# SCAPP: An algorithm for improved plasmid assembly in metagenomes

**David Pellow [1,*], Maraike Probst[2], Ori Furman [3], Alvah Zorea [3], Arik Segal [4,5], Itzik Mizrahi [3], and Ron Shamir [1,*]**

[1] Blavatnik School of Computer Science, Tel Aviv University, Tel Aviv, 6997801, Israel
[2] Institute of Microbiology,Unversity of Innsbruck, Innsbruck, A-6020, Austria.
[3] Life Sciences, Ben-Gurion University of the Negev, Beer-Sheva, 8410501, Israel.
[4] Health Sciences, Ben-Gurion University of the Negev, Beer-Sheva, 8410501, Israel.
[5] Soroka University Medical Center, Beer-Sheva, 8410501, Israel.

*To whom correspondence should be addressed.

## Abstract

**Motivation:** Metagenomic sequencing has led to the identification and assembly of many new bacterial genome sequences. These bacteria often contain plasmids: small, circular DNA molecules that may transfer across bacterial species and confer antibiotic resistance and are less studied and understood. In order to assist in the study of plasmids we developed SCAPP (Sequence Contents-Aware Plasmid Peeler) - an algorithm and tool to assemble plasmid sequences from metagenomic sequencing.
**Results:** SCAPP builds on some key ideas from the Recycler algorithm while improving plasmid assemblies by integrating biological knowledge about plasmids. We compared the performance of SCAPP to Recycler and metaplasmidSPAdes on simulated metagenomes, real human gut microbiome samples, and a human gut plasmidome dataset that we generated. We also created plasmidome and metagenome data from the same cow rumen sample and used it to create a novel assessment procedure. In most cases SCAPP outperformed Recycler and metaplasmidSPAdes across this wide range of datasets.
**Availability:** `https://github.com/Shamir-Lab/SCAPP`
**Contact:** {dpellow,rshamir}@tau.ac.il

## 1 Introduction

Plasmids play a critical role in microbial adaptation, such as antibiotic resistance or other metabolic capabilities, and genome diversification through horizontal gene transfer. However, plasmid evolution and ecology across different microbial environments and populations are poorly characterized and understood. Thousands of plasmids have been sequenced and assembled directly from isolated bacteria, but constructing complete plasmid sequences from short read data remains a hard challenge. The task of assembling plasmid sequences from shotgun metagenomic sequences, which is our goal here, is even more daunting.

There are several reasons for the difficulty of plasmid assembly. First, plasmids represent a very small fraction of the sample's DNA and thus may not be fully covered by the read data in high-throughput sequencing experiments. Second, they often share sequences with the bacterial genomes and with other plasmids, resulting in tangled assembly graphs. For these reasons, plasmids assembled from bacterial isolates are usually incomplete, fragmented into multiple contigs, and contaminated with sequences from other sources. The challenge is reflected in the title of a recent review on the topic: "On the (im)possibility of reconstructing plasmids from whole-genome short-read sequencing data" (Arredondo-Alonso *et al.*, 2017). In a metagenomic sample, these problems are accentuated since the assembly graphs are much larger, more tangled and fragmented.

There are a number of tools that can be used to assemble or detect plasmids in isolate bacterial samples, such as plasmidSPAdes (Antipov *et al.*, 2016), PlasmidFinder (Carattoli *et al.*, 2014), cBar (Zhou and Xu, 2010), gPlas (Arredondo-Alonso *et al.*, 2019), and others. However, there are currently only two tools that attempt to reconstruct complete plasmid sequences in metagenomic samples: Recycler (Rozov *et al.*, 2017) and metaplasmidSPAdes (Antipov *et al.*, 2019). metaplasmidSPAdes iteratively generates smaller and smaller subgraphs of the assembly graph by removing contigs with coverage below a threshold that increases in each iteration. As lower coverage segments of the graph are removed, longer contigs may be constructed in the remaining subgraph. Cyclic contigs are considered as putative plasmids and then verified using the profile of their genetic contents. The main idea behind Recycler is that a single shortest circular path through each node in the assembly graph can be found efficiently. The circular paths that have uniform read coverage are iteratively "peeled" off the graph and reported as possible plasmids. The peeling process reduces the residual coverage of each involved node, or remove it altogether.

**Box 1. Overview of SCAPP**

```
Annotate the assembly graph:
  a: Map reads to nodes of the assembly graph
  b: Find nodes with plasmid gene matches
  c: Assign plasmid sequence score to nodes
for each strongly connected component do
    Iteratively peel uniform coverage cycles through plasmid gene nodes
    Iteratively peel uniform coverage cycles through high scoring nodes
    Iteratively peel shortest cycle through each remaining node if it meets plasmid criteria
Output the set of confident plasmid predictions
```

Here we present SCAPP (Sequence Contents-Aware Plasmid Peeler), a new algorithm building on Recycler that leverages external biological knowledge about plasmid sequences. In SCAPP the assembly graph is annotated with plasmid-specific genes (PSGs) and nodes are assigned weights reflecting the chance that they are plasmidic based on a plasmid sequence classifier (Pellow *et al.*, 2020). In the annotated assembly graph we prioritize circular paths that include plasmid genes and highly probable plasmid sequences. SCAPP also uses the PSGs and plasmid scores to filter out likely false positives from the set of potential plasmids.

## 2 Methods

SCAPP accepts as input a metagenomic assembly graph, with nodes representing the sequences of assembled contigs and edges representing $k$-long sequence overlaps between contigs, and the paired-end reads from which the graph was assembled. SCAPP processes each component of the assembly graph and iteratively assembles plasmids from them. The output of SCAPP is a set of cyclic sequences representing confident plasmid assemblies.

A high-level overview of SCAPP is provided in **Box 1** and **Fig. S1** in **Supplement S1**; the full algorithmic details are presented below. For brevity, we describe only default parameters below, see **Supplement S2** for alternatives. SCAPP is available from `https://github.com/Shamir-Lab/SCAPP`, and fully documented there.

### 2.1 The SCAPP algorithm

The full SCAPP algorithm is given in **Algorithm 1**. The peel function, which defines how cycles are peeled from the graph, is given in **Algorithm 2**.

**Algorithm 1** SCAPP pipeline

```
Input: Assembly graph $G = (V, E)$ and read set R of the sample
Output: $P$: potential plasmids, $O$: confident plasmid predictions
Create annotated graph $G' = (V', E')$:
  a: Initially $G' = G$
  b: Map $R$ to $V'$
  c: $score(v) \leftarrow$ sequence plasmid probability $\forall v \in V'$
  d: $w(v) = (1 - score(v))/len(v) \cdot cov(v) \ \forall v \in V'$
  e: $V^m = \{v \in V' | v$ contains a PSG$\}$, $w(v) = 0 \ \forall v \in V^m$
$V' \leftarrow V' \setminus \{v \in V' | \, deg(v) = 0 \vee v$ is probable chromosome node
  $\vee \, v$ is a non-compatible self-loop with $indeg(v) = outdeg(v) = 1\}$
$P \leftarrow \{v \in V' | v$ is a compatible self-loop$\}$
for each strongly connected component $CC \in G'$ do
    for $v \in V^m \cap CC$ in decreasing order by $len(v) \cdot cov(v)$ do
        Find lowest weight cycle $C$ through $v$
        if $C$ meets coverage and paired-end read criteria then
            $P \leftarrow P \cup \{C\}, G' \leftarrow peel(G', C)$
    for $v \in \{v \in CC | \, v$ is a probable plasmid node$\}$
    in decreasing order by $len(v) \cdot cov(v)$ do
        Find lowest weight cycle $C$ through $v$
        if $C$ meets coverage and paired-end read criteria then
            $P \leftarrow P \cup \{C\}, G' \leftarrow peel(G', C)$
    while $V'$ changes do
        $S \leftarrow \{\}$
        for $v \in V' \cap CC$ in decreasing order by $len(v) \cdot cov(v)$ do
            Find lowest weight cycle $C$ through $v$
            $S \leftarrow S \cup C$
        for $C \in S$ in increasing order of coefficient of variation do
            if $C$ meets coverage and paired-end read criteria then
                $P \leftarrow P \cup \{C\}, G' \leftarrow peel(G', C)$
$O \leftarrow \{C \in P | (C$ contains a PSG $\wedge \, plasmid \ score(C) > 0.5)$
  $\vee \, (C$ contains a PSG $\wedge \, C$ is self-loop $)$
  $\vee \, (plasmid \ score(C) > 0.5 \wedge C$ is self-loop $)\}$
```

**Algorithm 2** $peel(G, C)$

```
Input: Assembly graph $G = (V, E)$ annotated with node coverage,
       cycle $C \subset G$
Output: Updated graph $G' = (V' \subseteq V, E' \subseteq E)$ with cycle $C$ peeled
$G' = G$
$\mu_{cov'}(C) = \sum_{u \in C} f(u, C) cov'(u, C)$, the weighted mean of the
  discounted coverage of $C$ in $G$
for $v \in C$ do
    $cov(v) \leftarrow max\{cov(v) - \mu_{cov'}(C), 0\}$
    if $cov(v) = 0$ then
        $V' \leftarrow V' \setminus v$
        $E' \leftarrow E' \setminus \{e | e = (u, v) \cup e = (v, u) \, \forall u \in V\}$
```

### 2.2 Read mapping

The first step in creating the annotated assembly graph is to align the reads to the contigs in the graph. The links between paired-end reads aligning across contig junctions are used to evaluate potential plasmid paths in the graph. Read alignment is performed using BWA (Li, 2013) and the alignments are filtered to retain only primary read mappings, sorted, and indexed using SAMtools (Li *et al.*, 2009).

