교육학석사학위논문

Comparison of metagenomics contig binning methods

메타게놈 염기서열 클러스터링 방법 비교

2019년 2월

서울대학교 대학원

수학교육과

김 수 민

Comparison of metagenomics contig binning methods

메타게놈 염기서열 클러스터링 방법 비교

지도 교수 유 연 주

이 논문을 교육학석사 학위논문으로 제출함
2018년 11월

서울대학교 대학원
수학교육과
김 수 민

김수민의 교육학석사 학위논문을 인준함
2018년 12월

Abstract

With the recent advances in next generation sequencing technologies, shotgun metagenomics, direct sequencing of genetic materials from environmental samples, became available. Shotgun metagenomics enables research of previously under-examined or unknown microbes that cannot be cultured in laboratories. Metagenomics therefore has potential to identify novel genomes from samples and its abundance within samples. Since short-read sequencing outputs a large number of reads in a single run, the task is to construct the genome from which the reads originate. However, since the reads produced by next generation sequencing technologies generally have short lengths, assembling reads into contigs cannot reconstruct the sequence of complete genomes. Aligning reads to the reference genome is also hindered if the reference genome is not available or coverage is insufficient. Moreover, since environmental sample contains various microbial species or strains, different reads obtained from metagenomics shotgun sequencing may originate from different taxa. Therefore, clustering contigs into bins, where each bin corresponds to a species, is needed. After sequencing reads from microbial samples, reads are assembled into contigs, which can then be clustered into species to identify which species reside in the samples. Here we compare eight taxonomy independent contig binning methods that utilizes composition and coverage information to bin contigs into clusters. By comparing their performances across 26 in silico datasets with varying parameters, we suggest a guideline of choosing appropriate methods of binning contigs for various datasets.

Keyword : contig binning methods

Student Number : 2017-27735

Contents

Contents of Figures

Contents of Tables

Chapter 1. Introduction

1.1. Study Background

Microorganisms are single−celled organisms which generally exist in a unit or in the form of clads. Prokaryotes, including bacteria and archaea are the typical examples.

Microorganisms play a vital role in ecological system, and take up approximately one third of the biomass of the Earth [1]. Microorganisms also reside inside human body, forming a community that can have an effect on human's health. Though some microorganisms are benign, others may cause disease. For example, Plasmodium falciparum is a kind of pathogenic protozoa which causes malaria in human. Therefore, it is crucial to understand the genetic structure of microorganisms and their functions.

With the advance in sequencing technology, next generation sequencing emerged, which is cost−effective and efficient in time and performance compared to traditional Sanger sequencing. Illumina platform, one of the next generation sequencer, is widely used due to high accuracy with error rate of 0.1−1% and high outputs with 1.5Tb per run [2]. Next generation sequencing shed light to a new field called metagenomics. Metagenomics, unlike traditional culture-based approaches, refers to the study of genetic materials of microbes obtained directly from nature. This method enables sequencing genetic materials from samples obtained directly from an environment.

There are other related fields such as metataxonomics and meta-transcriptomics that aims to discern genetic compositions of data. Metataxonomics, also sometimes referred to as metagenomics, sequences specific marker genes that are highly conserved such as regions of the ribosomal RNA (rRNA) [3]. The amplicon sequencing of 16S rRNA gene from bacteria and 18S rRNA gene from eukaryotes are widely used, where amplicon indicates the source DNA fragment that are copied and multiplied. 16S rRNA marker genes are suited for phylogenetic profiling because they are present almost in all population, and they have nine hypervariable regions of varying lengths and sequences which can be used to distinguish species [4, 5]. Meta-transcriptomics sequences RNA in a sample, which can reflect the highly transcribed regions of the genes [3].

Although the databases of rRNA such as Greengenes [6], RDP [7], and SILVA [8] has increased significantly, containing genes from millions of species whereas genome databases only contains tens of thousands of species, the pace of new discovery of taxonomy from rRNA amplicon sequencing studies is slowed [3, 9].

A downside of metataxonomics is that 16S rRNA gene may not be in a single copy. Rather, bacteria can have different copy numbers, ranging from 2 to 15 per genome [10]. This can be misleading to the composition inference of samples [11]. Moreover, metataxomomics approach is not applicable to virus, since it does not have universally conserved regions [3]. A PCR bias may also occur since universal primers designed may not recover some rRNA. A study revealed that a minimum of 9.6% of prokaryotic genes are not recovered using PCR-based survey [12]. Metagenomics can be used to resolve these problems.

Shotgun sequencing aims to sequence short fragments, called reads, deriving from a long DNA strand. The long DNA strand is fragmented, and sequenced with chain termination methods repeatedly, creating a set of reads. With the advance of sequencing technology, a large number of reads greater than 10^6 with short reads became available [13]. Illumina HiSeq is an adequate method for sequencing a large number of reads required for in-depth resolution of metagenomics.

To obtain the genetic composition of samples, the reads generated are now either aligned to a reference genome or assembled into contigs to reconstruct the original genomes. Aligning to reference genomes are limited, since reference genomes for novel species are unavailable. In other cases, only 'draft' genomes may be available, which is highly fragmented and incomplete, with the possibility of low quality of these genome sequences [3].

Assembling reads is an alternative method, as it does not require reference genomes. Since shotgun sequencing produces fragments of the original strand repeatedly, there are overlapping regions between reads, which can be used to integrate reads into longer pieces, called contigs.

De Bruijn graph is a popular approach for assembling reads into contigs. The assembler breaks each read into k−mers, where k is a fixed integer and k−mer indicates a sequence of length k. De Bruijn graph is then constructed where each node represents a k−mer and each edge represents an overlap of size k−1 between adjacent nodes. The shortcoming of de Bruijn graph assembler is that when the size of repeats is greater than k, it is unable to determine the original size of the repeats, which leads to the incomplete reconstruction of the whole genome.

Due to sequence repeats, assembly results in a set of fragmented contigs [14]. Therefore, grouping contigs into metagenomic assembled genomes (MAGs), referred to as binning contigs, is needed to determine which contig are derived from which genome.

There are two kinds of binning methods, supervised (taxonomy dependent) method and unsupervised (taxonomy independent) method. Supervised methods search for homology against reference databases for clustering [15]. Unsupervised methods utilize information within the set of contigs to cluster them into species.

The two main features used for unsupervised clustering methods are sequence compositions and coverages [15]. Sequence compositions, also referred to as genome signatures, represent frequencies of all possible sequence of oligonucleotides in each contig. It has been shown that these features are genome−wide and species−specific [16]. Also, genome signatures are not obscured in the course of evolution [17]. Assuming oligonucleotide frequencies are unique for each species, genome signatures is represented by a numeric vector where each element is the frequency of an oligonucleotide [15]. Binners utilize these vectors for clustering contigs into bins.

Coverage information, also called abundance, is also used for unsupervised clustering methods. Coverage of a base position is defined as the number of times the base position is included in the reads. The coverage of a contig is the average of coverages of the bases included in the contig. Coverage profile specifies the coverage of each contig in each sample. The coverage of contigs derived from a single abundant species is likely be high. In this manner, the binning methods assumes that the coverages of contigs derived from the same genome are highly correlated across multiple samples [15].

There are three kinds of binning methods: (1) the sequence composition based methods, (2) the coverage based methods, and (3) the methods based on both sequence composition and coverages of contigs. Methods that uses sequence composition alone are prone to miss some species with low abundance and need long sequences as inputs [15]. Therfore, we chose to compare the performances of the methods based on both sequence composition and coverages of contigs.

1.2. Purpose of Research

Traditional studies on microorganisms were based on culturing steps. However, it is estimated that only 0.1–1.0% of the bacteria present in soil can be cultured under standard conditions, and the fraction of cultivable bacteria in marine samples is even ten to a thousand times lower [1]. A study of amplicon sequencing survey of 16S rRNA also revealed that only a small fraction of bacteria and archaea is discovered through cultivation–based studies [18]. Since culture–based approaches was limited to organisms that can be cultured in a culture medium, most of the microorganisms which cannot be cultured with any growth media have not been well studied.

With the advent of metagenomics, it is now possible to explore the previously unseen microbes. Studies of metagenomics discovered new lineages of viruses [19, 20, 21]. Metagenomics also enables phylogenetic profiling of samples. Time series metagenomics shotgun sequencing can identify the shifts in abundance of bacterial species taking place in premature infant guts [22]. Software such as MIDAS [23], Constrains [45], DESMAN [25], and Lineage [26] are a strain−level resolution algorithms that enables the reconstruction of phylogeny trees from samples.

Moreover, metagenomics can investigate functional roles of genes and their level of expression. Metagenomic studies on bovine digital dermatitis, a disease that causes lesions and limps in cattle, revealed differences in expressed functional genetic composition of healthy, active and inactive lesion stages [27]. Studies of antibiotic resistance genes discovered rivers, wastewater treatment plants, and bacteriophages are reservoirs of antibiotic resistance genes [28, 29, 30].

Metagenomics can also lead to a discovery of novel genes, proteins, enzymes, and chemical compounds that can be used for biotechnology [31]. A study revealed thiopeptide, antibiotics generated by bacteria, is prevalently expressed by microbiota residing in human by examining biosynthetic gene clusters (BGC) of metagenomics samples from Human Microbiome Project [32].

There is a great amount of information held in genomes of unexplored microbes, and metagenomics is one of the major methods that can be used to study these previously non−studied genomes

[31]. Discovering genomic sequence of novel strains or species and its relative abundance will bring valuable information in metagenomics.

With great potential, however, metagenomics faces a challenge of sequencing the whole genome from samples. It is estimated that the number of genomes present in a gram of soil is between 3,000 and 11,000 [31]. This poses a challenge of determining which reads derived from which genomes. Moreover, sequence repeats, low coverage, sequencing errors, and various strains present in each species results in fragmented contigs [14].

This brings the need for clustering the assembled contigs into species. Each contig is placed into a bin (or cluster), and the overall result of binning indicates that contigs belonging to the same bin are inferred to be derived from the same species. Binning contigs can identify the number of species present in the samples, the relative abundance of each species in each sample, and genetic composition of each cluster. The binned contigs can be further analyzed to resolve strains within each species. They can also be used to investigate the potential functions of microorganisms within the microbial community.

In this article, we compare four binning algorithms that utilizes information of both sequence composition and coverage, since supervised methods have advantages over unsupervised methods in the cases when reference genomes are unavailable. These methods use contigs, not reads, as input. The subject methods are : (1) CONCOCT [14], (2) COCACOLA [33], (3) MetaBat [34], (4) MaxBin2 [35], (5) GroopM [36], (6) BMC3C [37], (7) MyCC [38], and (8) GATTACA [39].

In the following chapter, we will discuss in detail methods of generation of data, preprocessing of the data, and eight clustering methods mentioned above. Then we will apply each algorithm to various synthetic data. Since we apply the methods on synthetic data, we can evaluate the performance of the methods by assigning contigs the genome they respectively originate from. We will analyze the performance of each method on five measures.

By applying various datasets on these algorithms, we aim to give a guideline of choosing appropriate method of binning for different datasets.

Chapter 2. Body

2.1. Methods

To evaluate the performance of contig binning methods, we generated 26 in silico datasets, each containing in silico paired−end reads. After data simulation, generated reads for each dataset are assembled to form a set of contigs. Reads from each sample are then aligned to the contigs to generate coverage information. Eight contig binning tools are in turn implemented with input fasta file of contigs and coverage information generated in the previous step. Data simulation, data preprocessing, and methods of contig binning methods are elucidated in the following sections, which are all implemented using AMD Ryzen Threadripper 1950X 16−Core Processor with 125GB ram.

2.1.1. Data simulation

For data simulation, we used StrainMetaSim [25]. The software models the species and strain coverage with normalized log−normal distributions. Each species $\mathrm{t} = 1, 2, \ldots, \mathrm{T}$, has relative abundance $\mathrm{r_{t,s}} = e^{y_{t,s}} / \sum_t e^{y_{t,s}}$, where $\mathrm{y_{t,s}} \sim N(\mu_t, \sigma_t)$ and T is the number of species present. μ_t and σ_t each follows normal and gamma distributions respectively, with $\mu_t \sim N(1, 0.25)$ and $\sigma_t \sim gamma(1, 1)$ [25].

For modeling the abundance of strains within a species, a symmetric Dirichlet distribution $\rho_s \sim Dir(\boldsymbol{a})$ is used, where $\boldsymbol{a}$ is a vector with dimension equal to the number of strains. The Dirichlet distribution ensures that the sum of the abundance of strains equals 1. $\boldsymbol{a}$ is set to a unit vector.

The relative frequency of strain d of species t in sample s is modeled as $k_{d \in t,s} = r_{t,s}\rho_{s,d}$. The strain coverage is then $L_R \times k_{d,s} \times N_s / L_d$, where L_R is the read length, N_s is the number of reads in sample s, and L_d is the length of the genome of strain d [25].

The sequence information of 419 strains from 100 microbial species was used to simulate the datasets, which are provided by StrainMetaSim, and can also be downloaded from NCBI (https://www.ncbi.nlm.nih.gov/). StrainMetaSim randomly selects the given number of strains from each species and computes corresponding relative frequencies and coverage for each strain in each sample according to the distributions set above. The same set of genomes are included in each sample with varying relative frequencies.

ART read simulator [40] is then used to simulate reads for each sample. ART accounts for sequencing errors such as substitutions, insertions, and deletions [40]. Among many platforms that ART supports, Illumina paired end sequencing was selected for read generation.

We generated 26 in silico datasets where the first 13 datasets contain no strain variation and the other 13 datasets contain multiple strain variations for some species.

For each of the first 13 datasets with no strain variation, 100 species are randomly selected one strain per species from the pool of 419 strains. Five out of 13 datasets contain 20 samples each with the number of reads 2,500,000, 5,000,000, 7,500,000, 10,000,000, and 12,500,000 per sample respectively, aiming to evaluate the effect of coverage in the performance of binning methods. Another five datasets contain 60 samples with the number of reads per sample 2,500,000, 5,000,000, 7,500,000, 10,000,000, and 12,500,000 respectively. The remaining three datasets, each with 40, 80, and 100 samples respectively, contains 12,500,000 reads for each sample, which are later used to compare binning performances across varying number of samples.

For the 13 datasets with multiple strain variations, we randomly chose one to five strains from each species, which sums up to 210 strains across all 100 species. Out of 13 datasets, five of them contain 20 samples each, with the number of reads 2,500,000, 5,000,000, 7,500,000, 10,000,000, and 12,500,000 respectively per sample. Another five datasets contain 60 samples with the number of reads per sample 2,500,000, 5,000,000, 7,500,000, 10,000,000, and

12,500,000 respectively. The remaining three contain 40, 80, and 100 samples respectively, each with the number of reads per sample set to 12,500,000.

2.1.2. Data preprocessing

After datasets are generated in silico, aggregated reads across samples are assembled into contigs for each of the 26 datasets. This is done by MEGAHIT version 1.1.3, a time-efficient single node assembler [41]. Although MEGAHIT is an efficient algorithm, assembling contigs takes up considerable time and memory. Thus, we used 32 threads of AMD Ryzen Threadripper 1950X 16-Core Processor. We provide time and peak memory used for each simulated dataset in Table A.1-Table A.26 in Appendix.

After reads are assembled into contigs, long contigs are cut into pieces of 10Kbp to account for chimeric ones. Then contigs are indexed with bwa index [42]. Then bwa mem [42], a read aligner using Burrows Wheeler transform, was used to map the reads in each sample against the assembled contigs, which is used for calculating the coverage.

SAM files generated by bwa mem is then converted to BAM files using samtools view [43], which in turn is sorted using samtools sort [43]. The sorted bam files are used to calculate coverage with a software jgi_summarize_bam_contig_depths included in the MetaBAT package [34]. The software outputs the mean coverage and the variance for each contig in each sample. Then only contigs with minimum length 1,500 are selected for further analysis. By slightly modifying the output, we obtained a coverage matrix Y of size $\mathrm{N} \times \mathrm{S}$ where N is the number of contigs with length $\geq 1{,}500$, S is the number of samples, and Y_{ij} $(\mathrm{i} = 1, 2, \dots, \mathrm{N}, \mathrm{j} = 1, 2, \dots, \mathrm{S})$ is mean coverage of contig i in sample j. y_i, the i^{th} row of Y, is called the coverage vector of contig i.

Lastly, we used fasta_to_features.py included in the CONCOCT packages [14] to calculate composition profile. We used tetramers, which are sequences of length 4, when calculating composition profile. For each contig, the number of tetra−nucleotide frequencies needed to represent each contig is 136. This is because among all possible $4^4 = 256$ tetramers, frequencies of reverse complements are summed except for palindromic tetramers, which are tetramers whose reverse complement is the same as their sequence spelled backwards. After generating composition profile, composition of contigs with length bigger or equal to 1,500 are selected as input for binning. The resulting composition profile is represented by the composition matrix Z of size $\mathrm{N} \times 136$, where N is the number of contigs with length $\geq 1{,}500$, 136 is the number of tetramer frequencies, and Z_{ij} $(\mathrm{i} = 1, 2, \dots, \mathrm{N}, \mathrm{j} = 1, 2, \dots, 136)$ is the j^{th} tetramer frequency of the i^{th} contig. The i^{th} row of Z, denoted z_{i} is called the composition vector of the i^{th} contig.

2.1.3. CONCOCT

After the preprocessing steps, coverage and composition values are normalized. Pseudo−counts are added to coverage vectors. That is, $\mathrm{Y'}_{\mathrm{i,j}} = \mathrm{Y}_{\mathrm{i,j}} + 100/L_i$, where L_i is the length of the i^{th} contig. Coverage vectors are then normalized over contigs and samples. Coverage vectors are first normalized across contigs to account for the differing read counts of each sample,

$$\mathrm{Y''}_{\mathrm{i,j}} = \frac{Y'_{i,j}}{\sum_{b=1}^{N} Y'_{b,j}}.$$

Then it is again normalized across samples

$$\mathrm{P}_{\mathrm{i,j}} = \frac{Y''_{i,j}}{\sum_{b=1}^{S} Y''_{i,b}},$$

which gives the $(\mathrm{i,j})$−entry of the coverage profile. Composition profile is calculated similarly as follows. Z_{i} is normalized to $\mathrm{Z'}_{\mathrm{i}}$where $\mathrm{Z'}_{\mathrm{i,j}} = \mathrm{Z}_{\mathrm{i,j}} + 1$. Then the $(\mathrm{i,j})$−entry of the composition profile is calculated by

$$\mathrm{Q}_{\mathrm{i,j}} = \frac{Z'_{i,j}}{\sum_{k=1}^{136} Z'_{i,k}},$$

which accounts for the heterogeneity in contig lengths.

In addition to calculating coverage and composition profiles, the total coverage of each contig is also calculated as $\mathrm{Y}_{\mathrm{i},\cdot}'' = \sum_{k=1}^{M} Y_{i,k}''$, which may potentially provide additional information in distinguishing species. For each contig, the normalized coverage vector, the normalized composition vector, and the total coverage is concatenated into a log-profile, which is of the form $[\log(Q_{i,1}), \dots, \log(Q_{i,136}), \log(P_{i,1}), \dots, \log(P_{i,S}), \log(Y_{i,\cdot}'')]$ for the i^{th}contig. All of the log-profiles are concatenated row-wise to form a log-profile matrix of size $\mathrm{N} \times (136 + \mathrm{S} + 1)$.

