with BinSPreader using assembly graph alone does not significantly increase the amount of recovered CRISPRs, but the usage of supplementary paired-end connectivity information gives one of the best results among all binners and BinSPreader runs particularly well (17 recovered CRISPRs out of 23 total assembled versus 13 without paired-end reads). On this dataset, METAMVGL manages to recover the similar number of CRISPRs as BinSPreader.

The results of AMR genes recovery (Tables S10 and S11) are pretty much consistent with CRISPRs recovery. BinSPreader and MetaCoAG still show the best performance, recovering every single assembled AMR gene on mock datasets. In contrast with CRISPRs results, running BinSPreader with paired-end information on **magsim-MGE** dataset yields the best result with MetaBAT2 as initial binner (138 recovered CRISPRs out of 145 assembled), whereas the number of recovered AMR genes after refining with METAMVGL was lower compared with initial MetaBAT2 binning (108 recovered genes after refining vs 115 original AMR genes).

The influence of supplementary connectivity information on the binning refining productivity can be seen on **IC9** dataset, where Hi-C data are available in addition to paired-end reads (Table S11). BinSPreader provided with Hi-C links recovered the maximum amount of AMR genes among all binners and refiners (191 recovered AMR gene out of 300 assembled); this result could be explained by the presence of Hi-C links between chromosomes and plasmids harboring AMR genes, allowing BinSPreader to propagate bin labels to plasmidic contigs more accurately.

Although the amount of recovered GI and functional elements appears to be an informative benchmark for metagenomic studies, the final goal of most research is to get as many high-quality MAGs containing all these elements as possible. In order to make a high-level assessment of MAG recovery, we applied MAG reporting standards developed by the Genomic Standards Consortium (Bowers et al., 2017). MIMAG standard uses different levels of genome completeness and contamination as well as rRNA gene presence. Depending on these metrics MAGs are divided into several groups including a medium-quality draft (≥ 50% completeness, <10% contamination) and a high-quality draft (>90% completeness, <5% contamination, full set of rRNA genes and, at least 18 tRNA). Because rRNA recovery is primarily limited by its assembly completeness, we constructed perfect binning from input assemblies that comprises MAGs with 100% purity and 100% completeness to use it as reference. We also added the second type of high-quality MAGs somewhat lowering the standard: we require a complete set of 16S or 18S rRNAs, as these particular rRNA genes are of most importance for further taxonomic annotation.

Results obtained for **Zymo** and **BMock12** datasets (Figures S12 and S13) emphasize that the assembly quality plays a crucial role in rRNA recovery. Only one high-quality MAG could be obtained from **BMock12** assembly due to the fragmentation of rRNA gene contigs and only two high-quality MAGs (including only 16S rRNA) could be recovered from **Zymo** (Tables S12 and S14) in general. Still, BinSPreader was able to recover these MAGs from VAMB bins with the help of supplementary paired-end connectivity information. Also, BinSPreader refining enriches MetaBAT2-produced bins with medium-quality MAGs (Figure S12) for **Zymo** dataset.

On **MBARC26** and **magsim-MGE** datasets (Figures S14 and S15), we can observe a great improvement in high-quality MAG recovery after the refinement with BinSPreader in multiple binning mode. In comparison with initial bins, BinSPreader refining clearly led to saturation of MAGs with rRNA genes and other small contigs, rather than increasing the number of medium-quality MAGs. The usage of multiple binning approaches increases the number of high-quality MAGs almost down to the assembly level.

Particularly, refining of VAMB binning of **MBARC26** dataset resulted in the recovery of all four possible high-quality MAGs. Different variations of BinSPreader modes yield one high-quality MAG with the full set of rRNA in the worst case, which is still unattainable for the most binners; moreover, all BinSPreader runs increased the number of high-quality MAGs containing only 16S rRNA dramatically, especially when multiple bin assignment mode was used. Even greater improvements could be observed in the refining of binning results obtained on **magsim-MGE** dataset. BinSPreader manages to recover all high-quality MAGs using metaWRAP and VAMB bins without losing any medium-quality MAGs. In addition, BinSPreader recovers 16S rRNA for almost every MAG in VAMB and MetaWRAP-produced bins. Refining MetaBAT2-produced bins using paired-end connectivity information leads to the recovery of five new medium-quality MAGs.

On the real **IC9** metagenome, BinSPreader retrieved all 16S and 23S rRNA genes present in the assembly regardless of initial binning and genome fraction (GF), as shown in Table S16, whereas the second-best refiner-binner combination, bin3C + DAS_TOOL, reconstructed only four 23S rRNA out of six and two 16S rRNA out of three (for rRNA genes assembled at 90% GF). Overall, BinSPreader recovered 71 rRNA genes out of 73 (against 36 for the next best refiner, MetaCoAG). On the **Sharon** dataset, BinSPreader supplemented with paired-end reads retrieved 20 out of 29 of all rRNA genes assembled with at least 50% GF, whereas second-best refiner, MetaCoAG, recovered only six rRNA genes (see Table S17).

Binning refining supplemented with paired-end and Hi-C linkage

To assess the effectiveness of paired-end reads information for binning refining, we used paired-end read libraries available for **Zymo**, **MBARC26**, **Bmock12**, and **magsim-MGE** datasets. We compared MetaBAT2, VAMB, and MetaWRAP bins refined with BinSPreader supplemented with paired-end reads (*BSP-PE mode*) and bins refined with BinSPreader provided with assembly graph only (*BSP mode*). We also assessed Binnacle and METAMVGL refiners that utilize paired-end reads as well. We evaluated binning results using AMBER (Meyer et al., 2018) and reported an F1-score for the initial and refined bins.