### 2.3 Plasmid-specific gene annotation

We created sets of PSGs by database mining and curation by plasmid microbiology experts from the Mizrahi Lab (Ben-Gurion University). Information about these PSG sets is found in **Supplement S3**. The sequences themselves are available from `https://github.com/Shamir-Lab/SCAPP/scapp/data`.

A node in the assembly graph is annotated as containing a PSG hit if there is a BLAST match between one of the PSG sequences and the sequence corresponding to the node ($\geq 75\%$ sequence identity along $\geq 75\%$ of the length of the gene).

### 2.4 Plasmid score annotation

We use PlasClass (Pellow *et al.*, 2020) to annotate each node in the assembly graph with a plasmid score. PlasClass uses a set of logistic regression classifiers for sequences of different lengths to assign a classification score reflecting the likelihood of each node to be of plasmid origin.

We re-weight the node scores according to the sequence length as follows. For a given sequence of length $L$ and plasmid probability $p$ assigned by the classifier, the re-weighted plasmid score is: $s = 0.5 + \frac{p-0.5}{1+e^{-0.001(L-2000)}}$. This tends to pull scores towards 0.5 for short sequences, for which there is lower confidence, while leaving scores of longer sequences practically unchanged.

Long nodes ($L > 10$ kbp) with low plasmid score ($s < 0.2$) are considered probable chromosomal sequences and are removed, simplifying the assembly graph. Similarly, long nodes ($L > 10$ kbp) with high plasmid score ($s > 0.9$) are considered probable plasmid nodes.

## 2.5 Assigning node weights

In order to apply the peeling idea, nodes are assigned weights so that lower weights correspond to higher likelihood to be assembled into a plasmid. Plasmid score and PSG annotations are incorporated into the node weights. Each node is assigned a weight $w(v) = (1-s)/C \cdot L$ where $C$ is the depth of coverage of the node's sequence and $L$ is the sequence length. This gives lower weight to more highly covered, longer nodes that have higher plasmid scores. Nodes with PSG hits are assigned a weight of zero, making them more likely to be integrated into any lowest-weight cycle in the graph that can pass through them.

## 2.6 Finding low-weight cycles in the graph

The core of the SCAPP algorithm is to iteratively find a lowest weight ("lightest") cycle going through each node in the graph for consideration as a potential plasmid. We use the bidirectional single-source, single-target shortest path implementation of the NetworkX Python package (Schult, 2008).

The order that nodes are considered matters since in each iteration potential plasmids are peeled from the graph, affecting the cycles that may be found in subsequent iterations. The plasmid annotations are used to decide the order that nodes are considered: first all nodes with PSGs, then all probable plasmid nodes, and then all nodes in the graph. If the lightest cycle going through a node meets certain criteria described below, it is peeled off, changing the coverage of nodes in the graph. Performing the search for light cycles in this order ensures that the cycles through more likely plasmid nodes will be considered before other cycles.

## 2.7 Assessing coverage uniformity

The lightest cyclic path, weighted as described above, going through each node is found and evaluated. Recycler sought a cycle with near uniform coverage, reasoning that all contigs that form a plasmid should have roughly the same coverage. However, this did not take into account the overlap of the cycle with other paths in the graph (see **Fig 1**). To account for this, we instead compute a discounted coverage score for each node in the cycle based on its interaction with other paths as follows:

The *discounted coverage* of a node $v$ in the cycle $C$ is its coverage $cov(v)$ times the fraction of the coverage on all its neighbors (both incoming and outgoing), $\mathcal{N}(v)$, that is on those neighbors that are in the cycle (see **Fig 1**):

$$cov'(v,C) = cov(v) \cdot \sum_{u \in C \wedge u \in \mathcal{N}(v)} cov(u) / \sum_{u \in \mathcal{N}(v)} cov(u)$$

A node $v$ in cycle $C$ with contig length $len(v)$ is assigned a weight $f$ corresponding to its fraction of the length of the cycle: $f(v,C) = len(v)/\sum_{u \in C} len(u)$. These weights are used to compute the weighted mean and standard deviation of the discounted coverage of the nodes in the cycle: $\mu_{cov'}(C) = \sum_{u \in C} f(u,C)cov'(u,C)$,

$$STD_{cov'} = \sqrt{\sum_{u \in C} f(u,C)(cov'(u,C) - \mu_{cov'}(C))^2}$$

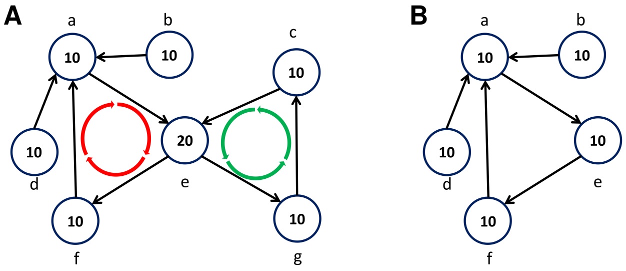

**Fig. 1.** Evaluating and peeling cycles. Numbers inside nodes indicate coverage. All nodes in the example have equal length. A: Cycles $(a, e, f)$ and $(c, e, g)$ have the same average coverage (13.33) and CV (0.35), but their discounted CV values differ: The discounted coverage of node $a$ is 6, and the discounted coverage of node $e$ is 10 in both cycles. The left cycle has discounted CV=0.22 and the right has discounted CV=0. By peeling off the mean discounted coverage of the right cycle (10) one gets the graph in B. Note that nodes $g$, $c$ were removed from the graph since their coverage was reduced to 0, and the coverage of node $e$ was reduced to 10.

The coefficient of variation of $C$, which evaluates its coverage uniformity, is the ratio of the standard deviation to the mean:

$$CV(C) = \frac{STD_{cov'}(C)}{\mu_{cov'}(C)}$$

## 2.8 Finding potential plasmid cycles

Once the set of lightest cycles has been generated, each cycle is evaluated as a potential plasmid based on its structure in the assembly graph, the PSGs it contains, its plasmid score, paired-end read links, and coverage uniformity. The precise evaluation criteria are described in **Supplement S4**. A cycle that passes them is defined as a potential plasmid. The potential plasmid cycles are peeled from the graph in each iteration as defined in **Algorithm 2** (see also **Fig 1**).

## 2.9 Filtering confident plasmid assemblies

In the final stage of SCAPP, PSGs and plasmid scores are used to filter out likely false positive plasmids from the output and create a set of confident plasmid assemblies. All potential plasmids are assigned a length-weighted plasmid score and are annotated with PSGs as was done for the contigs during graph annotation. Potential plasmids that belong to at least two of the following sets are reported as confident plasmids: (a) potential plasmids containing a match to a PSG; (b) potential plasmids with plasmid score $> 0.5$; (c) self-loop nodes.

# 3 Results

We tested SCAPP on simulated metagenomes, human gut metagenomes, a human gut plasmidome dataset that we generated and also on parallel metagenome and plasmidome datasets from the same cow rumen microbiome specimen that we generated (details in **Supplement S5**). The test settings and evaluation methods are described in **Supplement S6**.

## 3.1 Simulated metagenomes

We created five read datasets simulating metagenomic communities of bacteria and plasmids and assembled them. Datasets pf oncreasing

| | | | | Recycler | | mpSpades | | SCAPP | |
|---|---|---|---|---|---|---|---|---|---|
| Sample | # genomes | # plasmids | # unique (median length) | # plasmids (median length) | F1 | # plasmids (median length) | F1 | # plasmids (median length) | F1 |
| Sim1 | 30 | 82 | 56 (69.7) | 14 (5.3) | 0.0 | 24 (19.1) | 12.5 | 38 (49.8) | 2.1 |
| Sim2 | 180 | 333 | 219 (70.3) | 39 (3.5) | 3.1 | 23 (15.0) | 9.1 | 65 (28.1) | 5.0 |
| Sim3 | 320 | 745 | 497 (53.4) | 58 (6.7) | 3.6 | 36 (14.6) | 3.8 | 112 (29.8) | 6.0 |
| Sim4 | 450 | 1024 | 644 (55.0) | 81 (5.3) | 3.6 | 96 (3.5) | 4.6 | 147 (28.9) | 4.1 |
| Sim5 | 625 | 1365 | 886 (50.4) | 99 (3.7) | 4.6 | 68 (6.6) | 7.1 | 152 (26.4) | 5.2 |

Table 1. Performance on simulated metagenome datasets. The number of unique plasmids (# unique) accounts for plasmids with copy number greater than one. Median lengths of the plasmids (in kbp) are reported in parentheses.

complexity were created, including two simple ($<$ 200 genomes) and three more complex ones. We randomly selected bacterial genome references from RefSeq that contained long ($>$ 10 kbp) plasmids, along with the associated plasmids and used realistic distributions for genome abundance and plasmid copy number (described in **Supplement S6**). Paired-end short reads (read length = 126 bp) were simulated from the genome references using InSilicoSeq (Gourlé *et al.*, 2018) with the HiSeq error model and assembled. 25M paired-end reads were generated for Sim1, Sim2 and Sim3, and 50M for Sim4 and Sim5.