Principal component analysis (PCA) is applied to the log-profile to reduce its high dimensionality. The dimension is reduced to D, maintaining 90% of the variance in the data. The matrix after PCA is denoted by $\mathrm{X}^{(1)}$, which is of size $\mathrm{N} \times \mathrm{D}$. Note that the i^{th} row of $\mathrm{X}^{(1)}$, $\mathrm{x}_{\mathrm{i}}^{(1)}$ $(\mathrm{i} = 1, 2, \dots, \mathrm{N})$, is the feature vector of contig i.

To bin the contigs given $\mathrm{X}^{(1)}$, the number of clusters K is first inferred using automatic relevance determination [44]. Next, Gaussian mixture model is applied to bin contigs. Given K clusters, the data likelihood is defined as

$$\mathrm{L}\left(\mathrm{X}^{(1)} \middle| \mathrm{K}, \pi, \mu, \Sigma^2\right) = \sum_{\mathrm{i}=1}^{\mathrm{N}} \log \sum_{\mathrm{k}=1}^{\mathrm{K}} \pi_{\mathrm{k}} N(\mathrm{x}_{\mathrm{i}}^{(1)} | \mu_{\mathrm{k}}, \Sigma_{\mathrm{k}}^2),$$

where N is a Gaussian distribution, μ_k is the mean vector of component k, Σ_{k}^2 is the variance matrix of component k, and π_k is the mixture proportion of component k. This form of likelihood models each cluster as a shape of an ellipse in a D dimensional space, which is empirically observed by plotting the first two principal components of the log−profile on real data [14]. The prior for the mixture proportions is set as the mixture proportion $\pi = (\pi_1, \pi_2, \dots, \pi_K)$, where π_k is the probability a contig is drawn from cluster k. Prior distributions for μ_k and $\Sigma_{\mathrm{k}}^2, (\mathrm{k} = 1,2,\dots,\mathrm{K})$ are set using Gaussian−Wishart prior modeled as

$$\mathrm{P}(\mu_k, \Delta_k) = \prod_{k=1}^{K} N(\mu_k | m_0, (\beta_0 \Delta_k)^{-1}) W(\Delta_k | v_0, W_0),$$

where Δ_k is the inverse covariance and W is the density function of Wishart distribution given v_0 and W_0. Other parameter values used are set to $m_0 = 0$, $\beta_0 = 0.001$, $\mathrm{v}_0 = D$, and W_0 is set as a diagonal matrix of size D where the d^{th} diagonal element $\mathrm{w}_{\mathrm{dd}}^0 (\mathrm{d} = 1,2,\dots,\mathrm{D})$ of W_0 is $1/(\mathrm{DV_d})$. $\mathrm{V_d}$ is the sample variance of the d^{th} component.

The Bayesian inference is then performed. Starting with a large number of potential components, the sampling model is

$$\mathrm{P}\left(\mathrm{X}^{(1)}\middle|\pi\right) = \int P\left(X^{(1)}, \theta\middle|\pi\right)\ \mathrm{d}\theta,$$

integrating over all the other parameters θ. This integral is estimated using variational Bayesian approximation, solving for maximizing the lower bound of the integral. The optimized mixing coefficients π naturally selects the number of clusters since mixing coefficients of unwanted components approaches zero [44]. The process of CONCOCT is summarized below.

```
The CONCOCT clustering method

Input : The feature profile $\mathrm{X}^{(1)}$
1 : Initialize the number of potential clusters
2 : Conduct variational Bayesian approximation with
      mixture Gaussian model that automatically
      determines the number of clusters
Output : Inferred clusters of contigs
```

2.1.4. COCACOLA

The feature matrix of the coverage and composition profile for COCACOLA is obtained using the normalized coverage matrix P and the normalized composition matrix Q used by CONCOCT. The feature profile $\mathrm{X}^{(2)}$ of COCACOLA is $\mathrm{X}^{(2)} = [\mathrm{P}\ \ Q]$.

The coverage and composition matrix W of size $\mathrm{K} \times (S + 136)$, where K is the inferred number of species and S is the number of samples, is used. The k^{th} row of W, denoted by $\mathrm{w_k}$, indicates the coverage and composition profile of the inferred species k. Let $\mathrm{H_{k,i}}$ be the indicator function such that $\mathrm{H_{k,i}} = 1$ if contig i belongs to species k, and $\mathrm{H_{k,i}} = 0$ otherwise. Then for each $\mathrm{i} = 1,2,\ldots,\mathrm{N}$ the following equation holds.

$$\mathrm{x}_i^{(2)} = \mathrm{H_{1,i}w_1} + \mathrm{H_{2,i}w_2} + \cdots + \mathrm{H_{K,i}w_K}$$

The equation can be represented by a matrix form,

$$(\mathrm{X}^{(2)})^{\mathrm{T}} = W^T \times H,\ \mathrm{W} \geq 0,\ \mathrm{H_{k,i}} = 1 \text{ or } 0,\ \textstyle\sum_{k=1}^{K} H_{k,i} = 1$$

W and H are obtained by minimizing

$$\arg\min_{W,H\geq 0} \| (\mathrm{X}^{(2)})^{\mathrm{T}} - \mathrm{W^T H} \|_{\mathrm{F}}^2, \mathrm{H_{k,i}} = 1 \text{ or } 0,\ \textstyle\sum_{k=1}^{K} H_{k,i} = 1,$$

where $\|\cdot\|_{\mathrm{F}}$ is Frobenius norm. Note that $\mathrm{w_k}$, the k^{th} row of W, indicates the centroid of cluster k. Minimizing is then relaxed to

$$\arg\min_{W,H\geq 0} \| (\mathrm{X}^{(2)})^{\mathrm{T}} - \mathrm{W^T H} \|_{\mathrm{F}}^2, \qquad \mathrm{W, H} \geq 0$$

which allows soft clustering where multiple species can be assigned to each contig. To cope with this, Non−negative Matrix Factorization [45] is used to impose sparsity to facilitate hard clustering and is as follows :

$$\arg\min_{W,H\geq 0} \| (\mathrm{X}^{(2)})^{\mathrm{T}} - \mathrm{W}^{\mathrm{T}}\mathrm{H} \|_{\mathrm{F}}^{2} + \alpha \sum_{i=1}^{N} \| \mathrm{H}_{\cdot,\mathrm{i}} \|_{1}^{2}$$

where α is a parameter for which bigger value indicates stronger sparsity.

Additional information on read linkage of paired end reads is required. If any two contigs are linked by paired end reads a high number of times across multiple samples, it is likely that the contigs are from a single species. The linkage information can be summarized in a network regularization item $\mathrm{R_g}$ [46]. The minimization is now subject to

$$\arg\min_{W,H\geq 0} \| (\mathrm{X}^{(2)})^{\mathrm{T}} - \mathrm{W}^{\mathrm{T}}\mathrm{H} \|_{\mathrm{F}}^{2} + \alpha \sum_{i=1}^{N} \| \mathrm{H}_{\cdot,\mathrm{i}} \|_{1}^{2} + \beta R_{g},$$

where bigger β means the stronger belief in the paired end read linkage information.

COCACOLA opt alternating non−negative least squares (ANLS)

[45] for solving the above equation. The algorithm starts by initializing W and H by k−means clustering with L_1 distance. For each $k = 1,2,\dots,K$, w_k is initialized as the k−means centroid of $X^{(2)}$. H is then initialized to be the indicator matrix for this clustering.

After the initialization, W and H are updated separately while having the other value fixed. The algorithm iterates until a preset criterion is met or the iteration exceeds the maximum iteration number.

After binning contigs by calculating W and H, the post−processing of combining closely mixed clusters follows. The separable conductance $\mathrm{sep}(k_i, k_j)$ between cluster k_i, and cluster k_j $(i = 1,2,\dots,K, j = 1,2,\dots,K, i \neq j)$, is defined by the number of contigs that are included both in the sphere of k_i and that of k_j. The refinement step is done by merging the clusters with the largest separable conductance until the separable conductance falls below a certain threshold, which is set to 1. The process of COCACOLA is summarized below.

The COCACOLA clustering method

```
Input : feature profile $X^{(2)}$
1: Initialize $W$ for which each row is the k-means centroid of $X$
2: Initialize $H$, the indicator matrix of $W$
3. Use ANLS to solve for $W$ and $H$
4. Merge bins using separable conductance
Output : Inferred clusters of contigs
```

2.1.5. MetaBAT

The algorithm of MetaBAT uses tetranucleotide frequency distance probability (TDP) and abundance distance probability (ADP) to probabilistically model distances between contigs. After modeling TDP and ADP, these distance probabilities are combined into one. These pairwise composite distances form a matrix, which in turn is used for binning contigs by modified k-medoid clustering algorithm.

Tetranucleotide frequency distance probability (TDP) models the probability of Euclidean distances between two contigs of different sizes. TDP is approximated by the logistic regression :

$$\mathrm{P}\left(\mathrm{D}_{\mathrm{i,j}}, \mathrm{b}_{\mathrm{i,j}}, c_{i,j}\right) = \frac{1}{1 + e^{-(\mathrm{b}_{\mathrm{i,j}} + c_{i,j} \times D_{i,j})}},$$

where $b_{i,j}$, $c_{i,j}$ are estimated parameters for logistic regression, and $D_{i,j}$ is the distance between contig i and contig j,.

For the base coverage, we use normal distribution as empirically demonstrated by the article. For contig i and contig j, assume base coverage follows $N(\mu_i, \sigma_i^2)$ for contig i and $N(\mu_j, \sigma_j^2)$ for contig j. Then the area not shared by the two normal distributions are computed numerically using cumulative density functions. Then geometric mean of those across samples is obtained, which is defined as ADP of contig i and contig j.

For binning process, k−medoid clustering algorithm [47] is used. First, a seed contig is selected for initial medoid. Then with the calculated cut−off distance, contigs within the cut−off distance are collected. Next we find a new medoid among the union of the seed contig and the collected contigs. Collecting and seeking a new medoid steps are iterated until medoid remains unchanged. Then another contig is selected for the seed of the next cluster. Then the previous steps are iterated in this manner. The algorithm keeps large bins and removes other bins and free the residing contigs. Lastly, if the number of samples is bigger than 10, free contigs are recruited into existing clusters using abundance correlation [36].

The algorithm of MetaBAT is summarized below. Since it iteratively group contigs for each cluster at a time, the number of clusters K need not be predefined.

Note that removing clusters of small sizes may result in a fraction of contigs unbinned. The process of MetaBAT is summarized below.

```
The MetaBAT clustering method

Input : The coverage profile $\mathbf{Y}$ and the composition profile $\mathbf{Z}$
1: Compute ADP and TDP
2: Compute composite distance matrix
3: Conduct k−medoid algorithm
4: If the sample size ≥10, free contigs are recruited into existing
clusters using abundance correlation
Output : Inferred clusters of contigs
```

2.1.6. MaxBin2

MaxBin2 applies expectation maximization (EM) algorithm to bin contigs. It models the probability a sequence i belongs to a species t as

$$\mathrm{P}(\mathrm{i} \in \mathrm{t}) = \mathrm{P}_{dist}(\mathrm{i} \in t) \cdot \prod_{s=1}^{S} \mathrm{P}_{cov}(\mathrm{i} \in \mathrm{t} \mid cov(t_s)),$$

where $\mathrm{P}_{dist}(\mathrm{i} \in t)$ is defined as the probability density function of the tetranucleotide distance between i and t, $P_{cov}(i \in \mathrm{t} \mid cov(t_k))$ is defined as the probability density function of the coverage distance between i and t given $cov(\mathrm{t_s})$, $cov(\mathrm{t_s})$ is the coverage of genome t in sample s $(\mathrm{s} = 1,2, \dots, \mathrm{S})$, and S is the number of samples.

Euclidean distance is used for P_{dist}. That is, the distance between i and t is $dist(\mathrm{i},\mathrm{t}) = \| z_i - \omega_t \|_{L2}$, where z_i and ω_t is the tetramer frequency vector of contig i and genome t respectively. The distance probability function P_{dist}, is the probability that i belongs to t modeled as

$$\mathrm{P}_{dist}(\mathrm{i} \in \mathrm{t}) = \frac{N(dist(\mathrm{i},\mathrm{t})|\mu_{intra},\sigma_{intra}^2)}{N(dist(\mathrm{i},\mathrm{t})|\mu_{intra},\sigma_{intra}^2) + N(dist(\mathrm{i},\mathrm{t})|\mu_{inter},\sigma_{inter}^2)} ,$$

where N is the normal distribution, μ_{intra} and σ_{intra} is the estimated mean and standard deviation of distances within a species, and μ_{inter} and σ_{inter} are the estimated mean and standard deviation of distances between distinct species.

Poisson distribution is used for distribution of coverage, so the coverage probability function P_{cov} is defined as

$$\mathrm{P}_{cov}(\mathrm{i} \in \mathrm{t}|cov(\mathrm{t_s})) = \mathrm{Poisson}(cov(\mathrm{i_s})|cov(\mathrm{t_s})),$$

where $cov(\mathrm{i_s})$ is the coverage of sequence i in sample s, and $cov(\mathrm{t_s})$ is the coverage of genome t in sample s.

Binning contigs is done by EM algorithm. First, the total number of genomes K is estimated as the median number of 107 single copy marker genes [48]. Composition and coverages for each cluster in each sample is also initialized.

Then expectation step follows by calculating the expected probability that the i^{th}contig $(\mathrm{i} = 1, 2, \ldots, \mathrm{N})$ belongs to the k^{th}cluster $(\mathrm{k} = 1, 2, \ldots, \mathrm{K})$. In the maximization step, the composition and coverage for each cluster is recalculated. After EM finishes, each conig is binned into the cluster with the highest probability defined in the expectation step with a threshold probability.

If more than one genomes are assigned to a single cluster, identified by computing the median number of marker genes, the cluster containing multiple genomes are subject again to EM algorithm. The process iterates until each cluster is assigned a single genome. The process of MaxBin2 is summarized below.

```
The MaxBin2 clustering method

Input: The coverage profile $\mathrm{Y}$ and the composition profile $\mathrm{Z}$
1: Estimate the number of clusters $\mathrm{K}$
2: Initialize the coverage and composition for each cluster
3: Perform EM for each cluster to bin contigs
4: If a cluster including multiple species are detected, it is subject again to EM algorithm until no such bin is detected.
Output : Inferred clusters of contigs
```

2.1.7. GroopM

GroopM is processed in five stages: parse, core, refine, recruit, and extract. GroopM uses coverage, composition, and contig length information for binning contigs. Note that in this study, we consecutively follow parse, core, and recruit stages. The following is the description of the parse and the core stages, which are main stages of GroopM.

In the first stage, parse, the coverage and composition profiles are calculated. For the coverage profile, truncated mean coverage (TMC) for each contig in each sample is calculated. TMC for each of the contig in each sample is defined as the expectation of the base position depths that fall within one standard deviation from the mean coverage.

For the composition profile, the tetranucleotide frequency vector is calculated. Then the dimension of the vector is reduced using PCA, maintaining 80% of the variance in the data.

Empirically, the TMC vectors of all contigs are plotted onto the TMC space, where each axis indicates each sample and each coordinate along each axis indicates the value of coverage in the corresponding sample. It is observed that the positioned contigs formed spears. Therefore, TMC profile and TMC space are transformed so that positioned contigs formed spherical clusters rather than spears with the size of the dimension reduced to three.

Next, the core stage aims to construct preliminary bins. It consists mainly of two−way clustering, Hough partitioning, and recruiting.

In two−way clustering, coverage space and composition space are considered separately. The process iteratively selects a contig in the densest part of either space, and contigs closely neighboring to the selected contig in the other space is moved closer together. Then Hough transform [49] is applied to partition contigs into bins.

Then unbinned contigs are clustered by computing the cutoff regieons for each preliminary bin. Then the nearest neighbor algorithm is performed to merge clusters. Lastly, a self−organizing map (SOM) [50] is used to refine bins, which reassigns contigs to bins in such a way that each area masked by a SOM has a one to one correspondence to each bin. The process of GroopM parse and core steps is summarized below.

```
The GroopM parse and core method

Input : The set of N contigs, the sorted bam files
1: The parse stage
    Compute TMC profile and PCA-applied composition profile
2: The core stage
    2.1. Perform two-way clustering
    2.2. Perform Hough partitioning and output preliminary bins
    2.3. Recruit contigs not clustered
    2.4. Perform nearest neighbor method to recruit and merge
    2.5. Refine clusters using a self-organizing map
Output : Preliminary binned contigs
```

2.1.8. BMC3C

In addition to the coverage and composition profiles, BMC3C utilizes the gene codon usage information to bin contigs. BMC3C employs ensemble k-means algorithm to increase the robustness of the result.

There are 64 codons, 61 of which encode amino acids and 3 of which encode the stop codons. There are 20 amino acids. Since each codon encodes an amino acid, this implies that more than one codons encode the same amino acid for some amino acids. This redundancy in codons is called genetic degeneracy.

Given a protein coding DNA and a specific amino acid in that DNA, it was observed that there is a bias in the frequency of codon usage in encoding the amino acid. That is, among the codons that encode the same amino acid, some codons are used more to encode the amino acid than do the others.

BMC3C thus uses information of the gene codon usage for clustering. Five independent statistics [51] that measure the value of gene codon usage were calculated. For a particular gene, define $\mathrm{TC2}$ as the set of degenerate sites with two candidates T or C where degenerate sites are particular bases in the gene where multiple varieties of nucleotides can be positioned not violating the kind of amino acid it encodes. Define $\mathrm{C2}$ as the set of degenerate sites with base C in $\mathrm{TC2}$. Likewise, define $\mathrm{AG2}$ as the set of degenerate sites with two candidates A or G and $\mathrm{G2}$ as the set of degenerate sites with G in $\mathrm{AG2}$. $\mathrm{N4}$ is defined as the set of degenerate sites with all four possibilities A, C, G, or T. $\mathrm{C4}, \mathrm{G4}$, and $\mathrm{A4}$ are defined as the set of degenerate sites with C, G, and A in $\mathrm{N4}$ respectively. The five statistics are given as follows :

$$\mathrm{PC2} = \frac{|\mathrm{C2}|}{|\mathrm{TC2}|}, \mathrm{PG2} = \frac{|\mathrm{G2}|}{|\mathrm{AG2}|}, \mathrm{PC4} = \frac{|\mathrm{C4}|}{|\mathrm{N4}|}, \mathrm{PG4} = \frac{|\mathrm{G4}|}{|\mathrm{N4}|}, \mathrm{PA4} = \frac{|\mathrm{A4}|}{|\mathrm{N4}|}.$$

$\mathrm{PC2}$ is the proportion of C in $\mathrm{TC2}$, $\mathrm{PG2}$ is the proportion of G in $\mathrm{AG2}$, $\mathrm{PC4}$ is the proportion of C in $\mathrm{N4}$, $\mathrm{PG4}$ is the proportion of G in

$\mathrm{N4}$, and $\mathrm{PA4}$ is the proportion of A in $\mathrm{N4}$.

These statistics are weighted averaged over all genes in each contig, where weights are given proportional to the length of the gene in the contig. The codon usage vector $\mathrm{u_i}$ $(\mathrm{i} = 1,2,\dots,N)$ of contig i can be expressed as follows :

$$\mathrm{u_i} = \frac{\sum_{\mathrm{g=1}}^{\mathrm{n_g^i}} L_{\mathrm{g}} \zeta_{\mathrm{g}}^{\mathrm{i}}}{\sum_{\mathrm{g=1}}^{\mathrm{n_g^i}} L_{\mathrm{g}}},$$

where n_g^i is the number of genes in contig i, L_g is the length of gene g $(\mathrm{g} = 1,2,\dots,n_g^i)$ in contig i, and $\zeta_g^i = [PC2_g, PG2_g, PC4_g, PG4_g, PA4_g]$ where each element is the gene codon usage measure of gene g. Each codon vector are concatenated row-wise to form the codon usage matrix U of size $\mathrm{N} \times 5$.