For **magsim-MGE** dataset, Table S1 shows that BSP-PE results in higher F1-scores than BSP for all three initial binners. For **Zymo** dataset, Table S3 shows that BSP-PE resulted in higher F1-score per sample than BSP for VAMB and MetaBAT2 binnigs (87.6% for BSP-PE versus 86.7% for BSP for MetaBAT2, 100% for BSP-PE versus 99.8% for BSP for VAMB) and the same F1-scores for MetaWRAP binning. For **BMock12** dataset, BSP resulted in higher F1-score for MetaBAT2 and MetaWRAP datasets than BSP-PE, but BSP-PE for VAMB binning showed the highest F1-score across all binners and refiners (94.6% for BSP-PE versus second highest 94.2% for BSP), as shown in Table S5. For **MBARC26** dataset, BSP-PE resulted in lower F1-scores than BSP for all three initial binners (Table S4). The possible reason for this is contamination in paired-end library for **MBARC26**, as applying METAMVGL and Binnacle to all three initial binnings resulted in lower F1-score (Table S4). For all samples and all initial binners, BSP-PE resulted in higher F1-scores than METAMVGL and Binnacle. F1-scores for separate bins are reported in Figures S1–S4.

The potential of Hi-C technology as a means to cluster metagenomic contigs into bins has been demonstrated on both synthetic and real microbial communities (DeMaere and Darling, 2019; Du and Sun, 2022; Ivanova et al., 2022). We followed two approaches to analyze possible integration of Hi-C technology and binning refining methods for MAG recovery.

First, we obtained initial binning for **Zymo** Hi-C library using dedicated Hi-C bin3C (DeMaere and Darling, 2019) binning tool and refined bin3C binning using BinSPreader (in both BSP and BSP-PE modes). As shown in Table S6, F1-scores reported by AMBER were higher for bin3C bins refined by BinSPreader (0.927 for BSP and BSP-PE against 0.865 for unrefined bin3C bins).

Second, we used **Zymo** Hi-C links as an additional source of information for BinSPreader (*BSP-HiC mode*) and benchmarked the results against BSP-PE and BSP modes for MetaBAT2, MetaWRAP, and VAMB bins. For MetaBAT2 binning, BSP-PE showed the highest F1-score (0.911), followed by BSP-HiC (0.903) and BSP (0.896). For MetaWRAP and VAMB binnings, BSP, BSP-PE, and BSP-HiC resulted in similar F1-scores.

Although BSP-HiC did not show any improvement upon BSP-PE in terms of standard contamination and completeness metrics for **Zymo** dataset, AMR gene detection results for the plasmid-rich **IC9** dataset described earlier (see Conservative genes recovery) show that BSP-HiC can be used to reconstruct additional functional elements located on the unbinned contigs that were not connected to the main genome on the assembly graph.

MAG distance estimation using prob Jaccard index

Sometimes binners produce very pure but incomplete bins (results of Completeness, contamination, and F1 show that this usually applies to MetaBAT2 and MetaWRAP bins). After refining, such bins tend to overlap on an assembly graph, and therefore, the size of such overlap could potentially be used to decide whether one needs to merge certain bins. Also, overlapped labeling of the edges of the assembly graph could measure possible contamination or otherwise shared genome content.

Figure 3 shows the hierarchical clustering of bin distance information calculated from **Zymo** MetaBAT2 bins. One could easily see the bins of different genomes clustered together as well as an overlap of *E. coli* and *S. enterica* bins. Figure 4 shows the hierarchical clustering of bin distance information calculated from **BMock12** MetaBAT2 bins. Again one could see several bins of the same species located together on the graph as well as significant bin overlap between two *Micromonospora* strains as well as contamination of *Marinobacter* bins.

DISCUSSION

Although metagenome-assembled genome binning methods based on TNF distance, coverage profiles, and single-copy marker genes are useful for untangling complex bacterial communities as a whole, they face challenges with the reconstruction of functional elements located in conservative genomic regions, such as rRNAs, CRISPRs, and AMR genes; this is unfortunate, given the phylogenetic and clinical relevance of these functional elements. Conservative genomic regions are usually associated with short repetitive edges of a metagenomic assembly graph. Therefore, there is a clear need for metagenomic binners or refiners that enrich MAGs with short and possible repetitive contigs.

BinSPreader is a binning refining tool that effectively utilizes assembly graph connectivity information and predicts contigs belonging to several MAGs. We show that existing binning refining tools, which utilize scaffold graphs instead of assembly graphs, are less effective than BinSPreader in terms of functional element recovery (Tables S9–S11) and in terms of rRNA genes recovery for artificial (Tables S12–S15) and real (Tables S16 and S17) metagenomes. Although BinSPreader does not show significant increase in 16S/18S rRNA genes reconstruction compared with initial binning for **BMock12** and **Zymo** datasets, we show that for these datasets ability for rRNA recovery is limited mostly by assembly quality (Tables S12 and S14). Experimental results on synthetic and simulated datasets show that BinSPreader also outperforms existing refiners in terms of standard contamination and completeness metrics (Figures S1–S4).

In addition to MAG recovery, BinSPreader provides two additional features: first, the read splitting feature, which takes into account possible overlap between MAGs and thus enables fuller MAG reconstruction after reassembly. We also introduced a bin distance measure that provides an overlap-based estimation of evolutionary distance between MAGs, thus potentially providing a novel source of information for taxonomic classification as well as detecting possible bin contamination.