**Table 1** presents features of the simulated datasets and reports the performance of Recycler, mpSpades, and SCAPP on them. For brevity we report only F1 scores; precision and recall scores are reported in **Supplement S7**. All tools had low F1 scores, although mpSpades achieved higher scores on four of the simulations. On one of the more complex simulations, SCAPP performed best. Note that SCAPP assembled many more and much longer plasmids than the other tools (performance of the tools stratified by plasmid length is presented in **Supplement S7**). There was relatively little overlap between the plasmids discovered by each tool as shown in **Supplement S7** as well.

## 3.2 Human gut microbiomes

We assembled plasmid sequences in twenty publicly available human gut microbiome samples selected from the study of Vrieze *et al.* (2012). There is no gold standard set of plasmids for these samples to measure performance against. Instead, we matched all assembled contigs to PLSDB (Galata *et al.*, 2018) and considered the set of the database plasmids that were covered by the contigs as the gold standard (see **Supplement S6** for details).

**Fig 2** summarizes the results of the three algorithms on the twenty samples. The mean F1 score of SCAPP across the 20 samples was 16.1, while mpSpades and Recycler achieved 10.3 and 10.9, respectively. SCAPP performed best in more cases, with mpSpades failing to assemble gold standard plasmid sequences in over half the samples. We note that all of the cases where SCAPP failed to report any of the gold standard plasmids occurred when the number of gold standard plasmids was very small and the other tools also failed to assemble them. On the largest samples with the most gold standard plasmids SCAPP performed best, highlighting its superior performance on the types of samples likely to be used in experiments aimed at plasmid assembly. The numbers of plasmids assembled by each tool and their median lengths are reported in **Supplement S8**. The average median plasmid lengths across all samples were 3.6, 5.0, and 4.4 kbp for Recycler, mpSpades, and SCAPP, respectively.

## 3.3 Human gut plasmidome

We sequenced the plasmidome of the human gut microbiome from a healthy adult male according to the protocol outlined in Brown Kav *et al.* (2013). Sample and sequencing details can be found in **Supplement S5**. This protocol was assessed to achieve samples with at least 65% plasmid contents by Krawczyk *et al.* (2018).

| Tool | # plasmids | median length | precision | recall | F1 |
|---|---|---|---|---|---|
| Recycler | 93 | 2.1 | 15.1 | 37.8 | 21.5 |
| SCAPP | 82 | 2.4 | 17.1 | 35.9 | 23.1 |
| mpSpades | 53 | 3.0 | 11.3 | 9.4 | 10.3 |

Table 2. Performance on the human gut plasmidome. Number of plasmids, the median plasmid length (in kbp), and performance measures for all tools.

We determined the gold standard set of plasmids as in the gut microbiome samples, resulting in 74 plasmids. Performance was computed as in the metagenomic samples and is shown in **Table 2**. mpSpades had lower precision and much lower recall than the other tools. SCAPP achieved better overall performance with higher precision and recall.

Notably, although the sample was obtained from a healthy donor, some of the plasmids reconstructed by SCAPP matched reference plasmids found in potentially pathogenic hosts such as *Klebsiella pneumoniae*, pathogenic serovars of *Salmonella enterica*, and *Shigella sonnei*. The detection of plasmids previously isolated from pathogenic hosts in the healthy gut indicates potential pathways for transfer of virulence genes.

We used MetaGeneMark (Zhu *et al.*, 2010) to find potential genes in the plasmids assembled by SCAPP. 294 genes were found, and we annotated them with the NCBI non-redundant (nr) protein database using BLAST. 46 of the plasmids contained 170 (58%) genes with matches in the database, of which 77 (45%) had known functional annotations, which we grouped manually in **Fig 3A**. There were six antibiotic and toxin resistance genes, all on plasmids that were not in the gold standard set, highlighting SCAPP's ability to find novel resistance carrying plasmids. 60 of the 77 genes (78%) with functional annotations had plasmid associated functions: replication, mobilization, recombination, resistance, and toxin-antitoxin systems. 29 out of the 33 plasmids that contained functionally annotated genes (88%) contained at least one of these plasmid associated functions. This provides a strong indication that SCAPP succeeded in assembling true plasmids.

We also examined the hosts that were annotated for the plasmid genes and found that almost all of the plasmids with annotations coded genes from a variety of hosts, which we refer to here as "broad-range" (see **Fig 3B**). Of the 40 plasmids with genes from annotated hosts, only 10 (25%) had genes with annotated hosts all within a single phylum. This demonstrates that these plasmids assembled and identified by SCAPP may be involved in one stage of transferring genes, such as the antibiotic resistance genes we detected, across a range of bacteria.

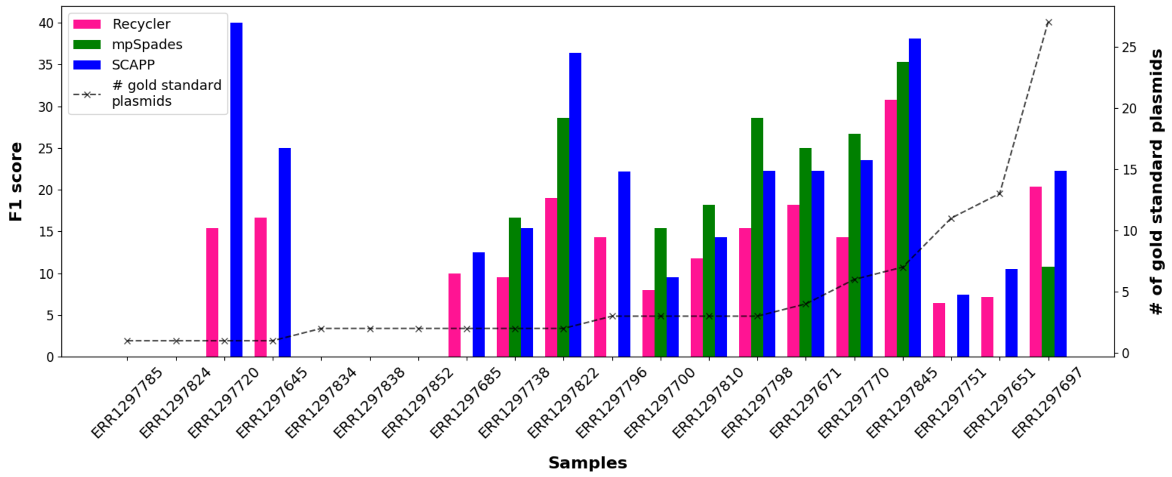

**Fig. 2.** F1 scores of the plasmids assembled by Recycler, mpSpades and SCAPP in the human gut microbiome samples (accessions given on x-axis), calculated using PLSDB plasmids as the gold standard. The dashed line shows the number of gold standard plasmids in each sample. Where bars are omitted the F1 score was 0.

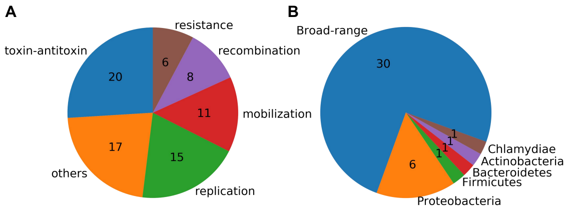

**Fig. 3.** Annotation of genes on the plasmids identified by SCAPP in the human gut plasmidome sample. A: Functional annotations of the plasmid genes. B: Host annotations of the plasmid genes. "Broad-range" plasmids had genes annotated with hosts from more than one phylum.

## 3.4 Parallel metagenomic and plasmidome samples

We performed two sequencing assays on the same cow rumen microbiome sample (details in **Supplement S5**). Total DNA and plasmid DNA (according to the protocol of Brown Kav *et al.* (2013)) were extracted from two subsamples, and sequenced and analyzed in parallel (see **Fig 4**). This enabled us to assess the plasmids assembled in the metagenome using the plasmidome. Because the plasmidome was from the same sample as the metagenome, it could provide a better assessment of performance than using PLSDB matches as the gold standard, especially as PLSDB tends to under-represent plasmids from non-clinical environments.