Composition matrix Q and coverage matrix P defined previously are used. The feature matrix of BMC3C is $\mathrm{X}^{(6)} = [\mathrm{Q} \;\; \mathrm{P} \;\; \mathrm{U}]$ of size $\mathrm{N} \times (136 + \mathrm{S} + 5)$, where N is the number of contigs and S is the number of samples.

BMC3C proceeds as follows. First, the number of clusters K^* is inferred. K-means [52] is performed and the resulting clusters are iteratively merged using separable conductance $\mathbf{sep}(\mathrm{k_i}, k_j)$ between cluster $\mathrm{k_i}$ and cluster $\mathrm{k_j}$ until the preset threshold is met. The remaining number of clusters is set to K^*.

Next, the k−means clustering follows with initial number of clusters set as $K = 10 \times K^*$ with $K < N$. The k−means algorithm aims to group the N contigs into K clusters by solving

$$\arg\min \sum_{k=1}^{K} \sum_{x_i \in F_k} \| x_i^{(6)} - w_k^{(6)} \|^2$$

where $x_i^{(6)}$ is the feature vector of contig i, $w_k^{(6)}$ is the centroid of cluster k, and F_k is the set of feature vectors of cluster k. After k−means algorithm is performed, define an indicator matrix M of size $N \times K$, where $M_{i,j} = 1$ if contig i is in cluster j, and $M_{i,j} = 0$ otherwise.

K−means algorithm has two downsides. One is that empirical observation tells different clusters are highly close together, making a single run of k−means unreliable. Also, the initial centroids affect the result of the clustering significantly. Therefore, in the next step, ensemble k−means is applied. That is, clustering with k−means is performed m times with K fixed. The co−association matrix $M' \in R^{K \times K}$ is then defined as

$$M' = \frac{\sum_{r=1}^{m} M_r M_r^T}{m}$$

where M_r is the indicator matrix of size $N \times K$ obtained from the r^{th} run of k−means. Note that $\left(M_r M_r^T\right)_{i,j}$ is 1 if the contig i and the contig j both comes from the same cluster and 0 otherwise. Therefore M' is a weighted adjacency matrix of a graph where each node corresponds to each contig.

Finally, the weighted graph is partitioned into subgraphs, where each subgraph corresponds to a cluster. This is done by normalized cut (Ncut) [53, 54]. The process of BMC3C is summarized below.

```
The BMC3C clustering method

Input : The feature matrix $[Q\ \ P\ \ B]$
1: Infer the number of potential clusters $K^*$
2: Perform k−means $m$ times with fixed $K = 10 \times K^*$
3: Calculate co−association matrix $M'$
4: Partition graph with adjacency matrix $M'$ using NCut
Output : Inferred clusters of contigs
```

2.1.9. MyCC

MyCC uses the marker gene information besides the coverage and composition profiles. MyCC uses the marker gene information when correcting the clustered bins.

The procedure of MyCC is as follows. MyCC first searches for the protein coding genes in the contigs by Prodigal [55]. Then FetchMG [56, 57] is performed to identify the 40 single copy marker genes of prokaryotes in the contigs. UCLUST [58] is also used to find species−level marker genes.

Next, the composition profile is calculated for each contig. In addition to the tetranucleotide frequency, pentanucleotide frequency and hexanucleotide frequency of palindromic sequences are also calculated. Pseudo counts are added to the computed composition profile, which is then normalized and centered log−ratio (CLR) [59,60] transformed. Coverage profile can also be added optionally before CLR.

The feature profile is then plotted on a two−dimensional space with Barnes−Hut−SNE [61]. Then the contigs are clustered using affinity propagation (AP) [62]. AP is performed based on the measure of similarity between two sequences defined as negative squared Eucledian distance between the two. AP is performed in two steps. First, the clustering is performed for long contigs. Then short contigs are added to the clusters generated.

The clusters created are corrected using the marker genes previously found. The clusters containing duplicates of a marker gene is split using spectral clustering [63]. The clusters close together are merged if the marker genes contained in a cluster are complementary to that of the other cluster. The process of MyCC is summarized below.

```
The MyCC clustering method
Input : The marker genes found, the feature profile
1: Normalize the feature profile by CLR transformation
2: Reduce the dimensionality of the feature profile by
    Farnes-Hut-SNE
3: Perform AP to cluster contigs
4: Correct clusters using the marker genes previously identified
Output : Inferred clusters of contigs
```

2.1.10. GATTACA

GATTACA is a memory and time efficient method that uses kmer indexing to estimate the coverage profile. GATTACA software includes commands for indexing k-mers, clustering contigs, and sample comparing.

To estimate coverage profile, a minimal perfect hash function (MPHF) [64] is used to index the k-mer counts for each contig in each sample with a small space of storage. Given a set of N contigs, for each sample, the set of all k-mers from the sample are identified excluding those with the count one to account for the possibility of sequencing error. Then each k-mer is indexed with a MPHF h where each k-mer is mapped to a single integer ranging consecutively from 0 to n-1. Then each index is mapped to the k-mer count it corresponds to. A Bloom filter is additionally used to store all k−mers to map the index backwards to the corresponding k−mer.

Then for each contig in each sample, the median counts of k−mers in the contig is calculated, which is used for the coverage profile. GATTACA also uses composition profile obtained by the normalizing method mentioned in section 2.1.3.

Similar to CONCOCT, GATTACA also uses the a Gaussian mixture model to construct the likelihood function. With a Dirichlet prior for Bayesian inference, GATTACA opt the variational Bayes Expectation Maximization (VBEM) to maximize the marginal log−likelihood. The process of GATTACA is summarized below.

```
The GATTACA clustering algorithm
Input : The coverage generated by k−mer counting,
        the normalized composition profile P
1: Estimate the coverage of each contig in each sample
   using indexing of kmer counts
2: Generate the feature profile using the coverage generated
   in the previous step and the composition profile given
3: Cluster contigs using Variational Bayes EM method for Gaussian
   mixture model
Output : Inferred clusters of contigs
```

2.1.11. Evaluation measures

Evaluation measures used to measure the performance of the above five methods are recall, precision, normalized mutual information, Rand index (RI), and adjusted Rand index (ARI), which are calculated by the script in the CONCOCT package [14]. Since we use simulated datasets, we can identify which genome a contig is derived from by finding the genome that the majority of the reads mapped to the contig is derived from. By assigning each contig to the genome from which it originates, we can construct a matrix A of size $\mathrm{K} \times \mathrm{T}$ where K is the number of clusters, T is the number of genomes, and the $(\mathrm{i}, \mathrm{j})-$entry of A, $n_{\mathrm{i,j}}$, is the number of contigs of the cluster i and the assigned genome class j.

From A, we measure how far the clustering is from the grouping by genomes. The recall is the proportion of contigs grouped adequately based on each genome class. For each genome class t $(\mathrm{t} = 1,2,\dots,\mathrm{T})$, a cluster $\mathrm{k_t}$ $(\mathrm{k_t} \in \{1,2,\dots,\mathrm{K}\})$, which has the maximum number of contigs from the genome class s, is identified. That is, $\mathrm{k_t} = \arg\max_{\mathrm{i} \in \{1,2,\dots,\mathrm{K}\}} n_{\mathrm{i,t}}$, which indicates the corresponding cluster of the genome class t is the cluster $\mathrm{k_t}$. The recall is then calculated by

$$\mathrm{Recall} = \frac{\sum_{\mathrm{t}=1}^{\mathrm{T}} \max_{\mathrm{i} \in \{1,2,\dots,\mathrm{K}\}} n_{i,t}}{\mathrm{N}} = \frac{\sum_{\mathrm{t}=1}^{\mathrm{T}} n_{\mathrm{k_t,t}}}{\mathrm{N}}$$

which gives the fraction of the contigs correctly assigned to the cluster inferred from each of the genome class.

The precision identifies for cluster $\mathrm{k}, (\mathrm{k} = 1,2,\ldots,\mathrm{K})$ a genome class $\mathrm{t_k}, (\mathrm{t_k} \in \{1,2,\ldots,\mathrm{T}\})$ which has the maximum number of contigs from cluster k. That is, $\mathrm{s_t} = \arg \max_{j\in\{1,2,\ldots,\mathrm{T}\}} n_{k,j}$, assuming that $\mathrm{t_k}$ is the genome class that cluster k corresponds to. Precision is then calculated by

$$\text{Precision} = \frac{\sum_{\mathrm{k}=1}^{\mathrm{K}} \max_{\mathrm{j}\in\{1,2,\ldots,\mathrm{S}\}} n_{\mathrm{k,j}}}{\mathrm{N}} = \frac{\sum_{\mathrm{k}=1}^{\mathrm{K}} n_{\mathrm{k,s_k}}}{\mathrm{N}},$$

which gives the fraction of the contigs correctly assigned to the genome class for each cluster.

The normalized mutual information, NMI, between the set of all clusters K and the set of all genome classes T is defined as

$$\mathrm{NMI(K,G)} = \frac{2 * \mathrm{MI(K,T)}}{\mathrm{H(K)} + \mathrm{H(T)}},$$

where $\mathrm{MI(K,T)}$ is the mutual information of K and T, $\mathrm{H(K)}$ is the entropy of K, and $\mathrm{H(T)}$ is the entropy of T.

The rand index, RI is the proportion all the correctly identified pairs of contigs out of all pairs of the contigs. That is,

$$\mathrm{RI} = \frac{n(TN) + n(TP)}{n(TN) + n(TP) + n(FN) + n(FP)} = \frac{n(TN) + n(TP)}{\binom{\mathrm{N}}{2}}.$$

The number of true positive cases, which is the number of cases a pair of contigs from the same cluster originate from the same genome class, denoted $n(\mathrm{TP})$, is

$$n(\mathrm{TP}) = \sum_{\mathrm{i}} \sum_{\mathrm{j}} \binom{n_{\mathrm{i,j}}}{2}.$$

If a pair of contigs from the same bin in fact originate from two different genome classes, it is called a false positive case. The number of false positive cases, denoted $n(FP)$ is

$$n(FP) = \sum_{\mathrm{i}} \binom{n_{\mathrm{i,\cdot}}}{2} - \sum_{\mathrm{i}} \sum_{\mathrm{j}} \binom{n_{\mathrm{i,j}}}{2}$$

where $n_{i,\cdot} = \sum_j n_{i,j}$.

If a pair of the contigs from two different clusters originate from a single genome class, it is called a false negative case (FN). The number of FN cases, denoted $n(\mathrm{FN})$, is

$$n(\mathrm{FN}) = \sum_{\mathrm{j}} \binom{n_{\cdot,j}}{2} - \sum_{\mathrm{i}} \sum_{\mathrm{j}} \binom{n_{i,j}}{2},$$

where $n_{\cdot,\mathrm{j}} = \sum_{\mathrm{i}} n_{\mathrm{i,j}}$.

If a pair of contigs from different clusters originate from different genome classes, it is called a true negative (TN) case. The number of TN cases, denoted $n(\mathrm{TN})$ is

$$n(\mathrm{TN}) = \binom{\mathrm{N}}{2} + \sum_{\mathrm{i}} \sum_{\mathrm{j}} \binom{n_{\mathrm{i,j}}}{2} - \sum_{\mathrm{j}} \binom{n_{\cdot,\mathrm{j}}}{2} - \sum_{\mathrm{i}} \binom{n_{\mathrm{i},\cdot}}{2}$$

The adjusted Rand index, ARI, accounts for the random chance of RI getting a nonzero value.

$$\mathrm{ARI} = \frac{\sum_{\mathrm{i}} \sum_{\mathrm{j}} \binom{n_{\mathrm{i,j}}}{2} - \frac{\sum_{\mathrm{i}} \binom{n_{\mathrm{i},\cdot}}{2} \sum_{\mathrm{j}} \binom{n_{\cdot,\mathrm{j}}}{2}}{\binom{\mathrm{N}}{2}}}{\frac{1}{2}\left\{\sum_{\mathrm{i}} \binom{n_{\mathrm{i},\cdot}}{2} + \sum_{\mathrm{j}} \binom{n_{\cdot,\mathrm{j}}}{2}\right\} - \frac{\sum_{\mathrm{i}} \binom{n_{\mathrm{i},\cdot}}{2} \sum_{\mathrm{j}} \binom{n_{\cdot,\mathrm{j}}}{2}}{\binom{\mathrm{N}}{2}}}$$

Contig lengths are taken into account so that the long contigs assigned correctly has more weights than the shorter contigs. This is done by setting each contig as a number of replicated data points where the number of replicates equals the length of the contig considered [14].

2.2. Results

We ran the eight algorithms on the 13 in silico datasets consisting of 100 species with no strain variation. We used the CONCOCT software version 0.4.2, the python−version of COCACOLA, the MetaBAT2 software version 2.12.1, the MaxBin software version 2.2.4, the GroopM version 0.3.4, the BMC3C software downloaded November 10, 2018, the MyCC software last modified in March 1, 2017, and the GATTACA software updated November 30, 2017.

For each method, we used one thread of the CPU to compare the elapsed time and the peak memory usage across algorithms. The elapsed time will be shortened considerably if multiple threads are used. We timed commands concoct for CONCOCT, cocacola.py for COCACOLA, metabat2 for MetaBAT, run_MaxBin.pl for MaxBin2.0, MyCC.py for MyCC, and gattaca.py cluster for GATTACA.

For GroopM, we summed up the time spent on groopm parse, core, recruit, and extract. For BMC3C, the time spent on prodigal, codon_usage.py, and bmc3c.m are summed up.

Note that the value of recall and precision are strongly affected by the number of clusters inferred. The recall value tends to be low when the number of clusters is low relative to the number of species, and the precision value tends to be high when the number of clusters is high. Therefore, when comparing the performance of different tools for binning, we mainly focus on the value of ARI, which combines both the recall and precision.

2.2.1. Results of the data with no strain variation

Species_20S_2.5R contains 20 samples each with the number of reads 2,500,000. The time spent on the assembling into contigs using MEGAHIT took 49 minutes and 30 seconds and the peak memory usage was 6.3GB. The number of contigs after cutting up the long contigs with the threshold 10,000bp is 131,360. Then the contig binning is done for only the contigs with the minimum contig length 1,500Kbp and the count is 42,054.

The elapsed time, the peak memory usage, the number of contigs binned for each algorithm, the numbers of the inferred clusters, and the performance scores are summarized in Table A.1 in Appendix. The number of clusters inferred by MaxBin2, 93, as shown in Table A.1, was the closest to the true value 100 whereas the number of clusters inferred by BMC3C is the farmost with 66.

Figure 1 shows the percentage of the base pairs assigned against each of the performance measure. The base pair assigned indicates the proportion of base pairs assigned into bins out of all base pairs in the set of all contigs.

As shown in Figure 1, the proportion of the assigned base pairs is between 0.8 and 0.9. This is because we only used contigs with minimum length 1,500bp as input.

We can see that the performance of CONCOCT is the highest across all the performance metrics except for the recall value. Especially, the ARI value of CONCOCT is clearly the highest. The performances of Maxbin2 and MyCC are considerably high. The performances of the rest of the methods are relatively low, where GroopM has the lowest overall performance.

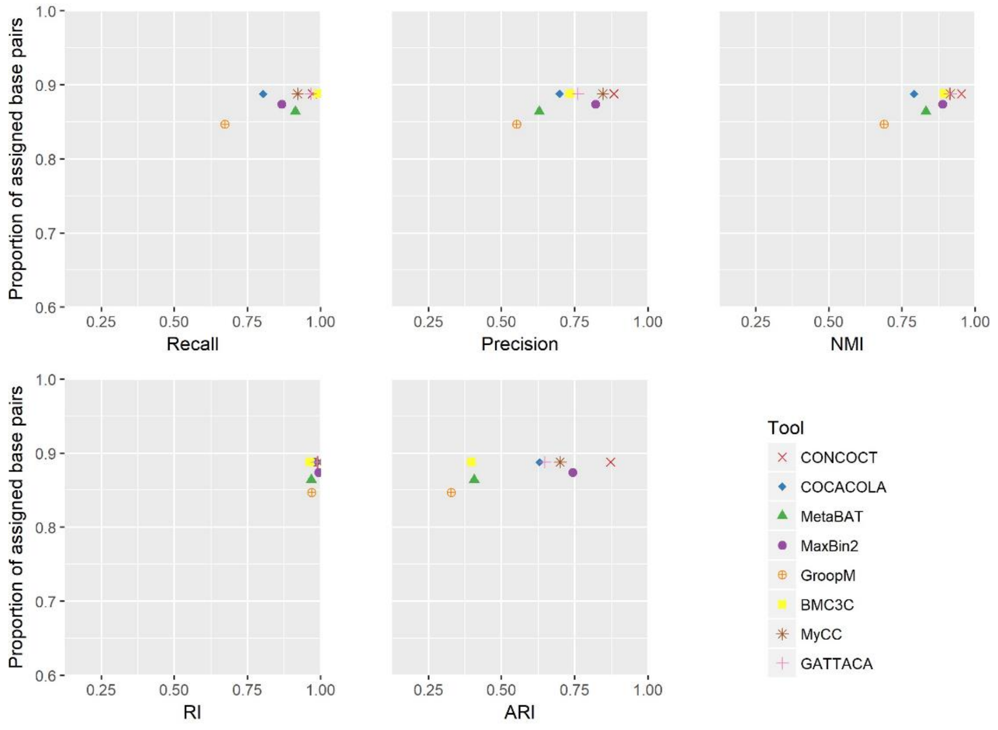


Figure 1. Performance scores for Species_20S_2.5R

We can see that the performance of CONCOCT is the highest across all the performance metrics except for the recall value. Especially, the ARI value of CONCOCT is clearly the highest. The performances of Maxbin2 and MyCC are considerably high. The performances of the rest of the methods are relatively low, where

GroopM has the lowest overall performance.

Species_20S_5.0R The second dataset consists of 20 samples each with the number of reads 5,000,000. The time spent on MEGAHIT is 3 hours and 37 minutes and the peak memory usage was 12.5GB. The number of the assembled contigs after cutting up the long contigs is 52,888, and 37,936 contigs with the length bigger than or equal to 1,500bp are used for each of the clustering method.