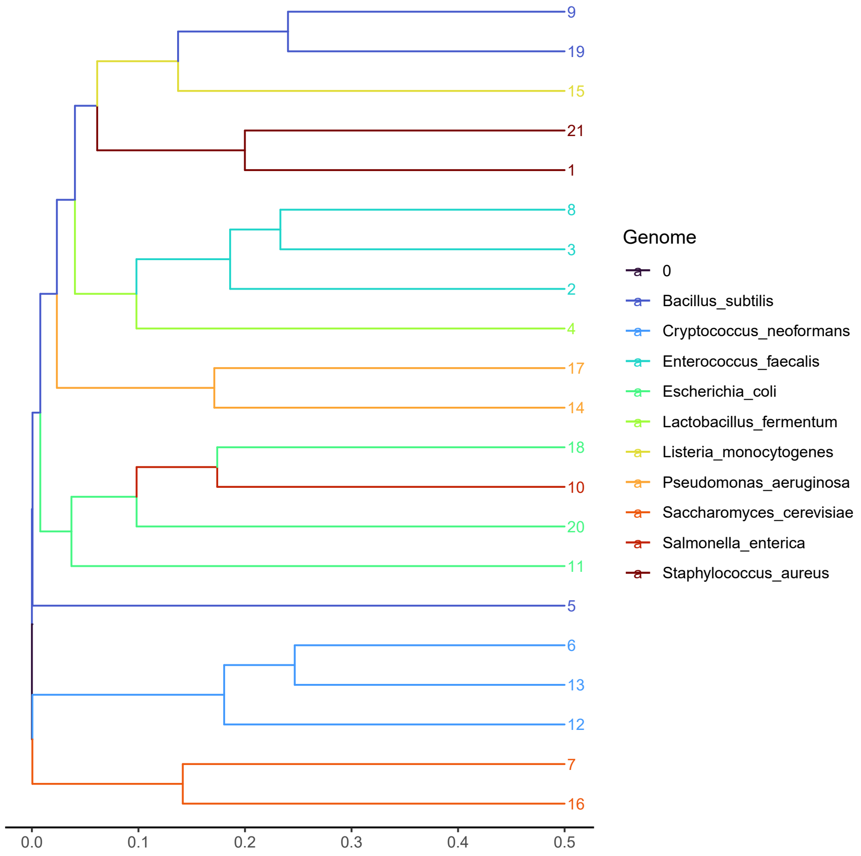


Figure 3. Hierarchical clustering of Zymo MetaBAT2 refined bins using the prob Jaccard distance between bin distributions on the assembly graph

The leafs are colored by reference, and leaf numbers are bin labels. *E. coli* and *S. enterica* bins have overlap on the assembly graph and therefore are cross-contaminated.

Limitations of the study

BinSPreader heavily relies on the quality of the input binning. In particular, it cannot clean the contaminated bins occurred when several MAGs are joined together by a binner. The second input to BinSPreader is an assembly graph where the graph connectivity is in the heart of BinSPreader algorithm. If the assembly graph is disconnected or otherwise fragmented, then BinSPreader naturally cannot propagate the binning in the absence of additional connectivity information (e.g. from scaffolds, paired-end links or HiC data).

STAR★METHODS

Detailed methods are provided in the online version of this paper and include the following:

- KEY RESOURCES TABLE
- RESOURCE AVAILABILITY
  - Lead contact
  - Materials availability
  - Data and code availability
- METHOD DETAILS
  - From scaffold binning to edge binning
  - Link graph
  - Binning refinement
  - Choosing regularization parameters
  - Sparse binning & propagation
  - Binning strategies: From edges back to scaffolds

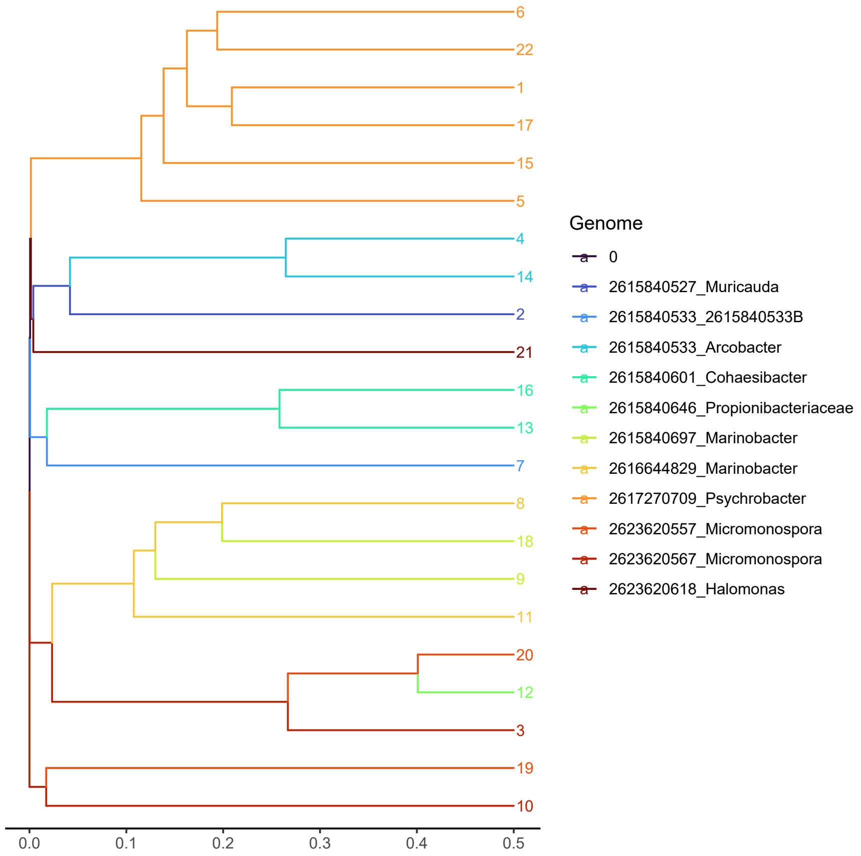


Figure 4. Hierarchical clustering of BMock12 MetaBAT2 refined bins using the prob Jaccard distance between bin distributions on the assembly graph

The leafs are colored by reference, and leaf numbers are bin labels. Two *Micromonospora* strains have significant overlap on the assembly graph, and one of the *Marinobacter* bins is clearly contaminated.

- ○ Measuring MAG distance using prob Jaccard index
- ○ Read extraction and MAG reassembly

SUPPLEMENTAL INFORMATION

Supplemental information can be found online at https://doi.org/10.1016/j.isci.2022.104770.