We ran the three plasmid discovery algorithms on both sets of sequences. The results are presented in **Table 3**. mpSpades made the fewest predictions and Recycler made the most. To compare the plasmids identified by the different tools, we considered two plasmids to be the same if their sequences matched at $> 80\%$ identity across $> 90\%$ of their length. The comparison is shown in **Fig S3** of **Supplement S9**. On the plasmidome subsample, 50 plasmids were identified by all three methods. Seventeen were common to the three methods in the metagenome. In both subsamples, the Recycler plasmids included all or almost all of those identified by the other methods plus a large number of additional plasmids. In the plasmidome, SCAPP and Recycler shared many more plasmids than mpSpades and Recycler.

Comparison of the assemblies to PLSDB (as was done for the human gut samples) gave very few results. The metagenome contained only one matching PLSDB reference plasmid, and none of the tools assembled it. The plasmidome had only seven PLSDB matches, and mpSpades, Recycler, and SCAPP had F1 scores of 2.86, 2.67, and 1.74, respectively.

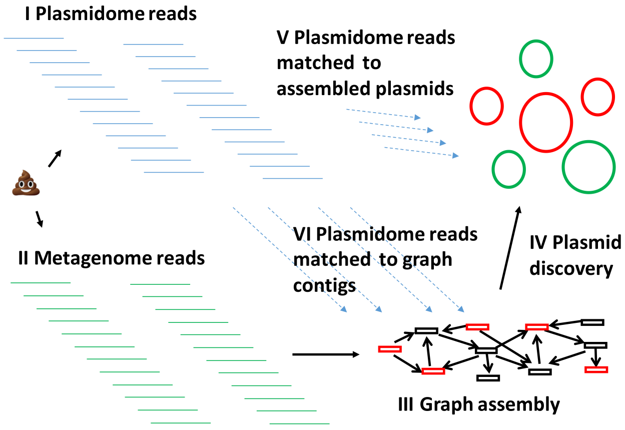
I Plasmidome reads
V Plasmidome reads matched to assembled plasmids
II Metagenome reads
VI Plasmidome reads matched to graph contigs
IV Plasmid discovery
III Graph assembly

**Fig. 4.** Outline of the read-based performance assessment. Plasmidome (I) and metagenome reads (II) are obtained from subsamples of the same sample. III: The metagenome reads are assembled into a graph. IV: The graph is used to detect and report plasmids by the algorithm of choice. V: The plasmidome reads are matched to assembled plasmids. Nearly fully matched plasmids (red) are used to calculate plasmid read-based precision. VI: The plasmidome reads are matched to the assembly graph contigs. Nearly fully covered contigs (red) are considered plasmidic. The fraction of total length of plasmidic contigs included in the detected plasmids gives the plasmidome read-based recall.

| | metagenome | | plasmidome | |
|---|---|---|---|---|
| Tool | # plasmids | median length | # plasmids | median length |
| Recycler | 60 | 4.3 | 147 | 1.7 |
| SCAPP | 25 | 5.8 | 110 | 1.8 |
| mpSpades | 26 | 6.2 | 65 | 2.0 |

Table 3. Number of plasmids assembled by each tool and their median lengths (in kbp) for the parallel metagenome and plasmidome samples.

The low numbers of PLSDB matches compared to the number of plasmids assembled demonstrate the potential of the tools to identify novel plasmids that are not in the database.

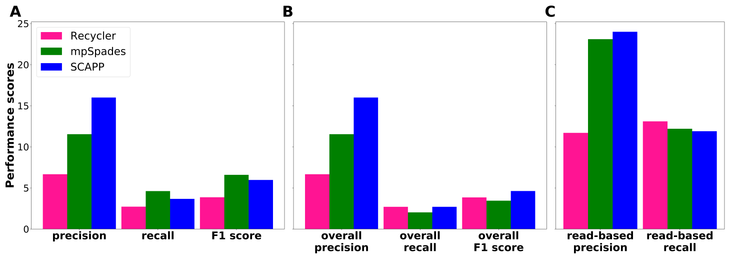

**Fig. 5.** Performance on the parallel datasets. A: Performance of each tool on the plasmids assembled from the metagenome using as gold standard the plasmids assembled from the plasmidome by the same tool. B: Overall performance on the plasmids assembled from the metagenome compared to the union of all plasmids assembled by all tools in the plasmidome. C: Plasmidome read-based performance.

We next compared the plasmids assembled by each tool in the two subsamples. For each tool, we considered the plasmids it assembled from the plasmidome to be the gold standard set, and computed scores as above for the plasmids it assembled in the metagenome, shown in **Fig 5A**. SCAPP had the highest precision. Since mpSpades had a much smaller gold standard set, it achieved higher recall and F1. Recycler output many more plasmids than the other tools in both samples, but had much lower precision, suggesting that many of its plasmid predictions may be spurious.

Next, we considered the union of the plasmids assembled across all tools as the gold standard set and recomputed the scores as before. We refer to them as “overall” scores. **Fig 5B** shows that overall precision scores were the same as in **Fig 5A**, while overall recall was lower for all the tools, as expected. mpSpades underperformed because of its smaller set of plasmids and SCAPP had the highest overall F1 score.

Finally, in order to fully leverage the power of parallel samples, we computed the performance of each tool on the metagenomic sample, using the reads of the plasmidomic sample, and not just the plasmids that the tools were able to assemble. We calculated the *plasmidome read-based precision* by mapping the plasmidomic reads to the plasmids assembled from the metagenomic sample (**Fig 4**). A plasmid with $> 90\%$ of its length covered by more than one plasmidomic read was considered to be a true positive. The *plasmidome read-based recall* was computed by mapping the plasmidomic reads to the contigs of the metagenomic assembly. Contigs with $> 90\%$ of their length covered by plasmidomic reads at depth $> 1$ were considered to be plasmidic. Plasmidic contigs that were integrated into the assembled plasmids were counted as true positives, and those that were not were considered false negatives. The recall was the fraction of the plasmidic contigs’ length that was integrated in the assembled plasmids. Note that the precision and recall here are measured using different units (plasmids and base pairs, respectively) so they are not directly related. For mpSpades, which does not output a metagenomic assembly, we mapped the contigs from the metaSPAdes assembly to the mpSpades plasmids using BLAST ($> 80\%$ sequence identity matches along $> 90\%$ of the length of the contigs).

The plasmidome read-based performance is presented in **Fig 5C**. The plasmidic contigs used as gold standard had a total length of 146.6 kbp. All tools achieved a similar recall of around 12. SCAPP and mpSpades performed similarly, with SCAPP having slightly higher precision (24.0 vs 23.1) but slightly lower recall (11.9 vs 12.2). Recycler had a bit higher recall (13.1), at the cost of lower precision (11.7). Hence, a much lower fraction of the plasmids assembled by Recycler in the metagenome were actually supported by the parallel plasmidome sample, adding to the other evidence that the false positive rate of Recycler exceeds that of other tools.

We assessed the significance of the improved plasmid assembly by SCAPP. Only one plasmid in PLSDB was covered by contigs in the metagenomic assembly, demonstrating the ability of SCAPP to identify novel plasmids that are not in the databases. Four of the plasmids assembled by SCAPP in the metagenome sample were also assembled with the same sequence in the plasmidome sample.

There were 293 contigs in the metagenomic assembly that were covered by plasmidomic reads, with a total length of 146.6 kbp. 17.4 kbp of this length were incorporated into plasmids assembled by SCAPP. In contrast, the plasmidome assembly had a total length of 8.9 Mbp. This clearly shows the potential advantage of plasmidome sequencing for determining plasmids. This is also apparent from the low recall seen in **Fig 5A**.

| Test | Ranking |
|---|---|
| Simple simulations (2) | mpSpades $\gg$ SCAPP $\gg$ Recycler |
| Complex simulations (3) | mpSpades $\approx$ SCAPP $>$ Recycler |
| Human gut metagenomes (20) | SCAPP $\gg$ mpSpades $>$ Recycler |
| Plasmidome | SCAPP $>$ Recycler $\gg$ mpSpades |
| Parallel: within tool | mpSpades $>$ SCAPP $\gg$ Recycler |
| Parallel: “overall”, across tools | SCAPP $>$ Recycler $>$ mpSpades |
| Parallel: precision | SCAPP $\approx$ mpSpades $\gg$ Recycler |
| Parallel: recall | Recycler $>$ mpSpades $\approx$ SCAPP |

Table 4. Summary of performance. Comparison of the performance of the tools on each of the datasets. When multiple samples were tested, the number of samples appears in parentheses, and average performance is reported. For the parallel samples results are for the evaluation of the metagenome based on the plasmidome, and precision and recall are plasmidome read-based. Unless otherwise stated, F1 score is used.