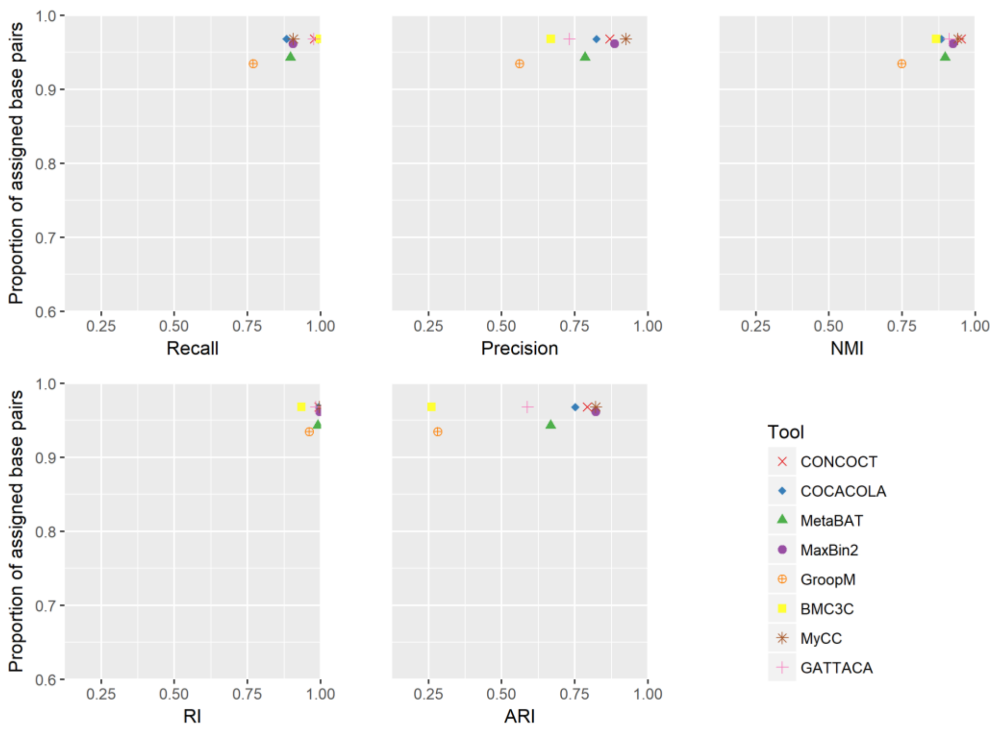


Figure 2. Performance scores for Species_20S_5.0R

As shown in Figure 2, the proportion of the assigned base pairs is between 0.9 and 1. The performances of MaxBin2, MyCC, and CONCOCT are relatively high, especially in precision, NMI, RI, and ARI. The performances of COCACOLA, MetaBAT and GATTACA follows. The performances of BMC3C and GroopM are relatively low, though the recall value for BMC3C is the highest.

Species_20S_7.5R The third dataset consists of 20 samples each with the number of reads 7,500,000. The time spent on MEGAHIT is 1 hour 51 minutes and the peak memory usage is 18.7GB. The number of the assembled contigs after cutting up the long contigs is 39,748 and 33,788 contigs with the length bigger than equal to

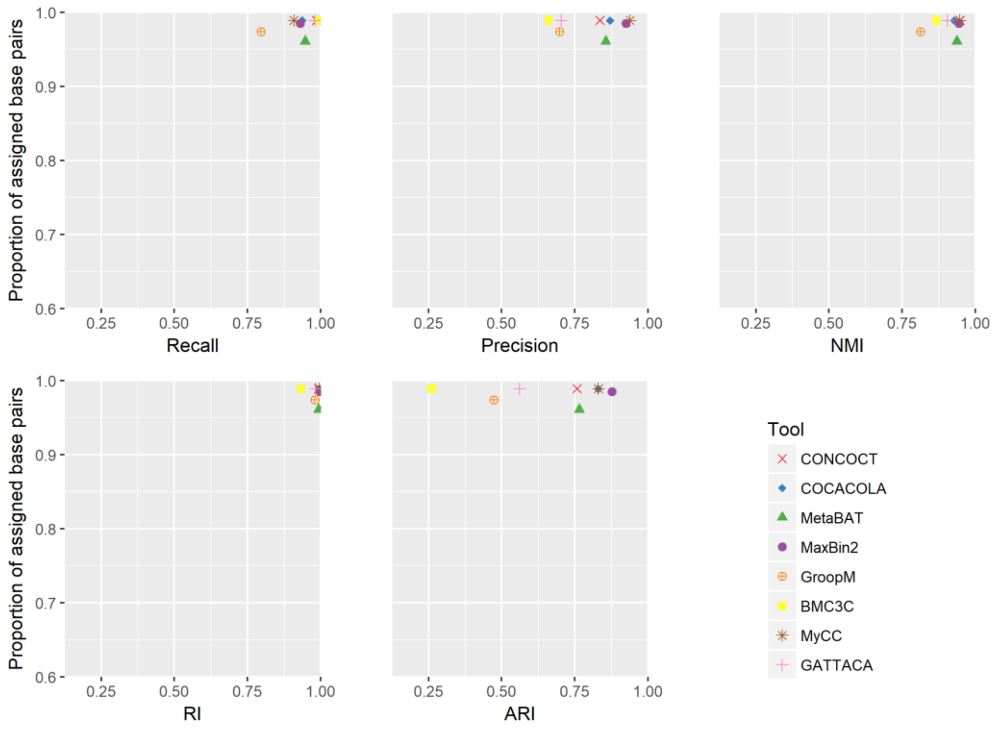


Figure 3. Performance scores for Species_20S_7.5R

1,500bp are used for each of the clustering method.

As shown in Figure 2, the proportion of the assigned base pairs is between 0.95 and 1. MaxBin2, MyCC, COCACOLA, MetaBAT, and CONCOCT performed relatively well with the ARI values bigger than

0.75, the NMI values bigger than 0.92, and the precision values bigger than 0.83. The performances of GATTACA, GroopM, and BMC3C are relatively low.

Species_20S_10.0R The fourth dataset consists of 20 samples each with the number of reads 10,000,000. The time spent on MEGAHIT is 2 hours and 28 minutes and the peak memory usage is 25.0GB. The number of the assembled contigs after cutting up the

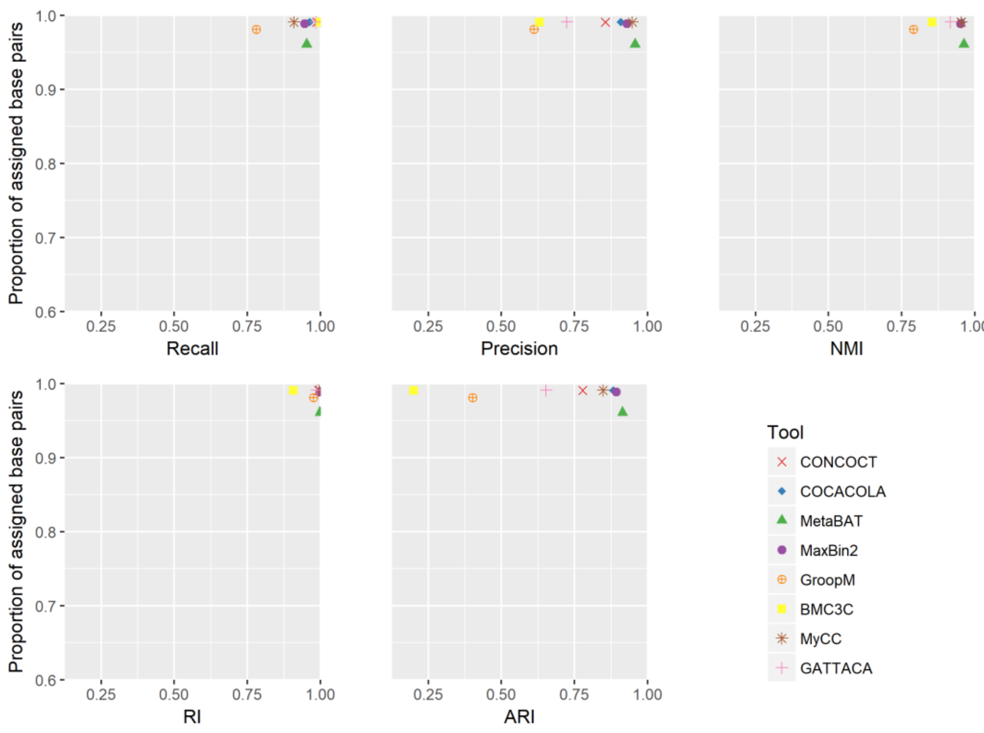


Figure 4. Performance scores for Species_20S_10.0R

long contigs is 37,939 and 32,254 contigs with the length bigger than or equal to 1,500bp are used for each of the clustering method.

As shown in Figure 4, the proportion of the assigned base pairs is between 0.95 and 1. MetaBAT outperformed other tools across all metrics except for the recall value. MaxBin2, COCACOLA, MyCC, and CONCOCT performed relatively well. The performances of GATTACA, GroopM, and BMC3C are relatively low.

The number of inferred clusters closest to 100 is 103 for both MetaBAT and MaxBin2, whereas the number of inferred clusters farthest to 100 is 63 for BMC3C.

Species_20S_12.5R The fifth dataset consists of 20 samples each with the number of reads 12,500,000. The time spent on MEGAHIT is 3 hours and 7 minutes and the peak memory used is 31.2GB. The number of the assembled contigs after cutting up the long contigs is 38,714 and 32,417 contigs with the length bigger than or equal to 1,500bp are used for each of the clustering method.

As shown in Figure 5, the proportion of the assigned base pairs is between 0.95 and 1. Similar with the fourth dataset, MetaBAT performs the best in all but the recall. MaxBin2, COCACOLA, and MyCC, also performed well. The performance of CONCOCT ranks next. The performances of GroopM, BMC3C, and GATTACA were relatively low.

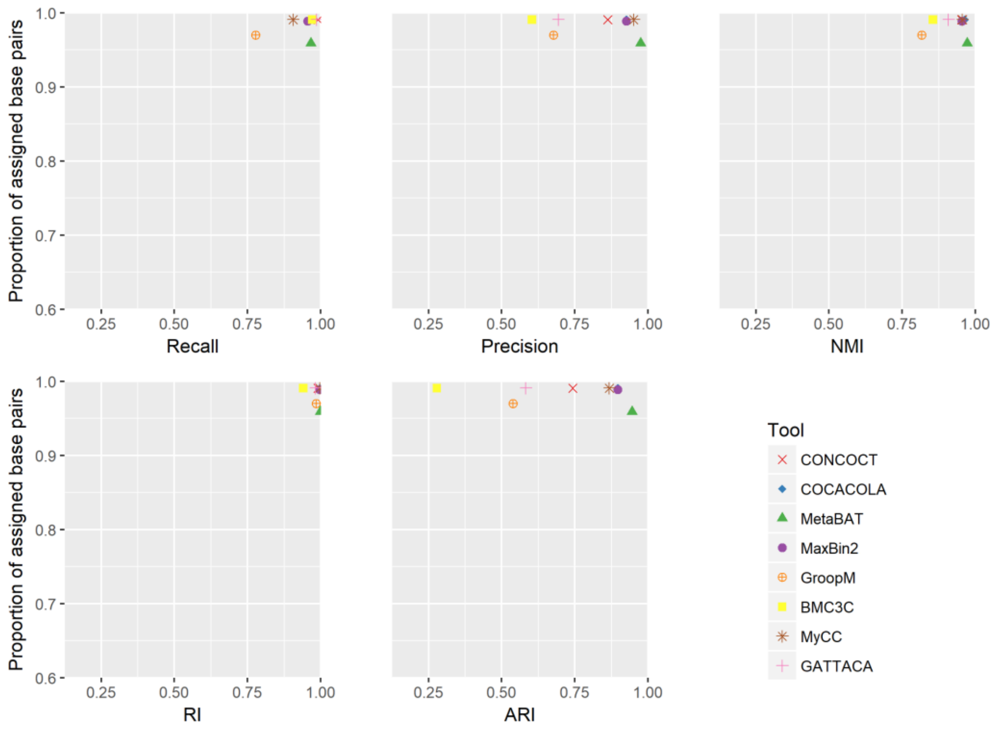


Figure 5. Performance scores for Species_20S_12.5R

Species_40S_12.5R The sixth dataset consists of 40 samples each with the number of reads 12,500,000. The time spent on MEGAHIT is 5 hours 43 minutes and the peak memory usage is 62.3GB. The number of the assembled contigs after cutting up the long contigs is 42,646 and 32,530 contigs with the length bigger than or equal to 1,500bp are used for each of the clustering method.

As shown in Figure 6, the proportion of the assigned base pairs is between 0.95 and 1. COCACOLA, MetaBAT, MaxBin2, and MyCC performed the best, with the precision value, the NMI value, and the RI value for each of the four method bigger than 0.94. The performances of CONCOCT and BMC3C rank next. The performances of GroopM and GATTACA are the lowest.

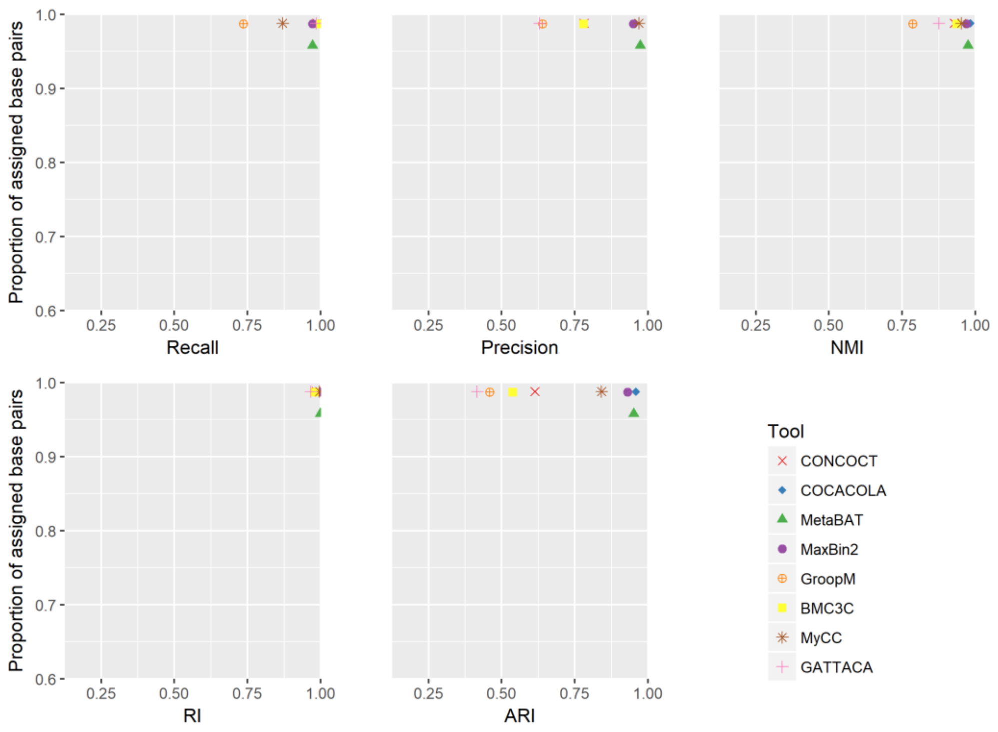


Figure 6. Performance scores for Species_40S_12.5R

Species_60S_2.5R The seventh dataset consists of 60 samples each with the number of reads 2,500,000. The time spent on MEGAHIT is 2 hours and 56 seconds and the peak memory used is 18.8GB. The number of the assembled contigs after cutting up the long contigs is 38,335 and 32,793 contigs with the length bigger than or equal to 1,500bp are used for each of the clustering method.

As shown in Figure 7, the proportion of the assigned base pairs is between 0.9 and 1. The proportion of the assigned base pairs for MetaBAT is the lowest. MyCC outperformed the other tools in all measures but the recall. The performance of MaxBin2 ranked next, taking second places in all but the recall value. The performance of COCACOLA took the third places in all but the recall value. The performances of MetaBAT, CONCOCT, GroopM, BMC3C, GATTACA are low.

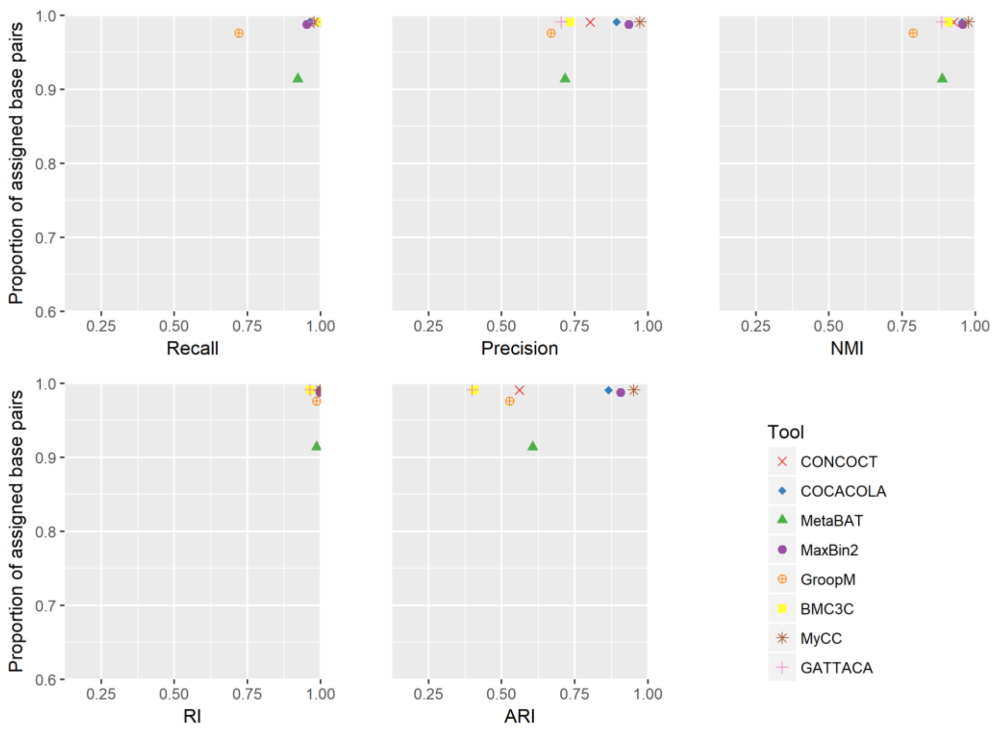


Figure 7. Performance scores for Species_60S_2.5R

Species_60S_5.0R The eighth dataset consists of 60 samples each with the number of reads 5,000,000. The time spent on MEGAHIT is 3 hours and 44 minutes and the peak memory used is 37.5GB. The number of the assembled contigs after cutting up the long contigs is 39,819 and 32,503 contigs with the length bigger than or equal to 1,500bp are used for each of the clustering method.

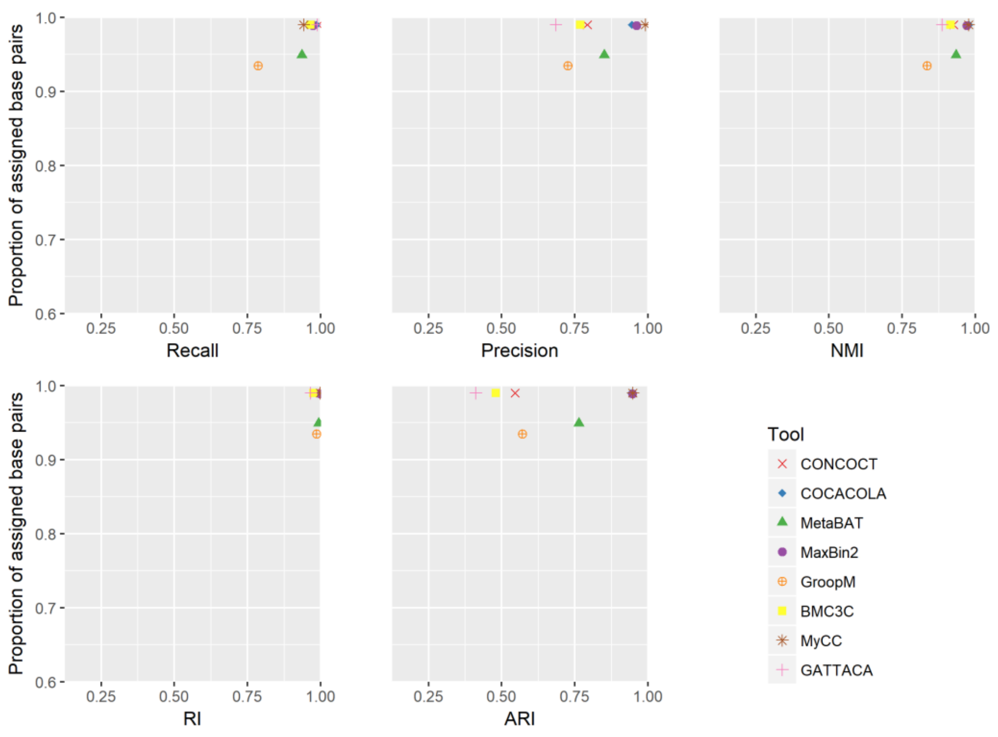


Figure 8. Performance scores for Species_60S_5.0R

As shown in Figure 8, the proportion of the assigned base pairs is between 0.9 and 1. The proportion of the assigned base pairs for GroopM and MetaBAT are the lowest. MyCC, MaxBin2, and COCACOLA are the top three methods with the highest performances in all measures but the recall values. The performance of MetaBAT follows next. The performances of GroopM, CONCOCT, BMC3C, and GATTACA are relatively low.