ACKNOWLEDGMENTS

The research was carried out in part by computational resources provided by the Resource Center "Computer Center of SPbU." The authors are grateful to Saint Petersburg State University for the overall support of this work. IT, SO, and AK were supported by the Russian Science Foundation (grant 19-14-00172).

AUTHOR CONTRIBUTIONS

AK and IT developed the BinSPreader concept. AK, YK, and IT implemented and maintained BinSPreader. SO and RK benchmarked BinSPreader and analyzed results. SO analyzed the datasets with respect to the standard performance metrics. RK performed analysis of conservative sequences recovery. All authors read and contributed to the final manuscript.

DECLARATION OF INTERESTS

The authors declare no competing interests.

Received: February 12, 2022
Revised: June 20, 2022
Accepted: July 12, 2022
Published: August 19, 2022

REFERENCES

Bland, C., Ramsey, T.L., Sabree, F., Lowe, M., Brown, K., Kyrpides, N.C., and Hugenholtz, P. (2007). CRISPR recognition tool (CRT): a tool for automatic detection of clustered regularly interspaced palindromic repeats. BMC Bioinf. *8*, 209. https://doi.org/10.1186/1471-2105-8-209.

Bowers, R.M., Kyrpides, N.C., Stepanauskas, R., Harmon-Smith, M., Doud, D., Reddy, T.B.K., Schulz, F., Jarett, J., Rivers, A.R., Eloe-Fadrosh, E.A., et al. (2017). Minimum information about a single amplified genome (MISAG) and a metagenome-assembled genome (MIMAG) of bacteria and archaea. Nat. Biotechnol. *35*, 725–731. https://doi.org/10.1038/nbt.3893.

Cheng, H., Concepcion, G.T., Feng, X., Zhang, H., and Li, H. (2021). Haplotype-resolved de novo assembly using phased assembly graphs with hifiasm. Nat. Methods *18*, 170–175. https://doi.org/10.1038/s41592-020-01056-5.

Chung, F.R.K. (1997). Spectral Graph Theory (American Mathematical Society).

DeMaere, M.Z., and Darling, A.E. (2019). bin3c: exploiting hi-c sequencing data to accurately resolve metagenome-assembled genomes. Genome Biol. *20*, 46. https://doi.org/10.1186/s13059-019-1643-1.

Du, Y., and Sun, F. (2022). HiCBin: binning metagenomic contigs and recovering metagenome-assembled genomes using hi-c contact maps. Genome Biol. *23*, 63. https://doi.org/10.1186/s13059-022-02626-w.

Dvorkina, T., Antipov, D., Korobeynikov, A., and Nurk, S. (2020). SPAligner: alignment of long diverged molecular sequences to assembly graphs. BMC Bioinf. *21*, 306. https://doi.org/10.1186/s12859-020-03590-7.

Ivanova, V., Chernevskaya, E., Vasiluev, P., Ivanov, A., Tolstoganov, I., Shafranskaya, D., Ulyantsev, V., Korobeynikov, A., Razin, S.V., Beloborodova, N., et al. (2022). Hi-c metagenomics in the ICU: Exploring clinically relevant features of gut microbiome in chronically critically ill patients. Front. Microbiol. *12*, 770323. https://doi.org/10.3389/fmicb.2021.770323.

Jaccard, P. (1912). The distribution of the flora in the alpine zone. New Phytol. *11*, 37–50. https://doi.org/10.1111/j.1469-8137.1912.tb05611.x.

Kang, D.D., Li, F., Kirton, E., Thomas, A., Egan, R., An, H., and Wang, Z. (2019). MetaBAT 2: an adaptive binning algorithm for robust and efficient genome reconstruction from metagenome assemblies. PeerJ *7*, e7359. https://doi.org/10.7717/peerj.7359.

Kolmogorov, M., Bickhart, D.M., Behsaz, B., Gurevich, A., Rayko, M., Shin, S.B., Kuhn, K., Yuan, J., Polevikov, E., Smith, T.P.L., and Pevzner, P.A. (2020). metaFlye: scalable long-read metagenome assembly using repeat graphs. Nat. Methods *17*, 1103–1110. https://doi.org/10.1038/s41592-020-00971-x.

Li, D., Liu, C.-M., Luo, R., Sadakane, K., and Lam, T.-W. (2015). MEGAHIT: an ultra-fast single-node solution for large and complex metagenomics assembly via succinct de Bruijn graph. Bioinformatics *31*, 1674–1676. https://doi.org/10.1093/bioinformatics/btv033.

Lieberman-Aiden, E., van Berkum, N.L., Williams, L., Imakaev, M., Ragoczy, T., Telling, A., Amit, I., Lajoie, B.R., Sabo, P.J., Dorschner, M.O., et al. (2009). Comprehensive mapping of long-range interactions reveals folding principles of the human genome. Science *326*, 289–293. https://doi.org/10.1126/science.1181369.

Maguire, F., Jia, B., Gray, K.L., Lau, W.Y.V., Beiko, R.G., and Brinkman, F.S.L. (2020). Metagenome-assembled genome binning methods with short reads disproportionately fail for plasmids and genomic islands. Microb. Genom. *6*. https://doi.org/10.1099/mgen.0.000436.

Mallawaarachchi, V., and Lin, Y. (2022). Metacoag: binning metagenomic contigs via composition, coverage and assembly graphs. In Research in Computational Molecular Biology, I. Pe'er, ed. (Springer International Publishing), pp. 70–85.

McArthur, A.G., Waglechner, N., Nizam, F., Yan, A., Azad, M.A., Baylay, A.J., Bhullar, K., Canova, M.J., De Pascale, G., Ejim, L., et al. (2013). The comprehensive antibiotic resistance database. Antimicrob. Agents Chemother. *57*, 3348–3357. https://doi.org/10.1128/aac.00419-13.