We detected potential genes in the plasmids assembled by SCAPP in the plasmidome sample and annotated them as we did for the human gut plasmidome. The gene function and host annotations are shown in **Fig S4** in **Supplement S9**. Out of 242 genes, only 34 genes from 17 of the plasmids had annotations, and only 18 of these had known functions, highlighting that many of the plasmids in the cow rumen plasmidome are as yet unknown. The high percentage of genes of plasmid function (15/18) indicates that SCAPP succeeded in assembling novel plasmids. Unlike in the human plasmidome, most of the plasmids with known host annotations had hosts from a single phylum.

## 3.5 Summary

We summarize the performance of the tools across all the test datasets in **Table 4**. The performance of two tools was considered similar (denoted $\approx$) if their scores were within 5% of each other. Performance of one tool was considered to be much higher than the other ($\gg$) if its score was $> 30\%$ higher.

We see that in the majority of the cases SCAPP was the highest performer. Moreover, in all cases except the two simple simulations, SCAPP performed best or close to the top performing tool.

## 3.6 Resource usage

The runtime and memory usage of the three tools are presented in **Table 5**. Recycler and SCAPP require assembly by metaSPAdes and pre-processing

| | | Runtime (minutes) | | |
|---|---|---|---|---|
| Dataset | Assembly peak RAM (GB) | Recycler | SCAPP | mpSpades |
| Human metagenomes | 20.7 | 115.4 | 130.1 | 102.8 |
| Plasmidome | 30.1 | 906.5 | 908.9 | 547.6 |
| Parallel metagenome | 148.1 | 2118.0 | 2229.7 | 2132.3 |
| Parallel plasmidome | 26.4 | 880.9 | 883.8 | 684.1 |

Table 5. Resource usage comparison for the three methods. Peak RAM of the assembly step (metaSPAdes for Recycler and SCAPP, metaplasmidSPAdes for mpSpades) in GB. Runtime (wall clock time, in minutes) is reported for the entire pipeline including assembly and any pre-processing and post-processing required. Human metagenome results are an average across the 20 samples.

of the reads and the resulting assembly graph. SCAPP also requires post-processing of the assembled plasmids. mpSpades requires post-processing of the assembled plasmids with the plasmidVerify tool. The reported runtimes are for the full pipelines necessary to run each tool – from reads to assembled plasmids.

In almost all cases assembly was the most memory intensive step, and so all tools achieved very similar peak memory usage (within 0.01 GB). Therefore, we report the RAM usage for this step.

The assembly step was also the longest step in all cases. SCAPP was slightly slower than Recycler as a result of the additional annotation steps, and mpSpades was 5 – 40% faster. However, note that mpSpades does not output a metagenomic assembly graph, so users interested in both the plasmid and non-plasmid sequences in a sample would need to run metaSPAdes as well, practically doubling the runtime.

Performance measurements were made on a 44-core, 2.2 GHz server with 792 GB of RAM. 16 processes were used where possible. Recycler is single-threaded, so only one process was used for it.

## 4 Conclusion

Plasmid assembly from metagenomic sequencing is a very difficult task, akin to finding needles in a haystack. This difficulty is demonstrated by the low numbers of plasmids found in real samples. Even in samples of the human gut microbiome, which is widely studied, relatively few plasmids from the extensive PLSDB plasmid database were recovered. Despite the challenges, SCAPP succeeded in assembling plasmids in real samples. SCAPP demonstrated generally improved performance over Recycler and mpSpades in a wide range of contexts. Specifically, SCAPP significantly outperformed mpSpades on a range of human gut metagenome and plasmidome samples, and significantly outperformed Recycler on a novel benchmark using parallel metagenomic and plasmidomic sequences. As the recall of all plasmid discovery tools is rather low, and the approaches of SCAPP and mpSpades are very different, a possible strategy is to run both tools in order to increase detection sensitivity.

SCAPP has several limitations. Like all de Bruijn graph-based assemblers, it may split a cycle into two when a shorter cycle is a sub-path of a longer cycle. It also has difficulties in finding very long plasmids, as these tend to not be completely covered and fragmented into many contigs in the graph. Note however that it produced longer cycles than Recycler. Compared to mpSpades, each algorithm produced longer cycles in different tests. Another limitation is the inherent bias in relying on known plasmid genes and plasmid databases, which tend to under-represent non-clinical samples. With further use of tools like SCAPP, perhaps with databases tailored to specific environments, further improvement is possible. In summary, by applying SCAPP across large sets of samples, many new plasmid reference sequences can be assembled, enhancing our understanding of plasmid biology and ecology.

## Funding

PhD fellowships from the Edmond J. Safra Center for Bioinformatics at Tel-Aviv University and Israel Ministry of Immigrant Absorption (to DP). Israel Science Foundation (ISF) grant 1339/18, US - Israel Binational Science Foundation (BSF) and US National Science Foundation (NSF) grant 2016694 (to RS), ISF grant 1947/19 and ERC Horizon 2020 research and innovation program grant 640384 (to IM).

## References

Antipov, D. *et al.* (2016). plasmidSPAdes: assembling plasmids from whole genome sequencing data. *Bioinformatics*, **32**(22), 3380–3387.

Antipov, D. *et al.* (2019). Plasmid detection and assembly in genomic and metagenomic data sets. *Genome research*, **29**(6), 961–968.

Arredondo-Alonso, S. *et al.* (2017). On the (im)possibility of reconstructing plasmids from whole-genome short-read sequencing data. *Microbial genomics*, **3**(10), e000128.

Arredondo-Alonso, S. *et al.* (2019). gplas: a comprehensive tool for plasmid analysis using short-read graphs. *bioRxiv*.

Brown Kav, A. *et al.* (2013). A method for purifying high quality and high yield plasmid dna for metagenomic and deep sequencing approaches. *Journal of microbiological methods*, **95**(2), 272–279.

Carattoli, A. *et al.* (2014). In silico detection and typing of plasmids using PlasmidFinder and plasmid multilocus sequence typing. *Antimicrobial agents and chemotherapy*, **58**(7), 3895–3903.

Galata, V. *et al.* (2018). PLSDB: a resource of complete bacterial plasmids. *Nucleic acids research*, **47**(D1), D195–D202.

Gourlé, H. *et al.* (2018). Simulating illumina metagenomic data with insilicoseq. *Bioinformatics*, **35**(3), 521–522.

Krawczyk, P. S. *et al.* (2018). PlasFlow: predicting plasmid sequences in metagenomic data using genome signatures. *Nucleic acids research*, **46**(6), e35.

Li, H. (2013). Aligning sequence reads, clone sequences and assembly contigs with bwa-mem. *arXiv preprint arXiv:1303.3997*.

Li, H. *et al.* (2009). The sequence alignment/map format and samtools. *Bioinformatics*, **25**(16), 2078–2079.

Pellow, D. *et al.* (2020). PlasClass improves plasmid sequence classification. *PLoS computational biology*, **16**(4), e1007781.

Rozov, R. *et al.* (2017). Recycler: an algorithm for detecting plasmids from de novo assembly graphs. *Bioinformatics*, **33**(4), 475–482.

Schult, D. A. (2008). Exploring network structure, dynamics, and function using NetworkX. In *In Proceedings of the 7th Python in Science Conference (SciPy*. Citeseer.

Vrieze, A. *et al.* (2012). Transfer of intestinal microbiota from lean donors increases insulin sensitivity in individuals with metabolic syndrome. *Gastroenterology*, **143**(4), 913–916.

Zhou, F. and Xu, Y. (2010). cBar: a computer program to distinguish plasmid-derived from chromosome-derived sequence fragments in metagenomics data. *Bioinformatics*, **26**(16), 2051–2052.

Zhu, W. *et al.* (2010). Ab initio gene identification in metagenomic sequences. *Nucleic acids research*, **38**(12), e132–e132.

# Supplementary information for SCAPP: An algorithm for improved plasmid assembly in metagenomes

## S1 SCAPP algorithm overview

Fig. S1 graphically outlines the SCAPP algorithm.