Species_60S_7.5R The ninth dataset consists of 60 samples each with the number of reads 7,500,000. The time spent on MEGAHIT is 9 hours and 8 minutes and the peak memory usage is 56.2GB. The number of the assembled contigs after cutting up the long contigs is

43,219 and 32,531 contigs with the length bigger than or equal to 1,500bp are used for each of the clustering method.

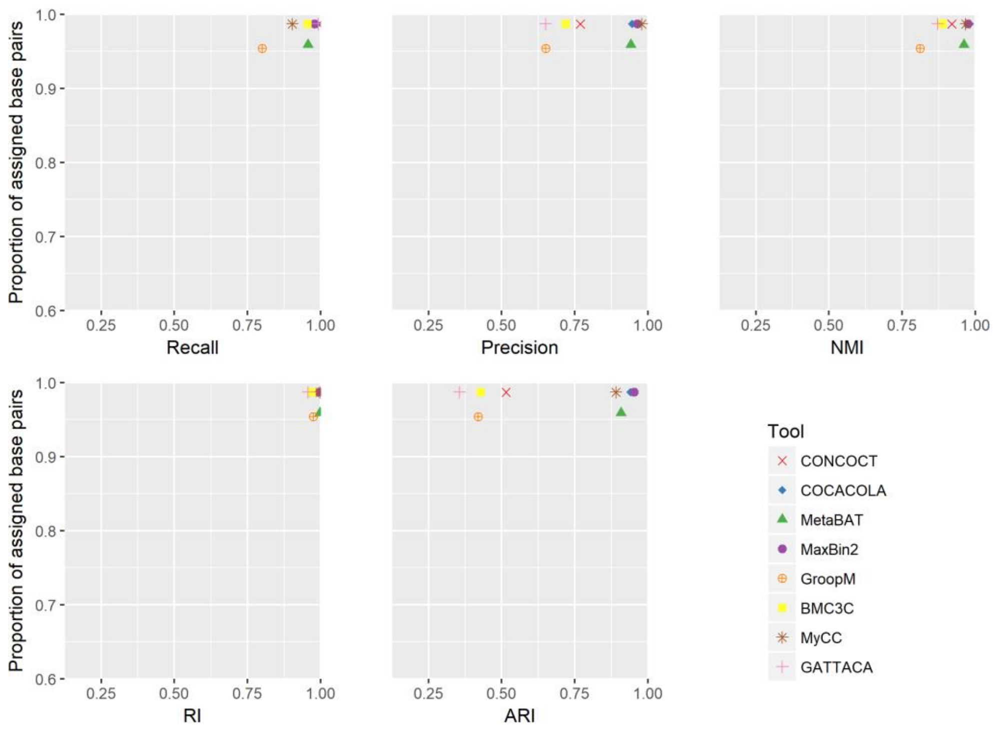


Figure 9. Performance scores for Species_60S_7.5R

As shown in Figure 9, the proportion of the assigned base pairs is between 0.95 and 1. MaxBin2, COCACOLA, MetaBAT, and MyCC are the four methods with the highest performance. Each of the four method has the precision value bigger than 0.94, the NMI value bigger than 0.96, the RI value bigger than 0.99, and the ARI value bigger than 0.89. The performances of CONCOCT, BMC3C GroopM, and GATTACA are relatively low.

Species_60S_10.0R The tenth dataset consists of 60 samples each with the number of reads 10,000,000. The time spent on

MEGAHIT is 7 hours and 29 minutes and the peak memory usage is 74.9GB. The number of the assembled contigs after cutting up the contigs is 46083 and 32,532 contigs with the length bigger than equal to 1500 are used for each of the clustering method.

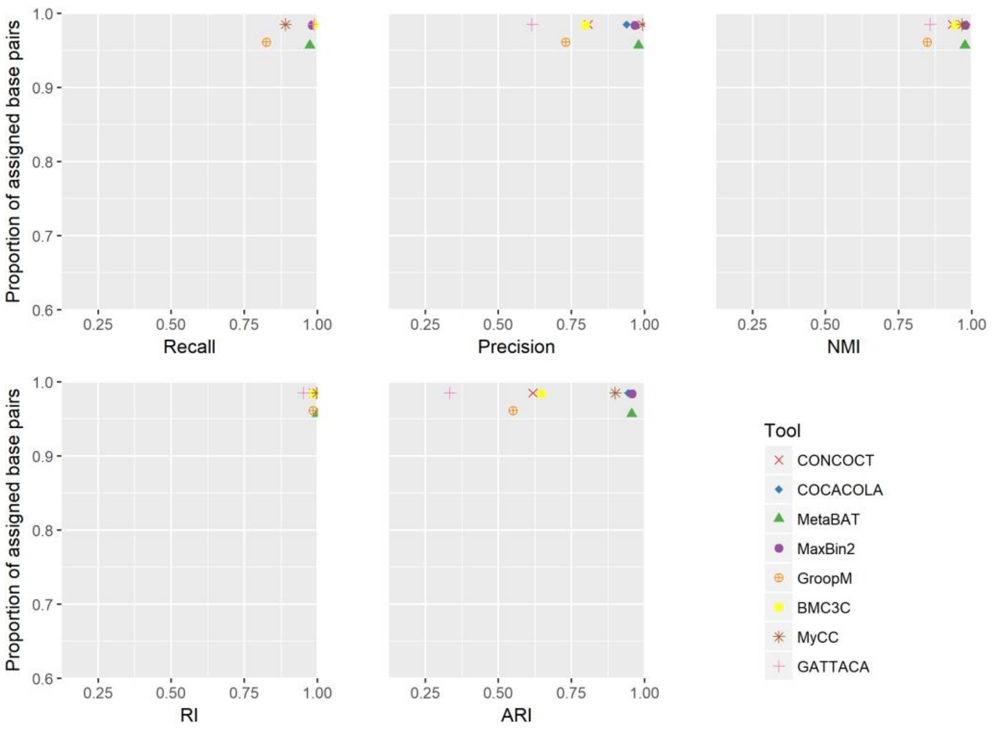


Figure 10. Performance scores for Species_60S_10.0R

The proportion of the assigned base pairs is between 0.95 and 1. Similar to Species_60S_7.5R, MaxBin2, MetaBAT, COCACOLA, and MyCC are the top four with high performances. The performances of CONCOCT and BMC3C follow next. The performances of GroopM and GATTACA are the lowest.

Species_60S_12.5R The eleventh dataset consists of 60 samples each with the number of reads 12,500,000. The time spent on

MEGAHIT is 9 hours and 26 minutes and the peak memory usage is 93.6GB. The number of assembled contigs after cutting up the long contigs is 48437 and 32,543 contigs with the length bigger than or equal to 1,500bp are used for each of the clustering method.

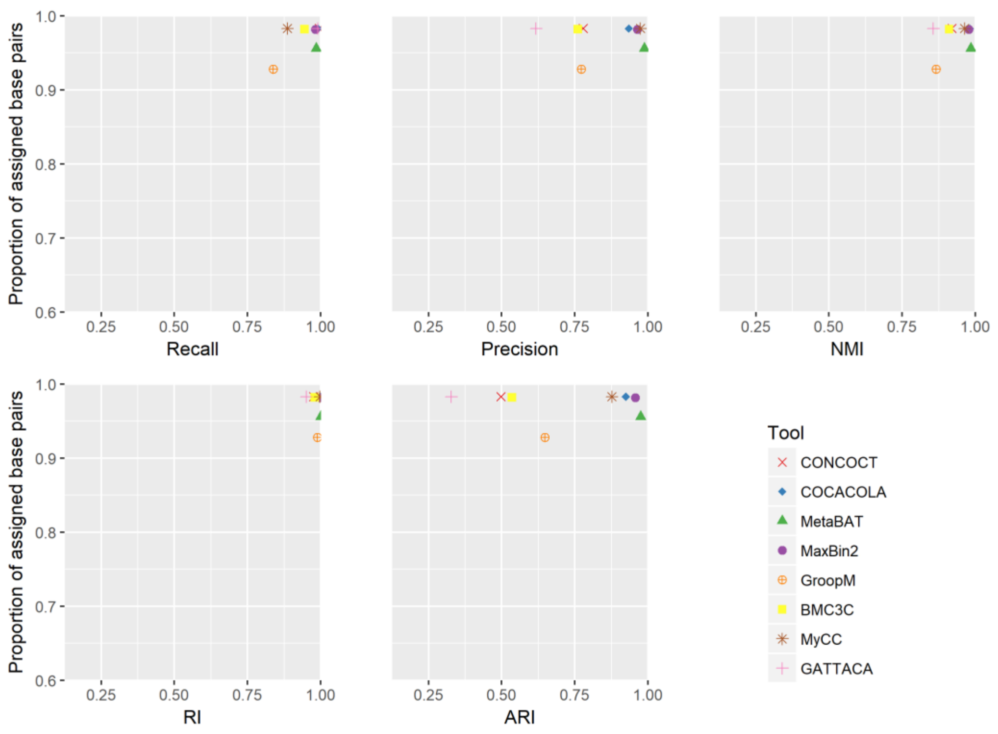


Figure 11. Performance scores for Species_60S_12.5R

As shown in Figure 11, the proportion of the assigned base pairs is between 0.9 and 1. The performance of MetaBAT is the highest for all but the recall value. The performances of MaxBin2, COCACOLA, and MyCC are also high. The performances of GroopM, BMC3C, CONCOCT, and GATTACA are relatively low.

Species_80S_12.5R The twelfth dataset consists of 80 samples each with the number of reads 12,500,000. The time spent on

MEGAHIT is 15 hours and 8 minutes and the peak memory used is 124.5GB. The number of the assembled contigs after cutting up the long contigs is 100,510 and 32,565 contigs with the length bigger than or equal to 1,500bp are used for each of the clustering method.

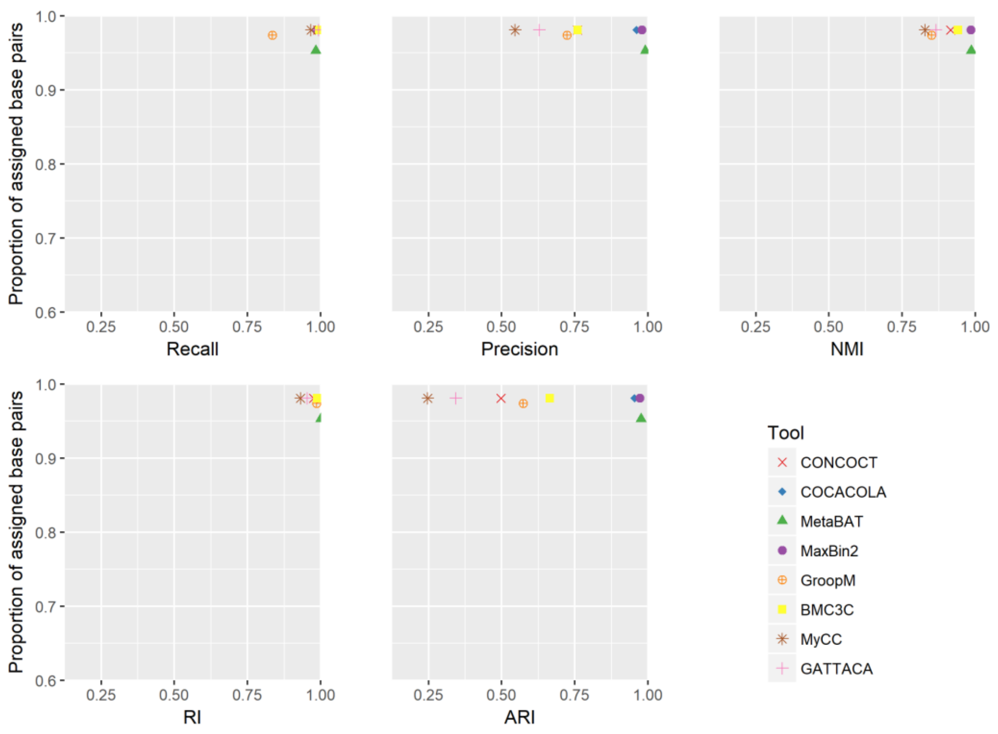


Figure 12. Performance scores for Species_80S_12.5R

As shown in Figure 12, The proportion of the assigned base pairs is between 0.95 and 1. The top three methods that performed well are MaxBin2, MetaBAT, and COCACOLA each with the recall value bigger than 0.98, the precision value bigger than or equal to 0.96, the NMI value bigger than or equal to 0.98, the RI value bigger than or equal to 0.998, and the ARI value bigger than or equal to 0.95, which are all close to 1. The performances of BMC3C, GroopM, and

CONCOCT follow next with relatively low precision and the ARI values. The performances of MyCC and GATTACA are the lowest. Note that the performance of MyCC dropped in all measures but the recall value.

Species_100S_12.5R The thirteenth dataset consists of 100 samples each with the number of reads 12,500,000. The time spent on MEGAHIT is 16 hours and 14 minutes and the peak memory used is 126.0KB. The number of the assembled contigs after cutting up the long contigs is 100,332 and 32,643 contigs with the length bigger than or equal to 1,500bp are used for each of the clustering method.

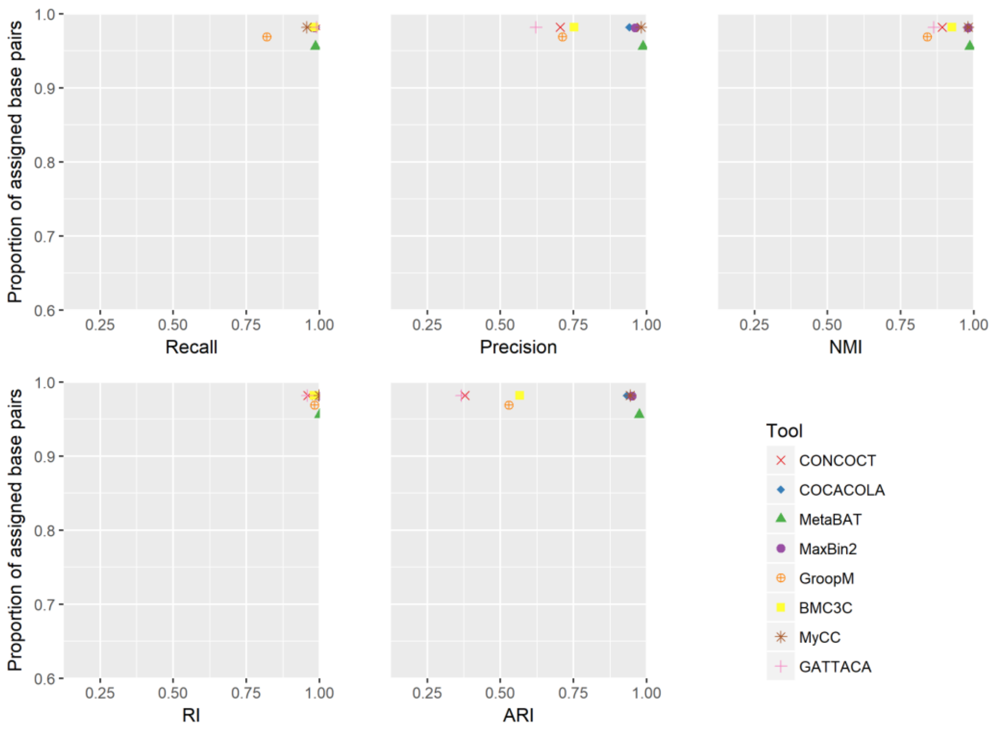


Figure 13. Performance scores for Species_100S_12.5R

As shown in Figure 13, The proportion of the assigned base pairs

is between 0.95 and 1. The top four methods that performed well are MetaBAT, MaxBin2, MyCC, and COCACOLA. The performances of GroopM, BMC3C, GroopM and CONCOCT are relatively low.

2.2.2. Results of the data with multiple strain variations

Next, we ran the binning tools on the simulated dataset of 100 species with multiple strains The following provides the results for each of the datsets.

Strains_20S_2.5R The first dataset with multiple strains consists of 20 samples each with the number of reads 2,500,000. The time spent on MEGAHIT is 1 hour and 51 minutes and the peak memory used is 94.5GB. The number of the assembled contigs after the cutting up process is 265,725 and 47,741 contigs with the length bigger than or equal to 1500bp are used for each of the clustering method.

As shown in Figure 14, the proportion of the assigned base pairs is between 0.6 and 0.65. The proportion is low compared to the proportion of the assigned base pairs for the datasets with no strain variation. From Figure 14, it is observed that the performance of CONCOCT is outstanding. The performances of GATTACA and MaxBin2 follow the performance of CONCOCT. The performances of COCACOLA, MetaBAT, GroopM, and MyCC are relatively low.

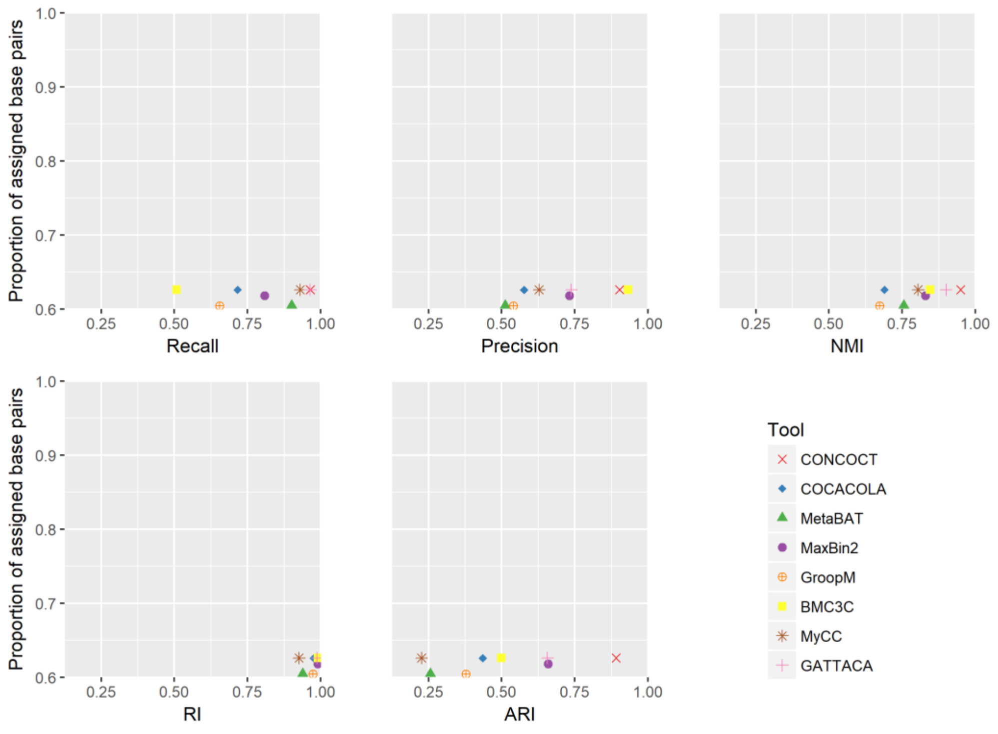


Figure 14. Performance scores for Strains_20S_2.5R

Strains_20S_5.0R The second dataset consists of 20 samples each with the number of reads 5,000,000. The time spent on MEGAHIT is 3 hours and 37 minutes and the peak memory used is 12.5GB. The number of the assembled contigs after cutting up the contigs is 295,937 and 57,488 contigs with the length bigger than or equal to 1,500bp are used for each of the clustering method.