Meyer, F., Fritz, A., Deng, Z.-L., Koslicki, D., Lesker, T.R., Gurevich, A., Robertson, G., Alser, M., Antipov, D., Beghini, F., et al. (2022). Critical assessment of metagenome interpretation: the second round of challenges. Nat. Methods *19*, 429–440. https://doi.org/10.1038/s41592-022-01431-4.

Meyer, F., Hofmann, P., Belmann, P., Garrido-Oter, R., Fritz, A., Sczyrba, A., and McHardy, A.C. (2018). AMBER: assessment of metagenome BinnERs. GigaScience *7*. https://doi.org/10.1093/gigascience/giy069.

Mikheenko, A., Saveliev, V., and Gurevich, A. (2015). MetaQUAST: evaluation of metagenome assemblies. Bioinformatics *32*, 1088–1090. https://doi.org/10.1093/bioinformatics/btv697.

Moulton, R., and Jiang, Y. (2018). Maximally consistent sampling and the jaccard index of probability distributions. In 2018 IEEE International Conference on Data Mining (ICDM) (IEEE), pp. 347–356.

Muralidharan, H.S., Shah, N., Meisel, J.S., and Pop, M. (2021). Binnacle: using scaffolds to improve the contiguity and quality of metagenomic bins. Front. Microbiol. *12*, 638561. https://doi.org/10.3389/fmicb.2021.638561.

Nicholls, S.M., Quick, J.C., Tang, S., and Loman, N.J. (2019). Ultra-deep, long-read nanopore sequencing of mock microbial community standards. GigaScience *8*, giz043. https://doi.org/10.1093/gigascience/giz043.

Nie, F., Wang, X., Jordan, M.I., and Huang, H. (2016). The constrained laplacian rank algorithm for graph-based clustering. In Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence. AAAI'16 (AAAI Press), pp. 1969–1976.

Nie, F., Xiang, S., Liu, Y., and Zhang, C. (2009). A general graph-based semi-supervised learning with novel class discovery. Neural Comput. Appl. *19*, 549–555. https://doi.org/10.1007/s00521-009-0305-8.

Nie, F., Xu, D., Tsang, I.W.-H., and Zhang, C. (2010). Flexible manifold embedding: a framework for semi-supervised and unsupervised dimension reduction. IEEE Trans. Image Process. *19*, 1921–1932. https://doi.org/10.1109/tip.2010.2044958.

Nissen, J.N., Johansen, J., Allesøe, R.L., Sønderby, C.K., Armenteros, J.J.A., Grønbech, C.H., Jensen, L.J., Nielsen, H.B., Petersen, T.N., Winther, O., and Rasmussen, S. (2021). Improved metagenome binning and assembly using deep variational autoencoders. Nat. Biotechnol. *39*, 555–560. https://doi.org/10.1038/s41587-020-00777-4.

Nurk, S., Meleshko, D., Korobeynikov, A., and Pevzner, P.A. (2017). metaspades: a new versatile metagenomic assembler. Genome Res. *27*, 824–834. https://doi.org/10.1101/gr.213959.116.

Parks, D.H., Imelfort, M., Skennerton, C.T., Hugenholtz, P., and Tyson, G.W. (2015). CheckM: assessing the quality of microbial genomes recovered from isolates, single cells, and metagenomes. Genome Res. *25*, 1043–1055. https://doi.org/10.1101/gr.186072.114.

Rappé, M.S., and Giovannoni, S.J. (2003). The uncultured microbial majority. Annu. Rev. Microbiol. *57*, 369–394. https://doi.org/10.1146/annurev.micro.57.030502.090759.

Rautiainen, M., and Marschall, T. (2020). GraphAligner: rapid and versatile sequence-to-graph alignment. Genome Biol. *21*, 253. https://doi.org/10.1186/s13059-020-02157-2.

Schmidt, H., and Hensel, M. (2004). Pathogenicity islands in BacterialPathogenesis. Clin. Microbiol. Rev. *17*, 14–56. https://doi.org/10.1128/cmr.17.1.14-56.2004.

Seeman, T. (2013). barrnap 0.9: rapid ribosomal rna prediction. https://github.com/tseemann/barrnap.

Sevim, V., Lee, J., Egan, R., Clum, A., Hundley, H., Lee, J., Everroad, R.C., Detweiler, A.M., Bebout, B.M., Pett-Ridge, J., et al. (2019). Shotgun metagenome data of a defined mock community using oxford nanopore, PacBio and illumina technologies. Sci. Data *6*, 285. https://doi.org/10.1038/s41597-019-0287-z.

Sharon, I., Morowitz, M.J., Thomas, B.C., Costello, E.K., Relman, D.A., and Banfield, J.F. (2012). Time series community genomics analysis reveals rapid shifts in bacterial species, strains, and phage during infant gut colonization. Genome Res. *23*, 111–120. https://doi.org/10.1101/gr.142315.112.

Shlemov, A., and Korobeynikov, A. (2019). PathRacer: Racing profile HMM paths on assembly graph. In Algorithms for Computational Biology (Springer International Publishing), pp. 80–94.

Sieber, C.M.K., Probst, A.J., Sharrar, A., Thomas, B.C., Hess, M., Tringe, S.G., and Banfield, J.F. (2018). Recovery of genomes from metagenomes via a dereplication, aggregation and scoring strategy. Nat. Microbiol. *3*, 836–843. https://doi.org/10.1038/s41564-018-0171-1.

Singer, E., Andreopoulos, B., Bowers, R.M., Lee, J., Deshpande, S., Chiniquy, J., Ciobanu, D., Klenk, H.-P., Zane, M., Daum, C., et al. (2016). Next generation sequencing data of a defined microbial mock community. Sci. Data *3*, 160081. https://doi.org/10.1038/sdata.2016.81.