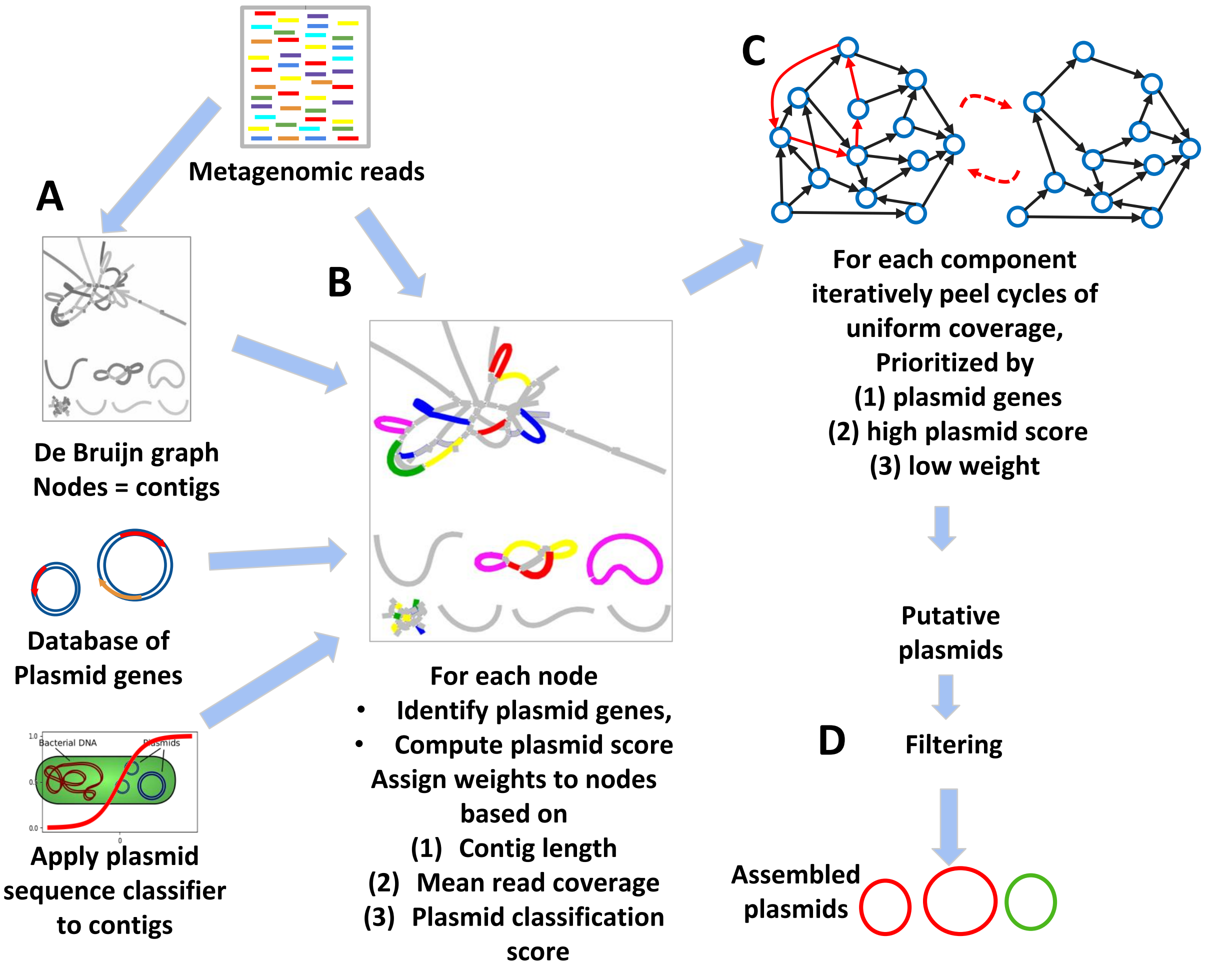

**Fig. S1.** Graphical overview of the SCAPP algorithm. A: The metagenomic assembly graph is created from the sample reads. B: The assembly graph is annotated with read mappings, presence of plasmid specific genes, and node weights based on sequence length, coverage, and plasmid classifier score. C: Potential plasmids are iteratively peeled from the assembly graph. An efficient algorithm finds cyclic paths in the annotated assembly graph that have low weight and high chance of being plasmids.Cycles with uniform coverage are peeled. D: Confident plasmid predictions are retained using plasmid sequence classification and plasmid-specific genes to remove likely false positive potential plasmids.

## S2 Alternatives for user set parameters

The SCAPP pipeline is highly flexible, and many of the options and parameters can be set by the user. In most cases, we recommend using the default options and settings. Some of the alternatives that can be chosen by the user are described below. All of the parameter settings that may be changed by the user are fully documented at: `https://github.com/Shamir-Lab/SCAPP`.

**Read mapping:** The user has the option of providing a sorted and indexed BAM alignment file created by any method.

**Plasmid-specific genes:** The user may add any set of PSGs or remove any of those included with SCAPP.

**Plasmid classification scores:** The sequences may be classified using PlasFlow and the PlasFlow classification output file can be provided to SCAPP.

**Algorithm thresholds:** Thresholds for finding plasmid gene matches, defining probable plasmid and chromosomal sequences, identifying potential plasmids, filtering them, and many more can all be user-defined. The full software documentation at `https://github.com/Shamir-Lab/SCAPP` details all of these user options.

## S3 Plasmid-specific genes

We created four sets of plasmid-specific genes (PSGs) by database mining and expert curation:

1. MOB genes: 890 amino acid sequences of plasmid maintenance genes curated by plasmid biologists from the Mizrahi Lab (Ben-Gurion University) and filtered computationally (see details of filtering below).
2. Plasmid ORFs: 4276 nucleotide sequences corresponding to ORFs annotated with 'mobilization', 'conjugation', 'partitioning', 'toxin-antitoxin', 'replication', or 'recombination' from a large set of putative plasmids found by the Mizrahi Lab and then filtered computationally.
3. ACLAME plasmid genes: 4813 nucleotide sequences of genes that make up 96 gene families in the ACLAME database (Leplae *et al.*, 2009) that were manually selected as possibly plasmid-specific. The set of genes was deduplicated and filtered computationally.
4. PLSDB-specific ORFs: 94478 plasmid-specific sequences determined as follows: We used MetaGeneMark (Zhu *et al.*, 2010) to predict genes in the plasmid sequences from PLSDB (v.2018_12_05) (Galata *et al.*, 2018). We then counted the number of BLAST matches ($> 75\%$ identity match along $> 75\%$ of the gene length) to these genes in both PLSDB and bacterial reference genomes from NCBI (downloaded January 9, 2019 ). We considered each predicted gene that appeared in the plasmids more than 20 times and was $> 20\times$ more prevalent in the plasmids than in the genomes to be plasmid-specific.

Sets 1–3 were filtered as follows: We counted matches between the sequences and PLSDB plasmids and NCBI bacterial reference genomes as for the PLSDB-specific ORFs (set 4). We excluded any gene that had more than 4 matches to bacterial genes *and* met one of the following conditions: (1) $\leq 4$ matches to plasmid genes and $> 4\times$ as many matches to bacterial genes as plasmid genes; or, (2) $> 4$ plasmid gene matches, but $\leq 4\times$ as many matches to plasmid genes as to bacterial genes.

## S4 Potential plasmid cycle criteria

Once the set of lightest cycles has been generated, each cycle is evaluated as a potential plasmid based on its structure in the assembly graph, the PSGs it contains, its plasmid score, paired-end read links, and coverage uniformity. A cycle is defined as a potential plasmid if one of the following criteria is met:

1. The cycle is formed by an isolated "compatible" self-loop node $v$, i.e.$len(v) > 1000$, $indeg(v) = outdeg(v) = 1$, and at least one of the following conditions holds:

   a. $v$ has a high plasmid score $s(v) > 0.9$.
   b. $v$ has a PSG hit.
   c. $< 10\%$ of the paired-end reads with a mate on $v$ have the other mate on a different node.

2. The cycle is formed by a connected compatible self-loop node $v$, i.e.$len(v) > 1000$, $indeg(v) > 1$ or $outdeg(v) > 1$, and $< 10\%$ of the paired-end reads with a mate on $v$ have the other mate on a different node.
3. The cycle is not formed by a self-loop and has:

   a. Uniform coverage: $CV(C) < 0.5$, and
   b. Consistent mate-pair links: a node in the cycle is defined as an "off-path dominated" node if the majority of the paired-end reads with one mate on the node have the other mate on a node that is not in the cycle. If less than half the nodes in the cycle are "off-path dominated", then we consider the mate-pair links to be consistent.

## S5 Sample and sequencing details

We sequenced the plasmidome of a human gut microbiome sample from a healthy adult male and the plasmidome and metagenome of a single cow rumen microbiome sample from a 4 month old calf. These sequences will be made publicly available upon publication. Sequencing of all samples was performed on the Illumina HiSeq2000 platform with a read length of 150bp and insert size 1000. 18,616,649 reads were sequenced in the human gut plasmidome, 27,127,784 in the cow rumen plasmidome, and 54,292,256 in the cow rumen metagenome sample.

Sequencing of the human gut microbiome was approved by the local ethics committee of Clalit HMO, approval number 0266-15-SOR. Extraction and sequencing of the cow rumen microbiome was approved by the local ethics committee of the Volcani Center, approval numbers 412/12IL and 566/15IL.

## S6 Experimental settings and evaluation

To create the simulated metagneomes, we randomly selected bacterial genome references from RefSeq that contained long ($> 10$ kbp) plasmids, along with the associated plasmids and used realistic distributions for genome abundance and plasmid copy number. For genome abundance we used the log-normal distribution, normalized so that the relative abundances sum to 1. This long-tailed distribution mimics the abundance distribution of real microbiome samples. For plasmid copy number we used a geometric distribution with parameter $p = min(1, log_{10}(L)/7)$ where $L$ is the plasmid length. This makes it less likely for a long plasmid to have a copy number above 1, while shorter plasmids can have higher copy numbers.