As shown in Figure 14, the proportion of the assigned base pairs is between 0.65 and 0.7. The performance of CONCOCT is the highest overall. The performances of MaxBin2 and GATTACA follow. The overall performances of BMC3C, COCACOLA, MetaBAT, GroopM, and MyCC are the lowest.

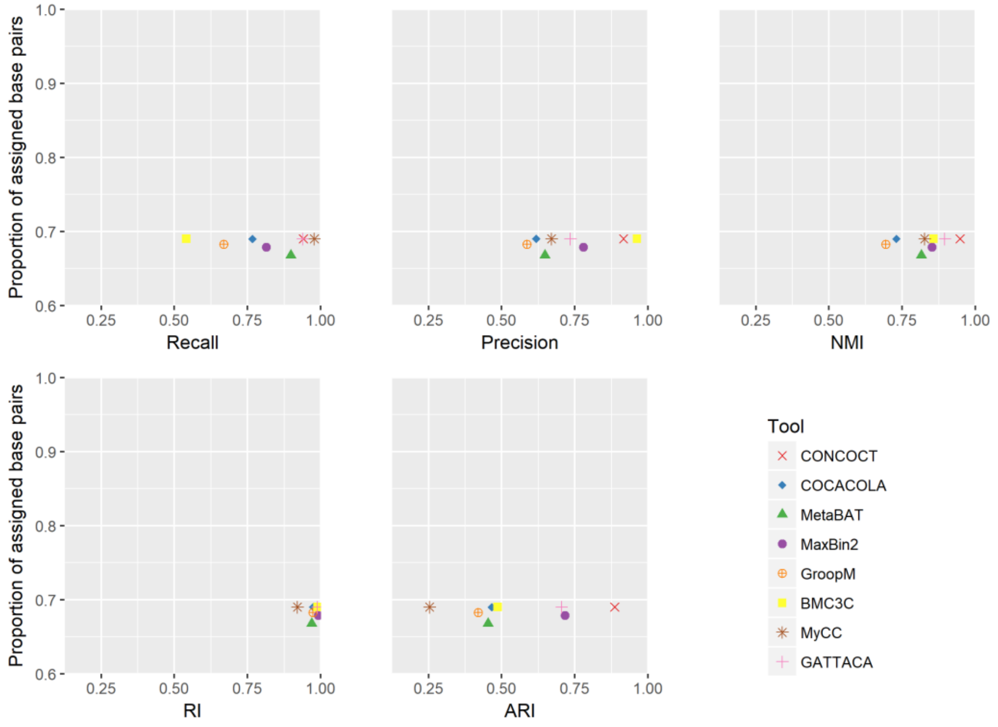


Figure 15. Performance scores for Strains_20S_5.0R

Strains_20S_7.5R The third dataset consists of 20 samples each with the number of reads 7,500,000. The time spent on MEGAHIT is 8 hours and 20 minutes and the peak memory used is 18.8GB. The number of the assembled contigs after the cutting up process is 180,741 and 56,518 contigs with the length bigger than or equal to 1,500bp are used for each of the clustering method.

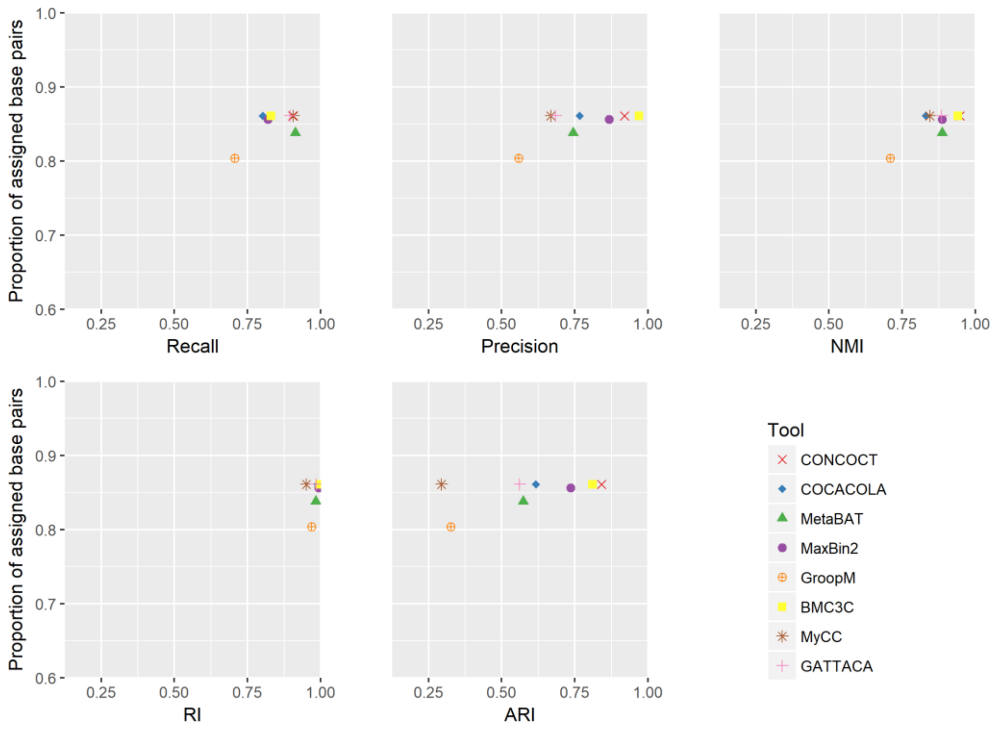


Figure 16. Performance scores for Strains_20S_7.5R

As shown in Figure 16, the proportion of the assigned base pairs is between 0.8 and 0.9. The performances of CONCOCT and BMC3C are the highest, followed by the performance of MaxBin2. The performances of COCACOLA, MetaBAT, GATTACA follow. The performances of GroopM, and MyCC are relatively low, though MyCC scored high in the recall value.

Strains_20S_10.0R The fourth dataset consists of 20 samples each with the number of reads 10,000,000. The time spent on MEGAHIT is 10 hours and 48 minutes and the peak memory used is 25.0GB. The number of the assembled contigs after the cutting up process is 475,822 and 64,387 contigs with the length bigger than or equal to 1,500bp are used for each of the clustering method.

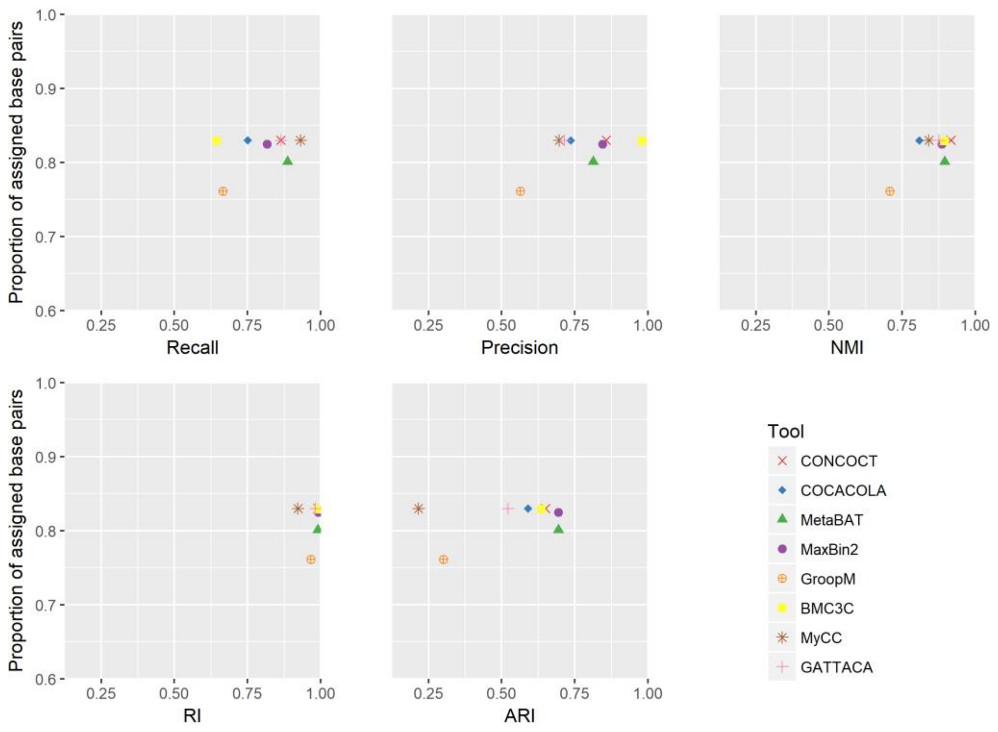


Figure 17. Performance scores for Strains_20S_10.0R

As shown in Figure 17, the proportion of the assigned base pairs is between 0.75 and 0.85. The performances of CONCOCT, BMC3C, MaxBin2, and MetaBAT are the highest, closely followed by the performances of COCACOLA and GATTACA. The performances of MyCC and GroopM are the lowest. Note that BMC3C scored the highest in precision, but scored the lowest in recall.

Strains_20S_12.5R The fifth dataset consists of 20 samples each with the number of reads 12,500,000. The time spent on MEGAHIT is 9 hours and 3 minutes and the peak memory used is 31.3GB. The number of the assembled contigs after the cutting up process is

252,511 and 60,692 contigs with the length bigger than or equal to 1,500bp are used for each of the clustering method.

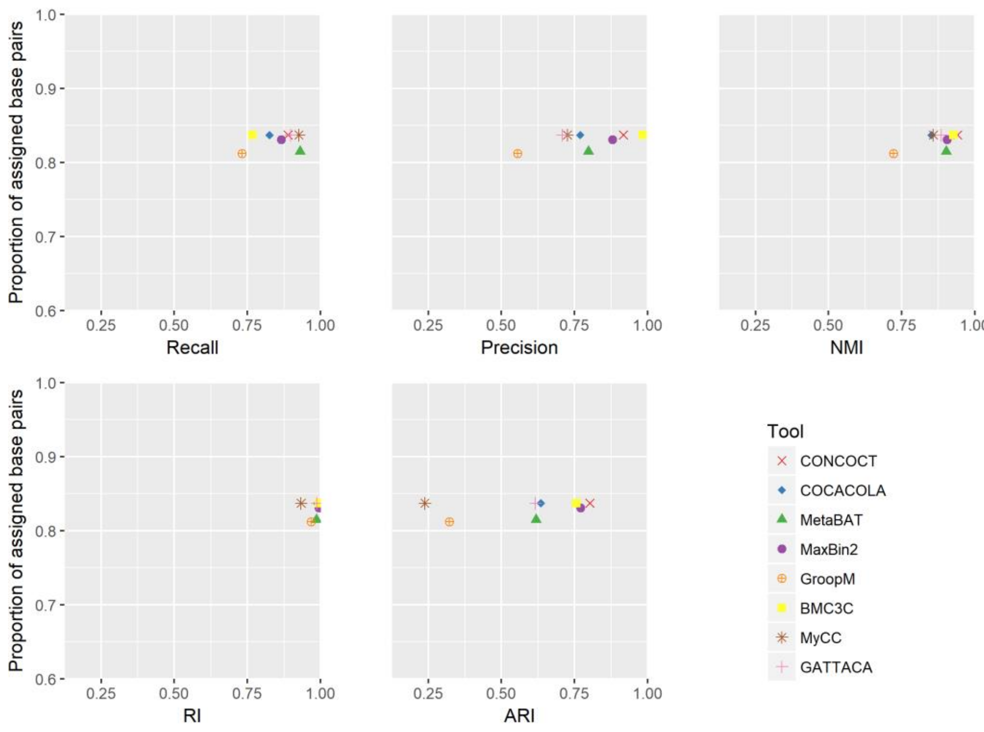


Figure 18. Performance scores for Strains_20S_12.5R

As shown in Figure 18, the proportion of the assigned base pairs is between 0.8 and 0.85. The performances of CONCOCT, MaxBin2, and BMC3C are relatively high. The performances of COCACOLA, MetaBAT, and GATTACA follow next. The performances of GroopM and MyCC are the lowest.

Strains_40S_12.5R The sixth dataset consists of 40 samples each with the number of reads 12,500,000. The memory and time spent on MEGAHIT is 17 hours and 31 minutes and the peak memory used is 62.5GB. The number of the assembled contigs after the cutting up process is 406,449 and 68,846 contigs with the length bigger than or equal to 1,500bp are used for each of the clustering method..

As shown in Figure 19, the proportion of the assigned base pairs is between 0.8 and 0.85. The performances of MaxBin2, BMC3C, and MetaBAT are the highest, followed by the performance of CONCOCT. The performances of COCACOLA and GATTACA follows. The performances of GroopM and MyCC are the lowest overall.

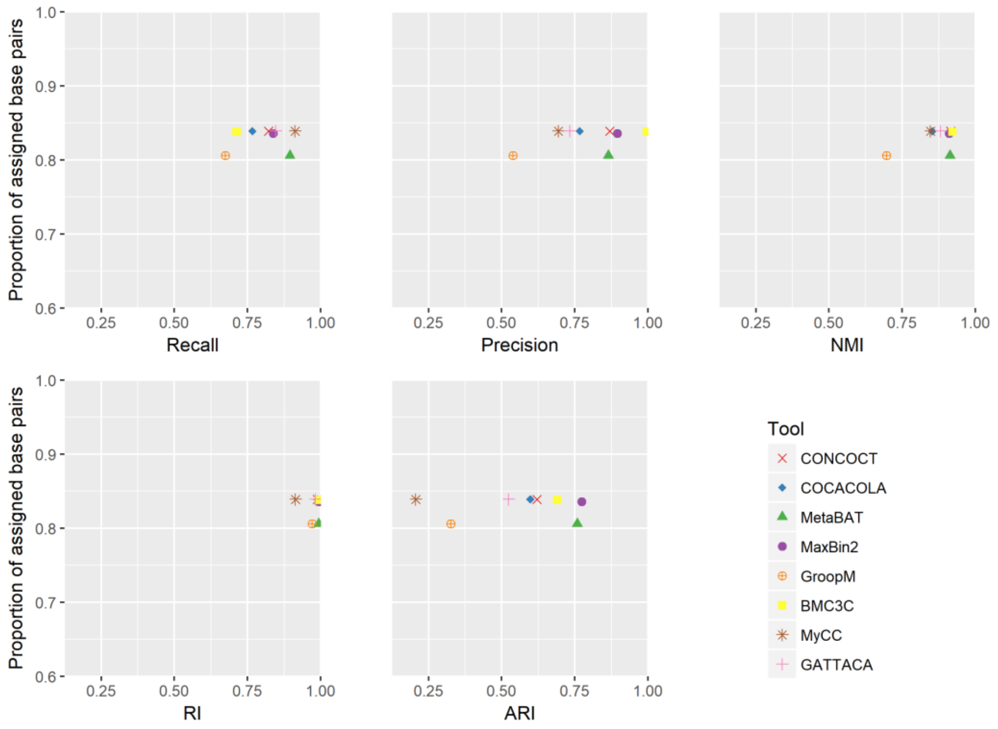


Figure 19. Performance scores for Strains_40S_12.5R

Strains_60S_2.5R The seventh dataset consists of 60 samples each with the number of reads 2,500,000. The memory and time spent on MEGAHIT is 5 hours and 2 minutes and the peak memory used is 18.7GB. The number of the assembled contigs after the cutting up process is 454,580 and 62,556 contigs with the length bigger than or equal to 1,500bp are used for each of the clustering method.

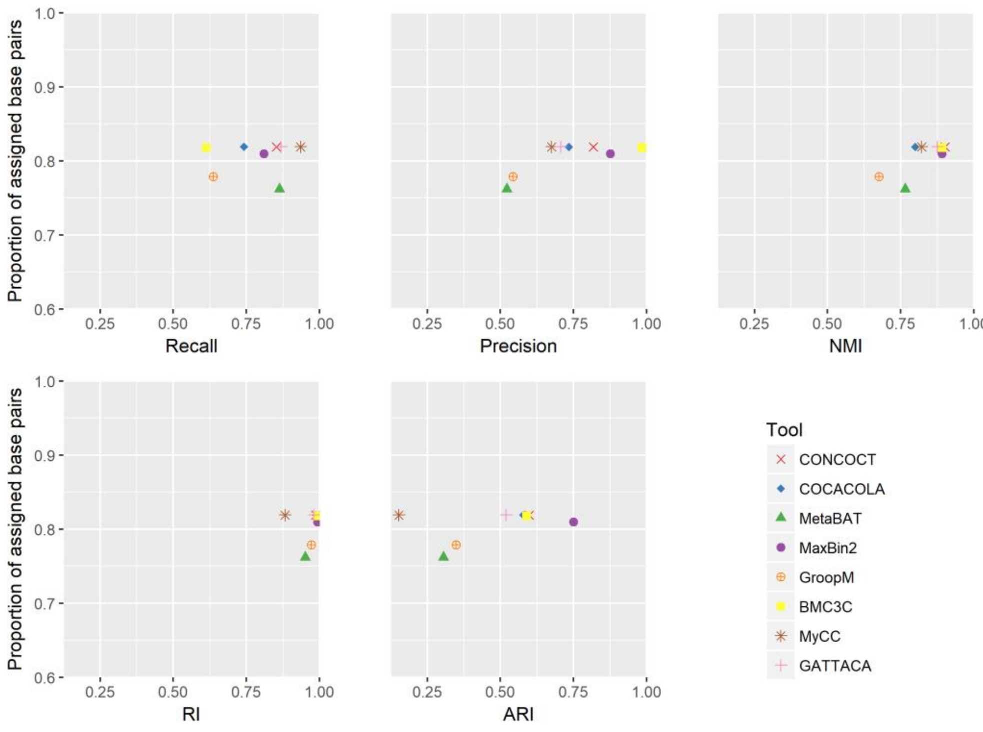


Figure 20. Performance scores for Strains_60S_2.5R

As shown in Figure 20, the proportion of the assigned base pairs is between 0.75 and 0.85. The performance of MaxBin2 is the highest, followed by the performances of CONCOCT and BMC3C. The performances of COCACOLA and GATTACA follow next. The performances of GroopM, MetaBAT and MyCC are the lowest. The precision value for BMC3C is the highest for BMC3C and the recall value for BMC3C is the lowest.

Strains_60S_5.0R The eighth dataset consists of 60 samples each with the number of reads 5,000,000. The time spent on MEGAHIT is 10 hours and 17 minutes and the peak memory used is 37.5GB. The number of the assembled contigs after the cutting up process is 533,058 and 56,318 contigs with the length bigger than or equal to 1,500bp are used for each of the clustering method.