Uritskiy, G.V., DiRuggiero, J., and Taylor, J. (2018). MetaWRAP—a flexible pipeline for genome-resolved metagenomic data analysis. Microbiome *6*, 158. https://doi.org/10.1186/s40168-018-0541-1.

Wu, Y.-W., Tang, Y.-H., Tringe, S.G., Simmons, B.A., and Singer, S.W. (2014). MaxBin: an automated binning method to recover individual genomes from metagenomes using an expectation-maximization algorithm. Microbiome *2*, 26. https://doi.org/10.1186/2049-2618-2-26.

Zhang, Z., and Zhang, L. (2021). METAMVGL: a multi-view graph-based metagenomic contig binning algorithm by integrating assembly and paired-end graphs. BMC Bioinf. *22*, 378. https://doi.org/10.1186/s12859-021-04284-4.

STAR★METHODS

KEY RESOURCES TABLE

REAGENT or RESOURCE	SOURCE	IDENTIFIER
Deposited data		
MBARC26 dataset	Singer et al. (2016)	NCBI SRA, accession number SRX1836716
BMock12 dataset	Sevim et al. (2019)	NCBI SRA, accession number SRX4901583
ZymoBIOMICS Microbial Community Standard	Nicholls et al. (2019)	https://github.com/LomanLab/mockcommunity
magsim-MGE dataset	Maguire et al. (2020)	https://osf.io/x2y8f/
simHC+ dataset	Wu et al. (2014), Mallawaarachchi and Lin (2022)	https://figshare.com/projects/MetaCoAG/121014
IC9 dataset	Ivanova et al. (2022)	NCBI SRA, accession number SRX10650162 for Illumina data, accession number SRX10650163 for Hi-C data
Sharon dataset	Sharon et al. (2012)	NCBI SRA, accession number SRX144807
Assembly graphs, scaffolds, abundance profiles, binning results for the datasets used in the study	This Study	https://figshare.com/projects/BinSPreader/132425
CARD database	McArthur et al. (2013)	http://arpcard.mcmaster.ca/
Software and algorithms		
AMBER v.2.0.3	Meyer et al. (2018)	https://github.com/CAMI-challenge/AMBER
Barrnap v.0.9	Torsten Seemann	https://github.com/tseemann/barrnap
bin3C v.0.1.1	DeMaere and Darling (2019)	https://github.com/cerebis/bin3C
Binnacle (January 16th version)	Muralidharan et al. (2021)	https://github.com/marbl/binnacle
BinSPreader v.0.1	This Study	https://github.com/ablab/spades/releases/tag/binspreader-recombseq
CheckM v.1.0.13	Parks et al. (2015)	https://github.com/Ecogenomics/CheckM
DAS_Tool v.1.1.3	Sieber et al. (2018)	https://github.com/cmks/DAS_Tool
MaxBin2 v.2.2.7	Wu et al. (2014)	https://sourceforge.net/projects/maxbin2/
MetaBAT2 v.2.12.1	Kang et al. (2019)	https://bitbucket.org/berkeleylab/metabat/src/master/
MetaCoAG v.1.0	Mallawaarachchi and Lin (2022)	https://github.com/metagentools/MetaCoAG
METAMVGL v.1.0	Zhang and Zhang (2021)	https://github.com/ZhangZhenmiao/METAMVGL
metaQUAST v.5.0.2	Mikheenko et al. (2015)	https://cab.spbu.ru/software/metaquast/
metaWRAP v.1.3	Uritskiy et al. (2018)	https://github.com/bxlab/metaWRAP
MinCED v.0.4.2	Bland et al. (2007)	https://github.com/ctSkennerton/minced
RGI v.5.2.1	McArthur et al. (2013)	https://github.com/arpcard/rgi
SPAdes v.3.15.3	Nurk et al. (2017)	https://cab.spbu.ru/software/meta-spades/
VAMB v.3.0.3	Nissen et al. (2021)	https://github.com/RasmussenLab/vamb

RESOURCE AVAILABILITY

Lead contact

Further information and requests for resources should be directed to and will be fulfilled by the Lead Contact, Anton Korobeynikov (a.korobeynikov@spbu.ru).

Materials availability

This study did not generate new unique reagents.

Data and code availability

- The paper analyzes existing, currently available data. The accession URLs for the datasets are listed in the Key resources table.
- BinSPreader is publicly available online from cab.spbu.ru/software/binspreader.
- Any additional information required to reanalyze the data reported in this paper is available from the lead contact upon request.

METHOD DETAILS

From scaffold binning to edge binning

Most binners output their results in a form of *scaffold binning*, i.e., a map B from a set of scaffolds P to a set of bins C. This representation is not entirely accurate, since long scaffolds in a metagenomic assembly may contain repetitive regions, which can belong to multiple species in a sample, and therefore in multiple bins. To alleviate this, BinSPreader transforms the initial scaffold binning to the *edge binning* using an assembly graph. Let G be an assembly graph in GFA format consisting of a set of edges $E(G)$, links $L(G)$ between them, and scaffolds $P(G)$ with their corresponding paths in the assembly graph. Given edge $e_i \in E(G)$, let $P(e_i) \subset P(G)$ be the set of scaffolds that contain e_i, and $C(e_i) \subset C$ be the set of bin labels of $P(e_i)$. For assembly graph G and scaffold binning B, BinSPreader transforms scaffold binning B to edge binning matrix Y, where

$$Y_{ij} = \begin{cases} \frac{1}{|C(e_i)|}, & \text{if bin } c_j \in C(e_i) \\ 0, & \text{otherwise.} \end{cases} \qquad \text{(Equation 1)}$$

Here each row Y_i represents a *soft binning* of edge e_i, which can be interpreted as the containment probability distribution over the set of bins. Edge binning represents a more fine-grained representation of initial binning than scaffold binning, as repetitive edges may contain multiple bins if they are traversed by several paths.