All metagenomes were assembled using the SPAdes assembler (v3.13) with the `--meta` option. The default of 16 threads were used, and the maximum memory was set to 750 GB. metaplasmidSPAdes (mpSpades) was run with the same parameters. mpSpades internally chooses the maximal value of $k$ to use for the $k$-mer length in the assembly graph. We matched the values of $k$ used in SPAdes to these values for each dataset. Defaults were used for all other options for Recycler and SCAPP. In practice, the maximum $k$ value was 77 for the simulations and human metagenomic samples, and 127 for the plasmidome and parallel metagenome-plasmidome samples.

| | Recycler | | | | mpSpades | | | | SCAPP | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Sample | # plasmids (median length) | precision | recall | F1 | # plasmids (median length) | precision | recall | F1 | # plasmids (median length) | precision | recall | F1 |
| Sim1 | 14 (5.3) | 0.0 | 0.0 | 0.0 | 24 (19.1) | 20.8 | 8.9 | 12.5 | 38 (49.8) | 2.6 | 1.8 | 2.1 |
| Sim2 | 39 (3.5) | 5.1 | 0.9 | 1.6 | 23 (15.0) | 47.8 | 5.0 | 9.1 | 65 (28.1) | 10.8 | 3.2 | 5.0 |
| Sim3 | 58 (6.7) | 17.2 | 2.0 | 3.6 | 36 (14.6) | 27.8 | 2.0 | 3.8 | 112 (29.8) | 16.1 | 3.4 | 6.0 |
| Sim4 | 81 (5.3) | 16.0 | 2.0 | 3.6 | 96 (3.5) | 17.7 | 2.7 | 4.6 | 147 (28.9) | 10.9 | 2.6 | 4.1 |
| Sim5 | 99 (3.7) | 22.2 | 2.5 | 4.6 | 68 (6.6) | 48.5 | 3.8 | 7.1 | 152 (26.4) | 17.1 | 3.1 | 5.2 |

Table 1. Full performance on simulated metagenome datasets. Median lengths of the plasmids assembled by each tool (in kbp) are reported in parentheses.

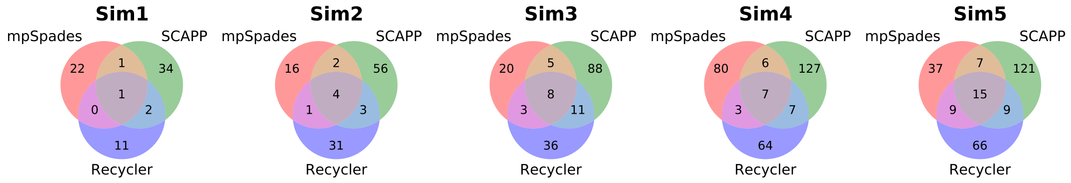

**Fig. S2.** Overlap of the plasmids assembled by the tools on each of the simulated metagenomes.

For a simulated metagenome, the set of plasmids included in the simulation was used as the gold standard. We used BLAST to match the assembled plasmids to the reference plasmid sequences. A plasmid assembled by one of the tools was considered to be a true positive if $> 90\%$ of its length was covered by BLAST matches to $> 90\%$ of a reference with $> 80\%$ sequence identity. The rest of the assembled plasmids were considered to be false positives. Gold standard plasmids that did not have assembled plasmids matching them were considered to be false negatives. Precision was defined as $TP/(TP + FP)$ and recall was defined as $TP/(TP + FN)$, where $TP$, $FP$, and $FN$ were the number of true positive, false positive, and false negative plasmids, respectively. The F1 score was defined as the harmonic mean of precision and recall.

For the human microbiome and plasmidome samples, the set of plasmids serving as the gold standard was selected from PLSDB (v.2018_12_05) (Galata *et al.*, 2018). The contigs from the metaSPAdes assembly were matched against the plasmids in PLSDB using BLAST. Matches between a contig and a reference plasmid with sequence identity $> 85\%$ were marked and a contig was said to match a reference if $> 85\%$ of its length was marked. Reference plasmids with $> 90\%$ of their lengths covered by marked regions of the matching contigs were used as the gold standard.

The set of plasmids assembled by each method was compared to the gold standard set using BLAST. A predicted plasmid was considered a true positive if there were sequence matches at $> 80\%$ identity between the plasmid and a gold standard plasmid that covered more than 90% of their lengths.

Note that in the case of the real samples, if two assembled plasmids matched to the same reference gold standard plasmid sequence(s), then one of them was considered to be a false positive. This strict definition penalized methods for unnecessarily splitting potential plasmid genomes into multiple different plasmids. If there were multiple gold standard reference plasmids that were matched to a single assembled plasmid, then none of them was considered as a false negative. The precision, recall, and F1 score were calculated as for the simulation.

For the parallel metagenome-plasmidome sample, plasmidomic reads were aligned to the plasmid sequences and metagenome assembly contigs using BWA (Li, 2013). Coverage at each base of each metagenomic contig was called using bedtools (Quinlan and Hall, 2010).

To compare the overlap between plasmids identified by the different tools, we considered two plasmids to be the same if their sequences matched at $> 80\%$ identity across $> 90\%$ of their length.

## S7 Extended results for simulated datasets

**Table 1** reports the full precision, recall, and F1 performance results for all tools on the simulated metagenome datasets. **Table 2** reports the performance results when stratified by length. **Figure S2** shows the limited overlap between the plasmids assembled by each tool in the simulated metagenomes.

## S8 Extended results for human metagenomes

**Table 3** reports the number of plasmids assembled by each tool and the median plasmid length for each of the human gut microbiome samples.

## S9 Extended results for parallel plasmidome-metagenome

Fig. S3 shows the overlap between the plasmids assembled by the tools in the parallel cow rumen plasmidome and metagenome samples.

Fig. S4 shows the annotations of the gene functions and hosts for the plasmids assembled in the rumen plasmidome.

| | | Recycler | | | | mpSpades | | | | SCAPP | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Sample | Length bin (# gold-standard) | # plasmids | precision | recall | F1 | # plasmids | precision | recall | F1 | # plasmids | precision | recall | F1 |
| Sim1 | 1-3 kb (0) | 2 | 0 | - | 0 | 2 | 0 | - | 0 | 0 | 0 | 0 | 0 |
| | 3-5 kb (4) | 4 | 0 | 0 | 0 | 2 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| | 5-10 kb (2) | 4 | 0 | 0 | 0 | 2 | 0 | 0 | 0 | 8 | 0 | 0 | 0 |
| | 10-20 kb (3) | 3 | 0 | 0 | 0 | 5 | 20.0 | 33.3 | 24.0 | 2 | 0 | 0 | 0 |
| | 20-50 kb (11) | 1 | 0 | 0 | 0 | 9 | 33.3 | 27.3 | 30.0 | 8 | 0 | 0 | 0 |
| | 50+ kb (36) | 0 | 0 | 0 | 0 | 3 | 66.7 | 5.6 | 10.3 | 19 | 5.3 | 2.8 | 3.6 |
| Sim2 | 1-3 kb (3) | 17 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 4 | 0 | 0 | 0 |
| | 3-5 kb (10) | 8 | 12.5 | 11.1 | 11.8 | 2 | 50.0 | 11.1 | 18.2 | 8 | 12.5 | 11.1 | 11.8 |
| | 5-10 kb (22) | 7 | 0 | 0 | 0 | 5 | 60.0 | 13.6 | 22.2 | 4 | 0 | 0 | 0 |
| | 10-20 kb (16) | 2 | 0 | 0 | 0 | 6 | 50.0 | 18.8 | 27.3 | 10 | 20.0 | 12.5 | 15.4 |
| | 20-50 kb (45) | 3 | 0 | 0 | 0 | 6 | 50.0 | 6.7 | 11.8 | 14 | 7.1 | 2.2 | 3.4 |
| | 50+ kb (123) | 2 | 1.0 | 1.6 | 3.2 | 3 | 66.7 | 1.6 | 3.2 | 25 | 16.0 | 3.3 | 5.4 |
| Sim3 | 1-3 kb (9) | 19 | 0 | 0 | 0 | 6 | 16.7 | 11.1 | 13.3 | 3 | 0 | 0 | 0 |
| | 3-5 kb (21) | 8 | 12.5 | 4.8 | 6.9 | 4 | 25.0 | 5.0 | 8.3 | 9 | 22.2 | 9.5 | 13.3 |
| | 5-10 kb (45) | 8 | 37.5 | 7.7 | 12.8 | 6 | 16.7 | 2.3 | 4.0 | 16 | 25.0 | 10.3 | 14.5 |
| | 10-20 kb (42) | 6 | 0 | 0 | 0 | 5 | 50.0 | 4.8 | 8.5 | 18 | 5.6 | 2.3 | 3.3 |
| | 20-50 kb (124) | 7 | 14.3 | 0.8 | 1.5 | 11 | 63.6 | 5.7 | 10.5 | 33 | 21.2 | 5.6 | 8.9 |
| | 50+ kb (256) | 10 | 60.0 | 2.0 | 4.5 | 4 | 75.0 | 1.2 | 2.3 | 33 | 24.2 | 3.1 | 5.6 |
| Sim4 | 1-3 kb (21) | 31 | 3.2 | 4.8 | 3.8 | 10 | 10.0 | 5.0 | 6.7 | 10 | 20.0 | 10.0 | 13.3 |
| | 3-5 kb (39) | 9 | 11.1 | 2.6 | 4.3 | 7 | 42.9 | 7.9 | 13.3 | 13 | 7.7 | 2.6 | 3.9 |
| | 5-10 kb (78) | 9 | 11.1 | 1.3 | 2.4 | 8 | 25.0 | 2.6 | 4.7 | 18 | 27.8 | 6.8 | 10.9 |
| | 10-20 kb (48) | 9 | 22.2 | 4.2 | 7.0 | 12 | 16.7 | 4.2 | 6.7 | 18 | 0 | 0 | 0 |
| | 20-50 kb (117) | 10 | 30.0 | 2.6 | 4.8 | 15 | 40.0 | 5.2 | 9.2 | 41 | 4.9 | 1.7 | 2.5 |
| | 50+ kb (340) | 13 | 61.5 | 2.4 | 4.5 | 7 | 85.7 | 1.8 | 3.5 | 47 | 21.3 | 3.0 | 5.3 |
| Sim5 | 1-3 kb (41) | 41 | 7.3 | 7.5 | 7.4 | 15 | 13.3 | 5.0 | 7.3 | 16 | 18.8 | 7.5 | 10.7 |
| | 3-5 kb (51) | 16 | 18.8 | 6.0 | 9.1 | 8 | 50.0 | 7.8 | 13.6 | 13 | 15.4 | 4.0 | 6.3 |
| | 5-10 kb (118) | 13 | 46.2 | 6.1 | 10.7 | 24 | 66.7 | 15.2 | 24.8 | 24 | 45.8 | 11.7 | 18.6 |
| | 10-20 kb (74) | 11 | 36.4 | 5.4 | 9.4 | 9 | 66.7 | 8.1 | 14.5 | 15 | 20.0 | 4.1 | 6.7 |
| | 20-50 kb (156) | 9 | 33.3 | 1.9 | 3.6 | 4 | 50.0 | 1.3 | 2.5 | 38 | 5.3 | 1.3 | 2.1 |
| | 50+ kb (446) | 9 | 66.7 | 1.3 | 2.6 | 6 | 66.9 | 0.9 | 1.8 | 46 | 17.4 | 1.8 | 3.3 |