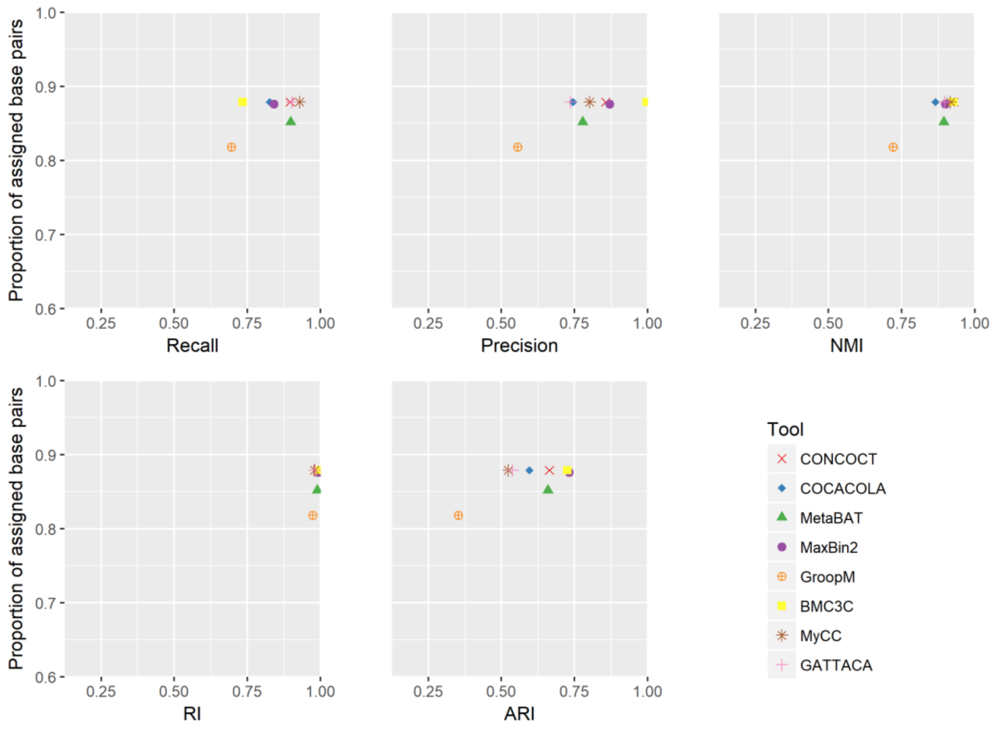


Figure 21. Performance scores for Strains_60S_5.0R

As shown in Figure 21, the proportion of the assigned base pairs is between 0.8 and 0.9. The difference in performance across the binning tools is not big. The ARI scores of MaxBin2, BMC3C, CONCOCT, and MetaBAT are the highest. The ARI scores of

COCACOLA, GATTACA. and MyCC follows next. The overall performance of GroopM is the lowest.

Strains_60S_7.5R The ninth dataset consists of 60 samples each with the number of reads 7,500,000. The time spent on MEGAHIT is 15 hours and 7 minutes and the peak memory used is 56.3GB. The number of the assembled contigs after the cutting up process is 672,454 and 58,131 contigs with the length bigger than or equal to 1,500bp are used for each of the clustering method.

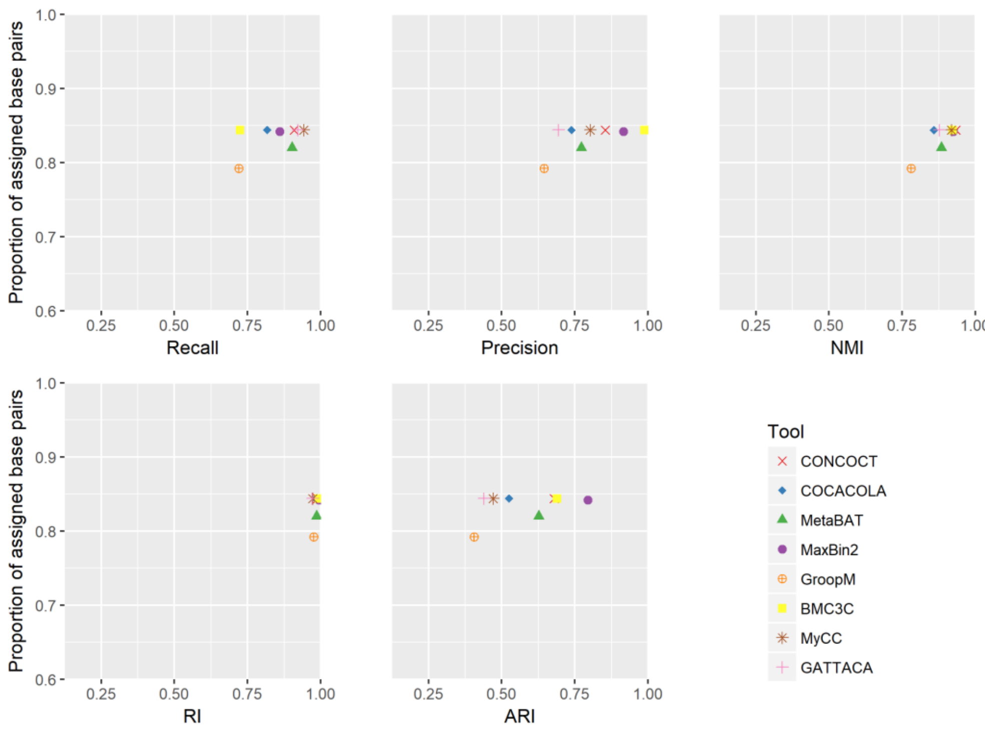


Figure 22. Performance scores for Strains_60S_7.5R

As shown in Figure 22, the proportion of the assigned base pairs is between 0.75 and 0.85. The performances of MaxBin2, BMC3C, CONCOCT, and MetaBAT are the highest, followed by the performances of COCACOLA, MyCC, and GATTACA. The performance of GroopM is the lowest with the lowest scores in all metrics and has the lowest proportion of the assigned base pairs.

Strains_60S_10.0R The tenth dataset consists of 60 samples each with the number of reads 10,000,000. The time spent on MEGAHIT is 18 hours and 8 minutes and the peak memory used is 74.9GB. The number of the the assembled contigs after the cutting up process is 754,126 and 69,490 contigs with the length bigger than or equal to 1,500bp are used for each of the clustering method.

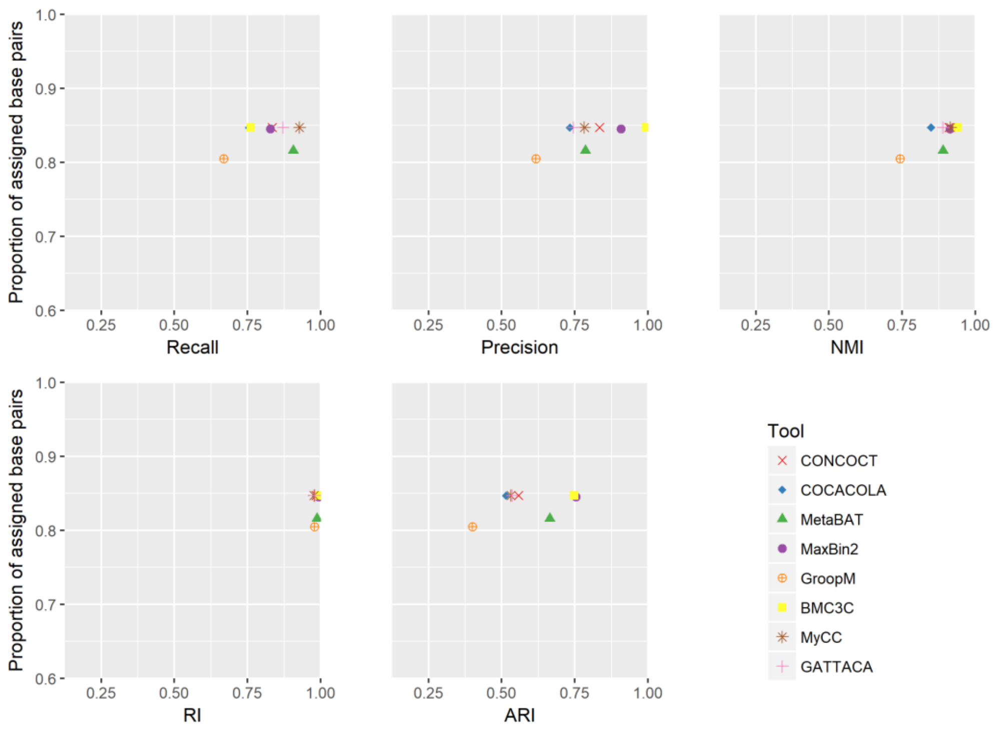


Figure 23. Performance scores for Strains_60S_10.0R

As shown in Figure 23, the proportion of the assigned base pairs is between 0.8 and 0.85. The performances of MaxBin2 and BMC3C is high, followed by the performances of MetaBAT, CONCOCT, MyCC, and COCACOLA. The performance of GroopM is the lowest.

Strains_60S_12.5R The eleventh dataset consists of 60 samples each with the number of reads 12,500,000. The time spent on MEGAHIT is 23 hours and 35 minutes and the peak memory used is 93.7GB. The number of the assembled contigs after the cutting up process is 889,500 and 68,181 contigs with the length bigger than or equal to 1,500bp are used for each of the clustering method.

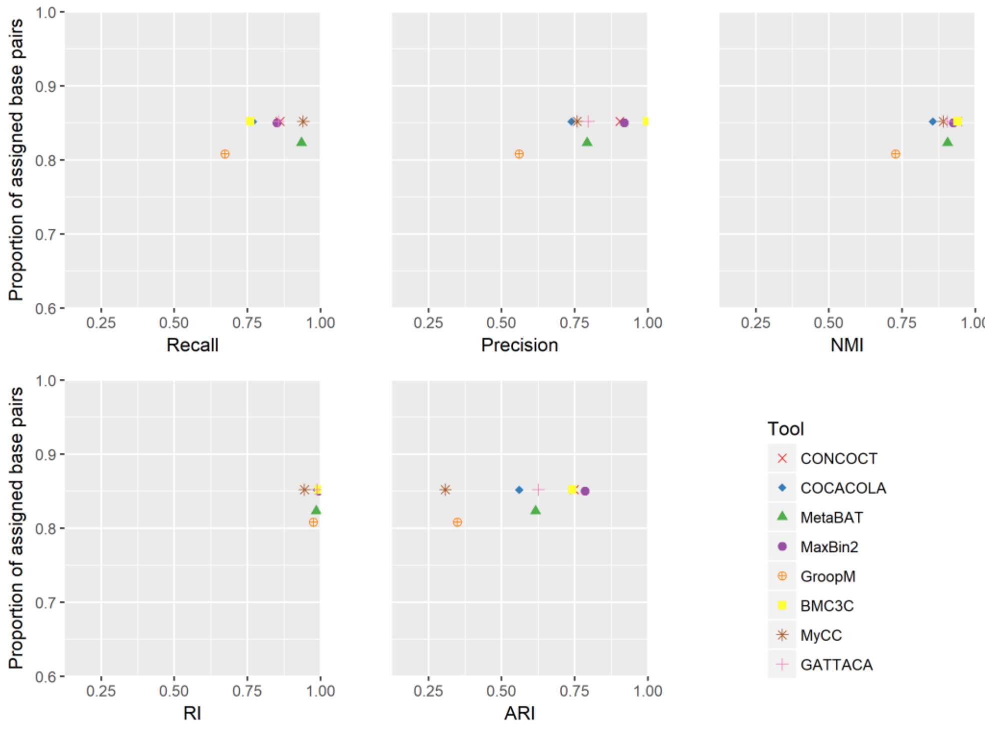


Figure 24. Performance scores for Strains_60S_12.5R

As shown in Figure 24, the proportion of the assigned base pairs is between 0.8 and 0.9. The performances of MaxBin2, CONCOCT, and BMC3C are relatively high, which are followed by the performances of MetaBAT, GATTACA, and COCACOLA. The performances of GroopM and MyCC is the lowest. The recall value for MyCC is the highest whereas its RI and ARI values are the lowest.

Strains_80S_12.5R The twelfth dataset consists of 80 samples each with the number of reads 12,500,000. The memory and time spent on MEGAHIT is 27 hours and 12 minutes and the peak memory used is 124.6GB. The number of the assembled contigs after the cutting up process is 1,235,916 and 62,785 contigs with the length bigger than or equal to 1,500bp are used for each of the clustering method..

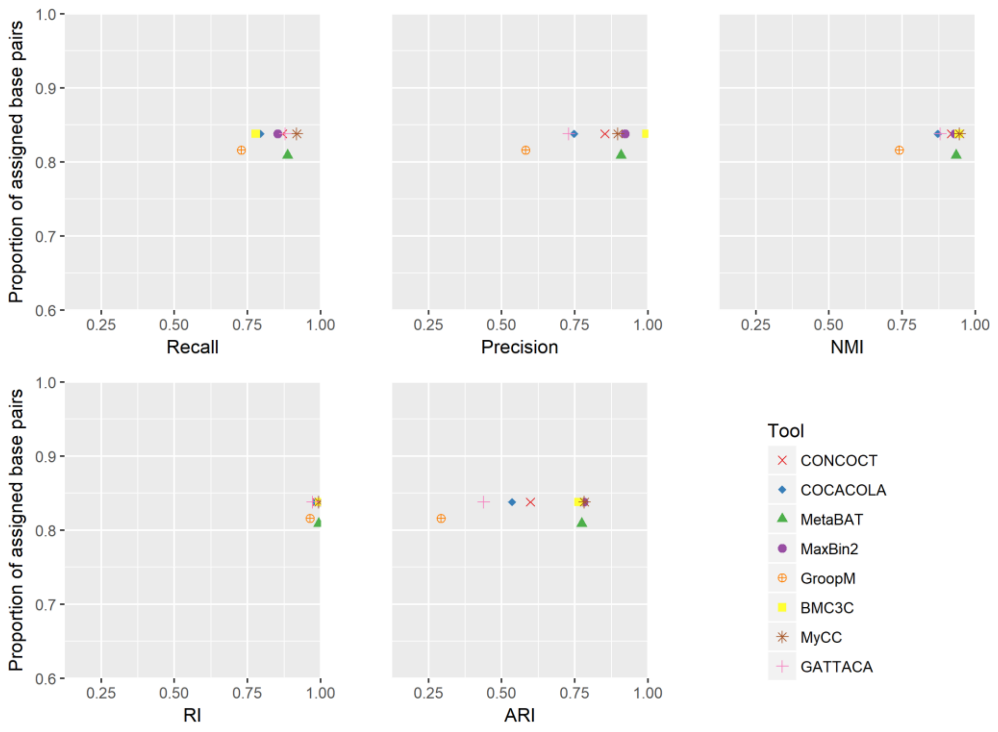


Figure 25. Performance scores for Strains_80S_12.5R

As shown in Figure 25, the proportion of the assigned base pairs

is between 0.8 and 0.85. The performances of MaxBin2, MyCC, and MetaBAT are high. The performance of CONCOCT follows. The performances of COCACOLA and GATTACA follows, and the performance of GroopM is the lowest.

Strains_100S_12.5R The thirteenth dataset consists of 100 samples each with the number of reads 12,500,000. The time spent on MEGAHIT is 18 hours 12 minutes and the peak memory used is 126.0GB. The number of the assembled contigs after the cutting up process is 651,838 and 59,981 contigs with the length bigger than or equal to 1,500bp are used for each of the clustering method.

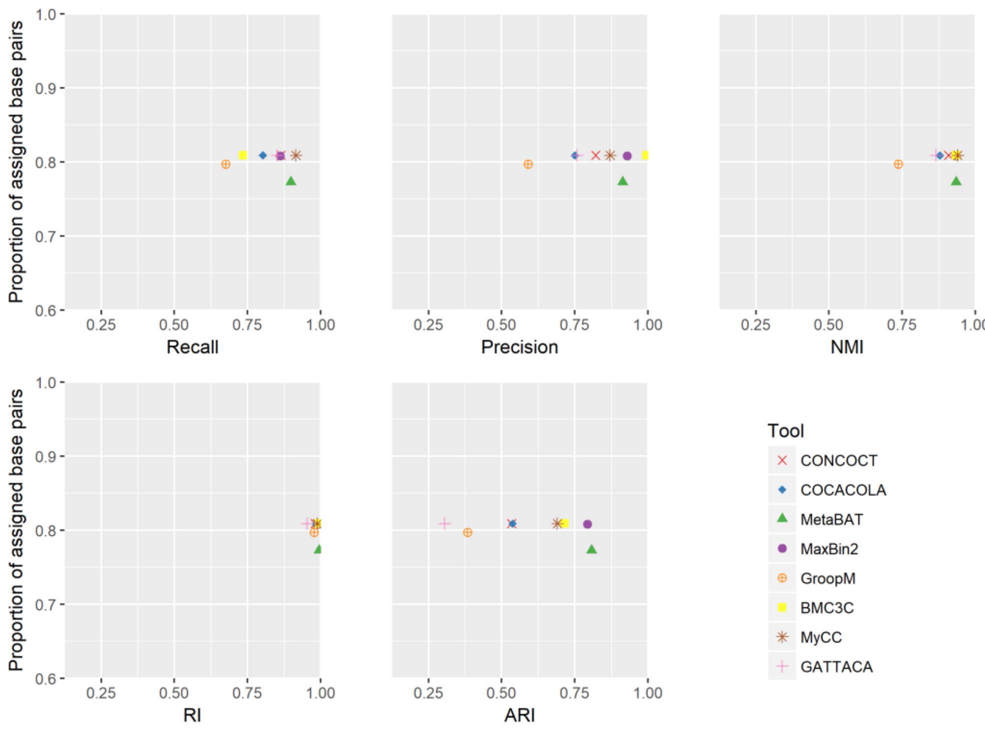


Figure 26. Performance scores for Strains_100S_12.5R

As shown in Figure 26, the proportion of the assigned base pairs

is between 0.75 and 0.85. The performances of MetaBAT, MaxBin2, BMC3C, and MyCC are relatively good. Next follows the performances of CONCOCT and COCACOLA, which is followed by the performances of GroopM and GATTACA.

2.3. The performance of each method

In addition to the species−level performance, strain−level performance is included in this section. The species−level performance scores indicate the performance values obtained by constructing the matrix $\mathbf{A}$ described in section 2.1.11, where $\mathbf{T}$, the number of columns of $\mathbf{A}$, is the number of species, 100. The species−level performance scores indicate the performance values obtained by constructing the matrix $\mathbf{A}$ described in section 2.1.11, where $\mathbf{T}$ is the number of strains, 210. Note that the recall values tend to be higher in the strain−level scores than in the species−level scores. The precision values tend to be lower in the species−level scores than in the strain−level scores. The precision values tend to be high if the number of the clusters inferred in high.

Note that the matrix $\mathbf{A}^T$ can be depicted as a heatmap, which is a plot that shows which cluster corresponds to which species based on a color key. The heatmaps of the results of clustering is provided in Appendix. The high number of contigs both in a cluster and in a genome is represented by yellow. We can also identify which of the contig binning methods performed well by comparing the heatmaps between different contig binning methods.