Link graph

While edges of the assembly graph G are used to store the initial binning and the end results, vertices of the assembly graph provide minimal required connectivity information for BinSPreader. Connectivity information is stored in a form of a weighted *link graph* H, where $V(H) = E(G)$, $E(H) = V(G)$ and the edge weight L_{ij} represents the weight of a link between assembly graph edges e_i and e_j. The higher L_{ij} is, the more likely is that e_i and e_j belong to the same bin. Initially BinSPreader uses adjacency matrix of an assembly graph G for weights with $L_{ij} = 1$ if the edges e_i and e_j are adjacent in G and zero otherwise.

Besides the adjacency weights, BinSPreader also by default considers the set of *scaffold links*: if two edges are joined in a scaffold, but not adjacent in the graph we add the link in H (add edge and set $L_{ij} = 1$) between them. Usually, such scaffold joins are made by an assembler to jump over coverage gaps or long unresolved repeats. In both cases adding these links increases the contiguity of the link graph and could help the binning propagation across assembly gaps.

In addition to the assembly graph itself, BinSPreader is able to construct links from paired-end and Hi-C (Lieberman-Aiden et al., 2009) libraries which can be provided optionally. Reads from paired-end libraries and Hi-C libraries are aligned using k-mer alignment similar to (Cheng et al., 2021). First, we index unique k-mers in the assembly graph. Then we align a Hi-C read pair if it contains two or more non-overlapping k-mers. We use $k = 31$ by default as most 31-mers in the metagenomic assembly graph are unique, but that value can be adjusted depending on the size of the sample. We then increase the link weight L_{ij} by the logarithm of the total number of read-pairs aligned to e_i and e_j from all input libraries.

Binning refinement

Informally speaking, we say that an edge binning is *smooth* if soft bins associated with a pair of edges joined by a link with high weight are similar. As such, binning refining problem can be defined as finding smooth edge binning F which is close in some sense to the initial edge binning Y. Given link graph H, we use a quadratic form of normalized Laplacian of H as a standard spectral graph theory measure of smoothness (Chung, 1997; Nie et al., 2010, 2016). Let D be a degree matrix of H, and L be an adjacency matrix of H. Then we define edge binning smoothness as

$$S(H,F) = \operatorname{tr}\left(F^T D^{-1/2}(D - L)D^{-1/2}F\right).$$

We define binning refinement problem as

$$S(H,F) + \sum_{i=1}^{n} \mu_i \;\; F_i - Y_i^2 \to \min_F, \qquad \text{(Equation 2)}$$

where the second term penalizes the distance between resulting binning F and original binning Y according to regularization parameters defined separately for every edge.

We use iterative algorithm for optimizing cost function (2), which is similar to one from Nie et al. (2009). Let $\tilde{L}$ be the normalized weight matrix $D^{-1/2}(D - L)D^{-1/2}$, where D is a degree matrix of H. Then let $P = I_\alpha \tilde{D}^{-1}\tilde{L}$, where $\tilde{D}$ is a diagonal of $\tilde{L}$, I is an identity matrix of size $|V(H)| \times |V(H)|$, and I_α is a diagonal matrix being $I_{ii} = 1/\mu_i$. Initially, we set $F(0) = Y$. At each iteration, for every assembly edge e_i the soft labels from neighboring links (e_i, e_j) with weight H_{ij} are added to the soft label of e_i with coefficient H_{ij}. At iteration $k+1$ we set

$$F(k+1) = PF(k) + (I - I_\alpha)Y \qquad \text{(Equation 3)}$$

As shown in Nie et al. (2009), the obtained sequence $F(k)$ will eventually converge to solution $\tilde{F}$, which is produced as the resulting edge binning.

We need to explicitly note that while all the matrices involved are quite large, they are extremely sparse and there is no need to store and calculate them explicitly. The soft binning for each edge at iteration k (the rows of $F(k)$) depends only on soft binnings of adjacent edges (which in ordinary de Bruijn graph case is not more than 8) as well as normalized link weights. This enables computational and memory-efficient way to perform the iterations by using sparse and succinct data structures.

Choosing regularization parameters

The choice of per-edge regularization parameters $\alpha_i = 1/\mu_i$ is different for different modes of BinSPreader. Firstly, we always set $\alpha_i = 1$ for all repetitive edges (i.e. the edge that belongs to multiple scaffolds). As it could be easily seen from Equation (3), the original binning for such edges will be ignored and soft binning for such edge is determined entirely via binning propagation. However, the binning from binned repetitive edges will be propagated down to their neighbors. This ensures proper and fair binning in case of e.g. partially unresolved repeats.

Setting $\alpha_i = 0$ for edge e_i would force use of original binning. This is done for all non-repetitive binned edges in *propagation mode* of BinSPreader. In this case, the original binning is essentially preserved and only propagated further on to unbinned edges.

Setting $0 < \alpha_i < 1$ for edge e_i allows one to balance between preserving the initial binning and propagating the binning from adjacent edges. In *correction mode* of BinSPreader α_i is set to 0.6 by default for all binned edges longer than 1000 bp, for shorter edges the value of α_i is gradually increasing up to $\alpha_i = 1$ for edges of length 1. The motivation for this is as follows: while short edges might be unique and belong only to the single scaffold, they are likely repetitive and belong to unresolved repeats. The shorter the edge is, the higher its likelihood of being repetitive and we equally treat all edges longer than 1000 bp. Certainly, the latter still might be repetitive and this is what the default value of $\alpha_i = 0.6$ tries to accommodate.

Sparse binning & propagation

Binnings of real metagenomic datasets are typically sparse, since large datasets contain strains with high enough coverage to contribute to metagenomic assembly, but not high enough to be binned using the abundance and nucleotide profiles.