Table 2. Performance on simulated metagenome datasets stratified by length. The number of gold-standard plasmids for each length bin is indicated in parentheses.

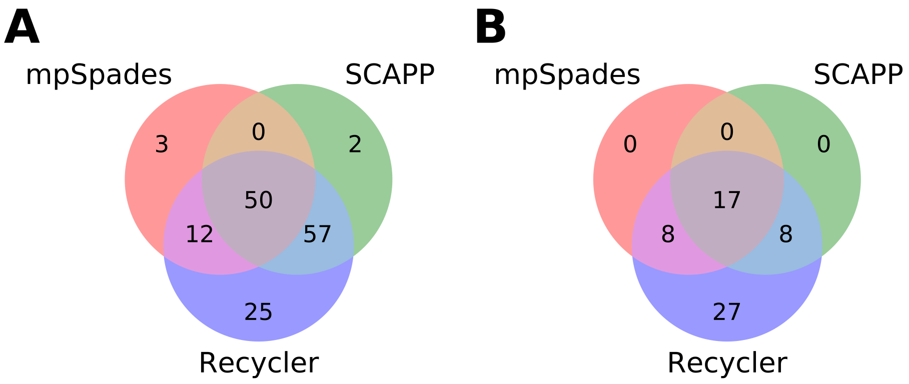

**Fig. S3.** Number of plasmids assembled by each tool on the parallel samples. A: Plasmidome sample. B: Metagenome sample. Discrepancies between the numbers in the diagram and Table 3 are due to cases of overlaps between two plasmids in one tool to one plasmid in another, which were counted as one.

| | Recycler | | mpSpades | | SCAPP | |
|---|---|---|---|---|---|---|
| Sample | # plasmids | median length | # plasmids | median length | # plasmids | median length |
| ERR1297785 | 14 | 2.4 | 4 | 4.7 | 8 | 4.3 |
| ERR1297824 | 15 | 3.2 | 6 | 5.2 | 8 | 4.8 |
| ERR1297720 | 12 | 3.4 | 3 | 4.2 | 4 | 3.4 |
| ERR1297645 | 11 | 3.2 | 7 | 5.2 | 7 | 4.5 |
| ERR1297834 | 5 | 4.4 | 3 | 6.4 | 3 | 6.3 |
| ERR1297838 | 17 | 2.0 | 4 | 4.6 | 8 | 5.4 |
| ERR1297852 | 17 | 5.1 | 5 | 5.3 | 6 | 5.2 |
| ERR1297685 | 18 | 5.2 | 7 | 6.1 | 14 | 4.3 |
| ERR1297738 | 19 | 5.1 | 10 | 4.9 | 11 | 5.1 |
| ERR1297822 | 19 | 4.4 | 5 | 4.5 | 9 | 4.1 |
| ERR1297796 | 11 | 2.9 | 5 | 3.6 | 6 | 2.8 |
| ERR1297700 | 22 | 2.9 | 10 | 4.4 | 18 | 4.4 |
| ERR1297810 | 14 | 3.5 | 8 | 4.4 | 11 | 4.6 |
| ERR1297798 | 11 | 2.9 | 5 | 6.4 | 7 | 2.9 |
| ERR1297671 | 8 | 3.8 | 5 | 4.2 | 6 | 4.0 |
| ERR1297770 | 23 | 3.4 | 10 | 4.7 | 12 | 4.4 |
| ERR1297845 | 20 | 3.3 | 11 | 5.9 | 15 | 3.8 |
| ERR1297751 | 20 | 4.0 | 6 | 5.6 | 16 | 4.4 |
| ERR1297651 | 15 | 3.2 | 4 | 4.9 | 6 | 4.5 |
| ERR1297697 | 25 | 3.8 | 11 | 5.4 | 21 | 4.5 |

Table 3. Number of plasmids and median lengths (in kbp) assembled in each human gut microbiome sample.

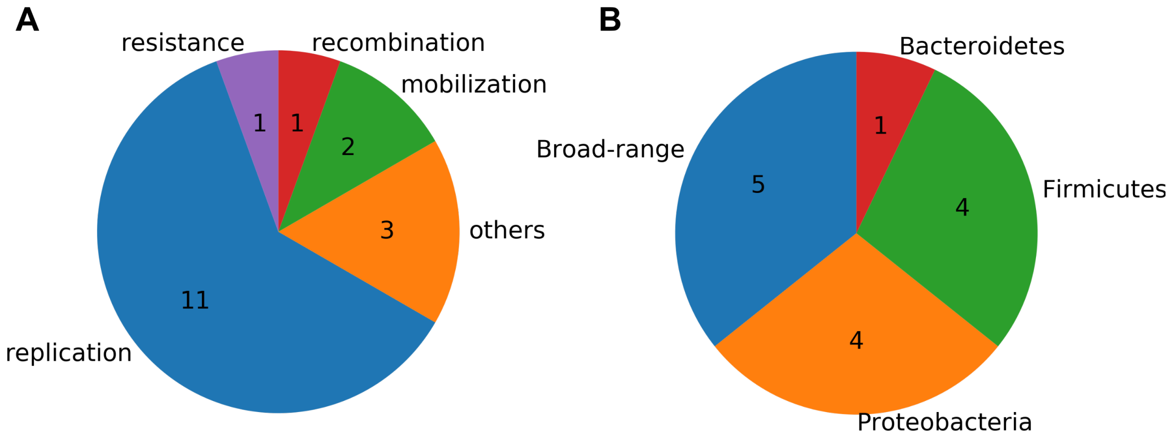

**Fig. S4.** Annotation of genes on the plasmids identified by SCAPP in the rumen plasmidome sample. A: Functional annotations of the plasmid genes. B: Host annotations of the plasmid genes.

## References

Galata, V. *et al.* (2018). PLSDB: a resource of complete bacterial plasmids. *Nucleic acids research*, **47**(D1), D195–D202.

Leplae, R. *et al.* (2009). ACLAME: a classification of mobile genetic elements, update 2010. *Nucleic acids research*, **38**(suppl_1), D57–D61.

Li, H. (2013). Aligning sequence reads, clone sequences and assembly contigs with bwa-mem. *arXiv preprint arXiv:1303.3997*.

Quinlan, A. R. and Hall, I. M. (2010). Bedtools: a flexible suite of utilities for comparing genomic features. *Bioinformatics*, **26**(6), 841–842.

Zhu, W. *et al.* (2010). Ab initio gene identification in metagenomic sequences. *Nucleic acids research*, **38**(12), e132–e132.