BinSPreader uses a special working mode of the binning refining algorithm for *sparse* binnings, where the total length of initially binned contigs is significantly lower than the total assembly length. Below we show why the standard mode of BinSPreader produces highly contaminated bins when refining sparse binnings and describe the *sparse mode* of BinSPreader designed to alleviate that problem.

Given assembly graph G with the set of regularization parameters α_i, and initial edge binning Y, we say that edge e_i is *refinable*, if $\alpha_i \neq 0$. If an initially unlabeled edge e is connected to an initially labeled edge by a path of refinable edges, it eventually will be labeled after applying binning refinement algorithm to graph G and binning Y. Therefore, in the standard correction mode of BinSPreader with $\alpha_i > 0$ every unlabeled edge residing in the same connected component with labeled edges will become labeled after the refining. As such, refining of initially sparse (incomplete) binnings that cover only a small part of G with n bins via the standard correction mode of BinSPreader will result in assigning the majority of contigs in the refined binning to one or several of these same n initial bins potentially inflating and contaminating them.

To reduce the number of refinable edges while still allowing binning propagation, we adjust regularization parameters α_i for initially unlabeled edges with *distance coefficients* β_i, reflecting assembly graph distance to the closest initially labeled edge. Given assembly graph G and initial binning Y, let $Dist(e, Y)$ be the length of the shortest path in assembly graph G from edge e to the closest edge which is labeled in Y. We say that edge e is *distant*, if $Dist(e, Y) > D$, where D is distance threshold with default value 10,000. To ensure that distance coefficients β_i change smoothly from 1 for labeled edges to 0 for distant edges we utilize the same binning refining algorithm.

We introduce two bins, one for all labeled edges in G and another one for all distant edges. Then we run the binning refining algorithm as in the standard correction mode of BinSPreader and set β_i to the obtained weight of the first (''labeled'') bin. This makes the values of β_i gradually decrease from being 1 in the case of initially binned edge e_i down to 0 when moving out of binning edges on the graph.

For sparse propagation the regularization parameters are then set as $\alpha_i{}' = \alpha_i\beta_i$, where α_i are regularization parameter values for the standard correction mode of BinSPreader. This allows us to keep the initial binning intact for the edges located ''far away'' from the binned ones.

In addition to adjusted regularization parameters, the sparse mode of BinSPreader also adds a dedicated bin for initially unbinned edges. However, while we allow the binning to propagate from binned edges down to unbinned ones we need to prevent the propagation of this special ''unbinned'' label. In order to do so, we modify the iteration procedure in sparse mode adjusting the weight matrix P accordingly.

Binning strategies: From edges back to scaffolds

After inferring refined edge binning $\tilde{F}$, BinSPreader uses it to produce the scaffold binning F'. BinSPreader can output results either in single assignment or multiple assignment mode, and utilizes either *majority length* or *maximum likelihood* strategy (default).

Given a scaffold s containing edges $e_1,\ldots,e_m$, and bin c_j the binning strategy defines a score function $Score(s,c_j)$. For majority length strategy we define $c(e_i) = \arg\max_j \tilde{F}_{ij}$ and use $Score(s, c_j) = \sum_{e_i : c(e_i) = j} length(e_i)$. For maximum likelihood strategy $Score(s, c_j) = \sum_{e_i \in s} length(e_i) \times \tilde{F}_{ij}$.

In single assignment mode BinSPreader outputs a single bin label arg $\max Score(s, c_j)$ for every scaffold s. In a multiple assignment mode, BinSPreader outputs a set of labels $\{c_j\}^{C_i}$with maximal $Score'$s, which cumulatively explain at least 95% of the total $Score$. Note that raw $Score(s, c_j)$ values are reported by BinSPreader as well, so one could use them for their own binning assignment procedures.

Measuring MAG distance using prob Jaccard index

The typical measure to estimate the overlap of two sets is Jaccard index (Jaccard, 1912). However, in the case of BinSPreader the sets (bins) are fuzzy as the result of binning refining is a set of weights that represent the bin labeling probability distribution. Let $\tilde{F}$ be a refined multiple edge binning. In order to estimate a possible overlap of bins on the assembly graph from the soft binning, we assign an edge probability distribution $\{p_i^{(j)}\}$ to every bin c_j by normalizing its edge weight vector $\tilde{F}_{*,j}$. We than calculate the prob-Jaccard index J_p from (Moulton and Jiang, 2018) among all pairs of bins. Given two bins c and d and their corresponding edge distributions $\{p_i^{(c)}\}$ and $\{p_i^{(d)}\}$ we calculate,

$$J_p(c,d) = \sum_{p_i^{(c)} > 0, p_i^{(d)} > 0} \left(\sum_j \max\left(\frac{p_j^{(c)}}{p_i^{(c)}}, \frac{p_j^{(d)}}{p_i^{(d)}} \right) \right)^{-1}.$$

J_p has several nice features including scale invariance, it is not lower than ordinary Jaccard index values for discrete uniform distributions (ordinary sets) and $1 - J_p$ is a proper metric on probability distributions, meaning that J_p could be used as a similarity index in e.g. hierarchical clustering and there will be no such effects like tree inversions.

Read extraction and MAG reassembly

In addition to providing multiple scaffold binning, accurate multiple edge binning provides an opportunity to improve upon existing metagenomic assembly using read extraction from a paired-end library provided to BinSPreader. For read extraction, we utilize an approach adapted from (Uritskiy et al., 2018) from contigs down to edges. Let $\tilde{F}$ be a refined multiple edge binning and $E_j(F)$ be a set of assembly graph edges e_i that contain bin c_j with weight $\tilde{F}_{ij} > t$, where t is a reassembly weight threshold with default value 0.1. We then align a set of reads from paired-end library to edges $E_j(F)$ separately for every bin c_j obtaining a set of read-pairs R_j, which includes all read-pairs where at least one read aligned to $E_j(F)$. This set of reads could be further reassembled or analyzed as necessary.