2.3.1. The performance of CONCOCT

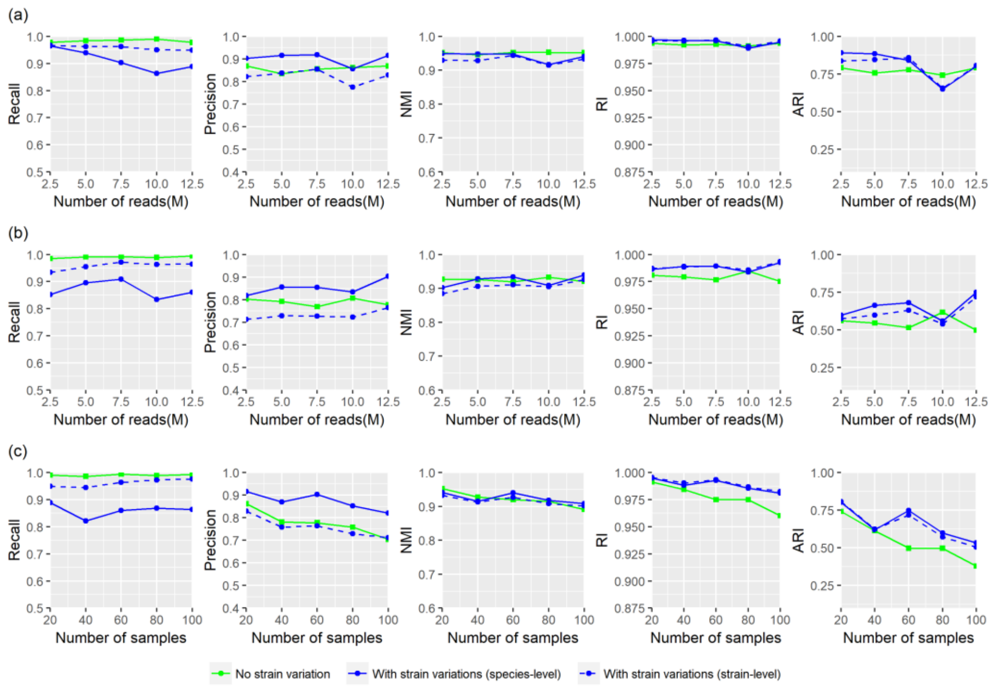


Figure 27. The performance of CONCOCT

Figure27 (a) and (b) shows the values of the performance statistics for CONCOCT against the number of reads in millions where the number of samples is 20 and 60 respectively. Figure 27 (c) shows the performance scores for CONCOCT against the number of samples where the number of reads for each sample is 12.5 million. The green solid lines correspond to the performance scores for the datasets with no strain variation, the blue solid lines correspond to the species−level performance for the datasets with multiple strain variations, and the blue dashed lines correspond to the strain−level performance scores for the datasets with multiple strain variations.

For the datasets with no strain variation, the performance of CONCOCT tends to decreases as the number of samples increases, as depicted in Figure 27 (c). In addition, by comparing Figure 27 (a) and Figure 27 (b), it is observed that the patterns of fluctuations of the two are similar and the performances for the datasets with the number of samples 60 is lower than the performances for the datasets with the number of samples 20 given the fixed number of reads per sample.

By comparing the green solid lines and the blue solid lines in Figure 27, we can see that the presence of multiple strains in the datasets does not negatively affect the performance of CONCOCT. Also by comparing the blue solid lines and the blue dashed lines in Figure 27, we can observe that when multiple strains are present, the species−level scores and the strain−level scores are similar except for the recall and the precision values.

Overall, the performance scores of CONCOCT are relatively high. The performance scores of CONCOCT decrease as the number of samples increases. CONCOCT displays high levels of resolution when the depth of coverage is relatively low. The original paper [14] states that the performance of CONCOCT increases as the depth of coverage increases, which does not accord with our observations. There is also a possibility that information lost during the process of PCA may result in lower performance. Further evaluations of the performance of CONCOCT is required.

2.3.2. The performance of COCACOLA

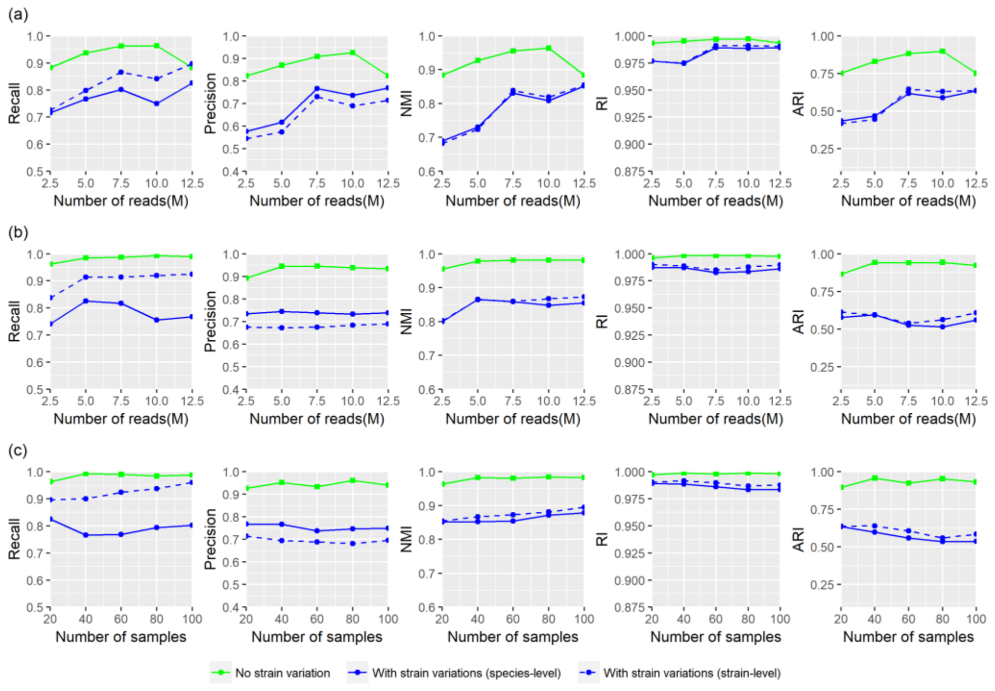


Figure 28. The performance of COCACOLA

Figure 28 (a) and (b) shows the values of the performance measures for COCACOLA against the number of reads in millions where the number of samples is 20 and 60 respectively. Figure 28 (c) shows the performance scores for COCACOLA against the number of samples where the number of reads for each sample is 12.5 million.

From Figure 28 (a), it is observed that for datasets with the sample size 20, the performance values tend to increase with the increase in the number of reads. For the datasets with multiple strain variations, the overall scores are relatively low.

By comparing the green solid lines and the blue solid lines in Figure 28, it is observed that the performance for the datasets with no strain variation is always higher than the performance for the datasets with multiple strain variations. By comparing blue solid lines and blue dashed lines in Figure 28, we can observe that for the datasets with multiple strain variations, the difference between the species-level scores and the strain-level scores is not big except for the recall values.

The performance of CONCOCT is consistently high for the datasets with no strain variations if either the number of samples or the depth of coverage in the dataset is high. For the datasets with multiple strain variations, the performance is low.

2.3.3. The performance of MetaBAT

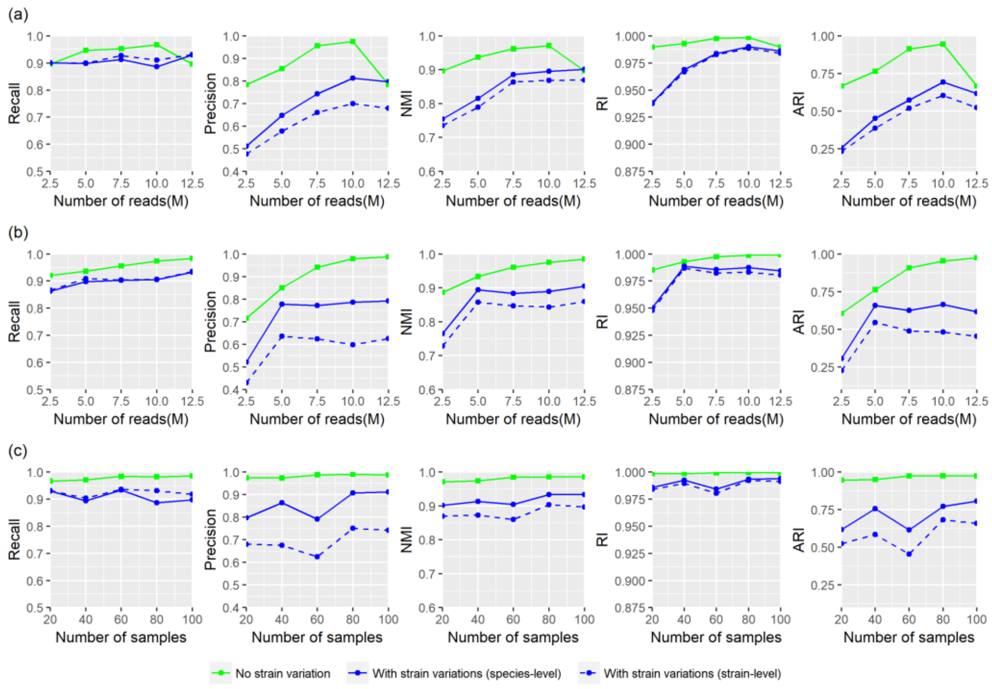


Figure 29. The performance of MetaBAT

Figure 29 (a) and (b) shows the values of performance measures for MetaBAT against the number of reads in millions where the number of samples is 20 and 60 respectively. Figure 29 (c) shows the performance scores for MetaBAT against the number of samples when the number of reads for each sample is 12.5 million.

As shown in Figure 29 (a), (b), for the datasets with no strain variation, the performance values of MetaBAT increase greatly with the increase in the number of reads. For the datasets with multiple strain variations, the increase in the performance values is more clearly demonstrated if the number of samples in the dataset is small.

As shown in Figure 29 (c), for the datasets with no strain variation, if the depth of coverage is large, the performance is close to the maximum value of the scores 1. By comparing the green solid lines and the blue solid lines in Figure 29, it is observed that the performance for the datasets with no strain variation is always higher than the performance for the datasets with multiple strain variations. By comparing blue solid lines and blue dashed lines in Figure 29, it is observed that for the datasets with multiple strain variations, the species-level scores are higher than the strain-level scores especially for the ARI values.

Overall, the performance of MetaBAT increase greatly with the increase in the the number of reads per sample when no strain variation is present. The presence of multiple strains hinders MetaBAT to perform well.

2.3.4. The performance of MaxBin2

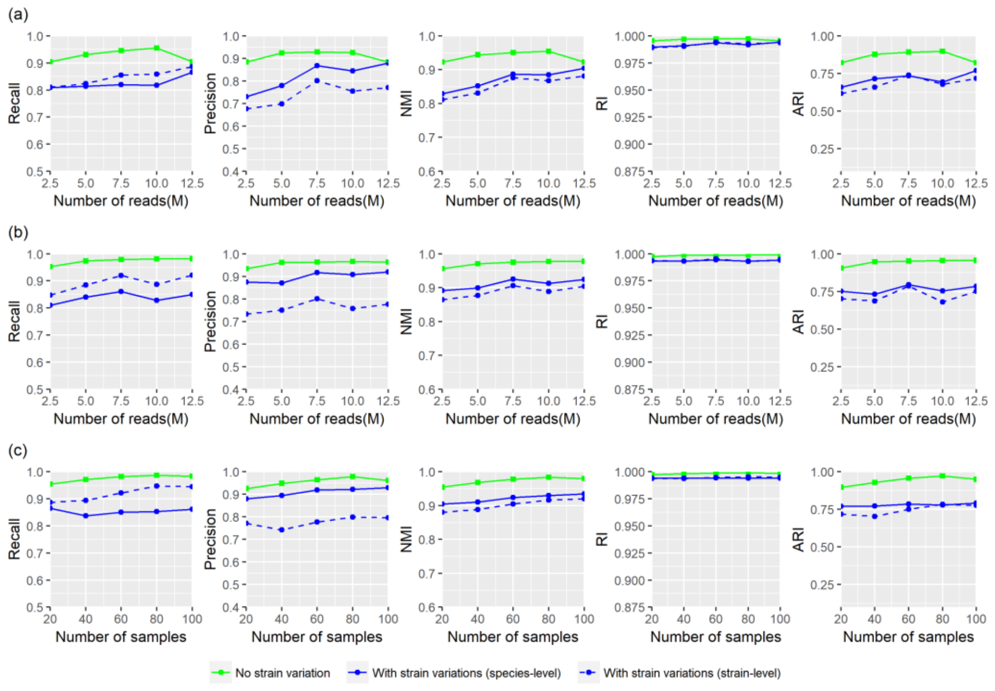


Figure 30. The performance of MaxBin2

Figure 30 (a) and (b) shows the values of performance measures for MaxBin2 against the number of reads in millions where the number of samples is 20 and 60 respectively. Figure 30 (c) shows the performance scores for MaxBin2 against the number of samples when the number of reads for each sample is 12.5 million.

As can be observed from Figure 30, the performance scores of MaxBin2 is relatively consistent across datasets, and the scores tend to increases as the number of reads per sample or the number of samples increases. By comparing the green solid lines and the blue solid lines in Figure 30, we can see that the performance scores for the datasets with no strain variation is always higher than the scores for the datasets with no strain variation. By comparing the blue solid lines and the blue dashed lines in Figure 30, we can see that for the datasets with multiple strain variations, the difference between the species−level scores and the strain−level scores is not big.

Overall, the performance of MaxBin2 is consistently high. The performance of MaxBin2 is higher for the datasets with large number of samples or high number of reads per sample. The MaxBin2 algorithm is capable of grouping the contigs belonging to the same species together for datasets with various strains.

2.3.5. The performance of GroopM

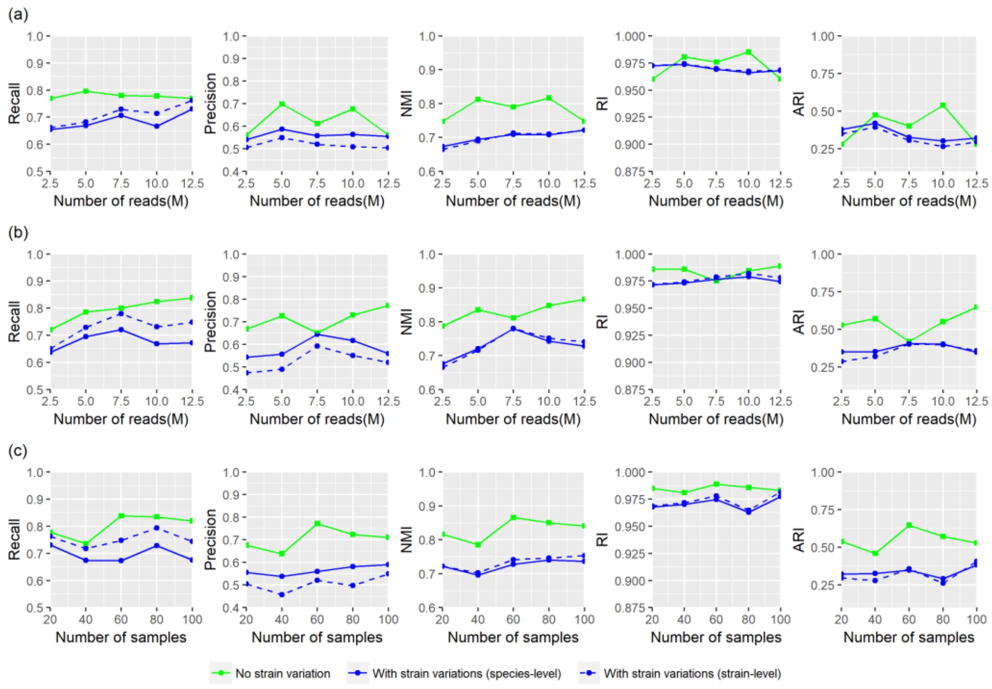


Figure 31. The performance of GroopM

Figure 31 (a) and (b) shows the values of performance measures for GroopM against the number of reads in millions where the number of samples is 20 and 60 respectively. Figure 31 (c) shows the performance scores for GroopM against the number of samples when the number of reads for each sample is 12.5 million.

As can be observed by comparing the green solid lines and the blue solid lines in Figure 31, the performance for the datasets with no strain variation tends to be higher than the performance for the datasets with multiple strain variations. By comparing the blue solid lines and the blue dashed lines in Figure 31, we can see that for the datasets with multiple strain variations, the difference between the species−level scores and the strain−level scores is not big.

The overall performance of GroopM is low. One of the possible reasons for this is that we did not use further stages of refine steps provided by the GroopM package. The use of varying minimum length of the contigs imposed to reduce the memory usage may have also affected the performance. The minimum contig length in the core stage was set to 2,500bp for for Strains_20S_10.0R, the the minimum contig length in the core stage was set to 3,500bp for for Strains_60S_7.5R, the the minimum contig length in the core stage was set to 5,000bp for for Strains_60S_10.0R, and the the minimum contig length in the core stage was set to 2,500bp for for Strains_60S_12.5R. For the other datasets, the length of the minimum contig length was set to 1,500bp.

2.3.6. The performance of BMC3C

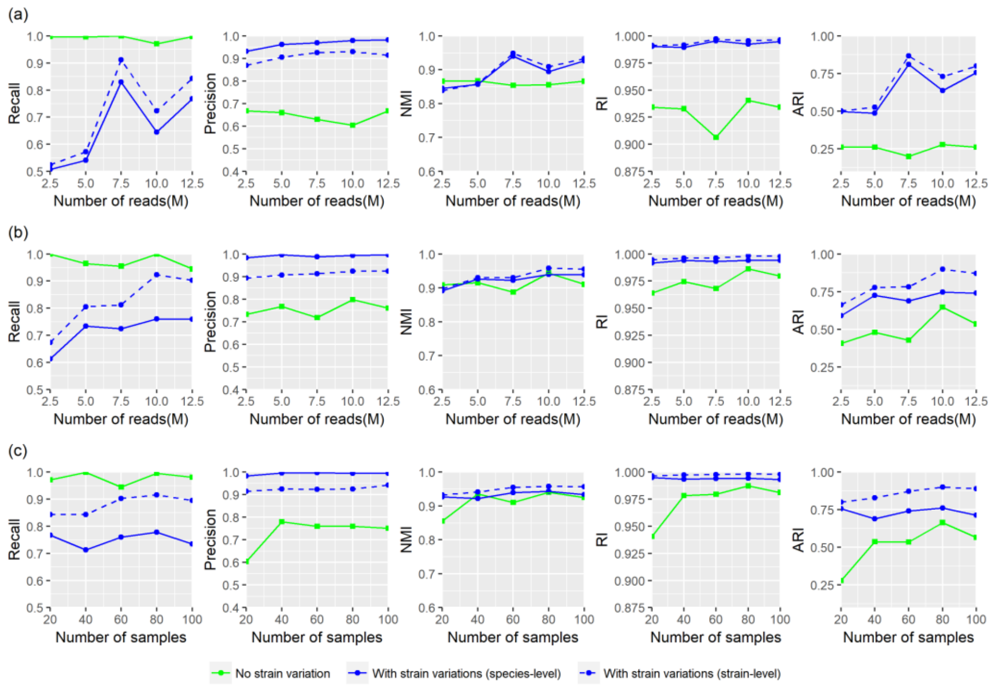


Figure 32. The performance of BMC3C

Figure32 (a) and (b) shows the values of performance measures for BMC3C against the number of reads in millions where the number of samples is 20 and 60 respectively. Figure 32 (c) shows the performance scores for BMC3C against the number of samples where the number of reads for each sample is 12.5 million.

By comparing the green solid lines and the blue solid lines in Figure 32, we can see that the score of BMC3C is higher for the datasets with multiple strain variations than for the datasets with no strain variation. For the datasets with multiple strain variations, the species−level scores, depicted by the dashed blue lines in Figure 32, are lower than the strain−level scores, depicted by the solid blue lines in Figure 32, except for the precision values. This aspect is not observed in the other methods, which shows the potential of BMC3C to cluster contigs into strains.

Overall, BMC3C tends to perform better on the datasets with multiple strains than on the datasets with no strain variation. BMC3C tends to overestimates the number of clusters, especially for the datasets with no strain variation. Since the strain−level performance values for the datasets with multiple variations is high, BMC3C is an adequate method for strain−level resolutions.

2.3.7. The performance of MyCC

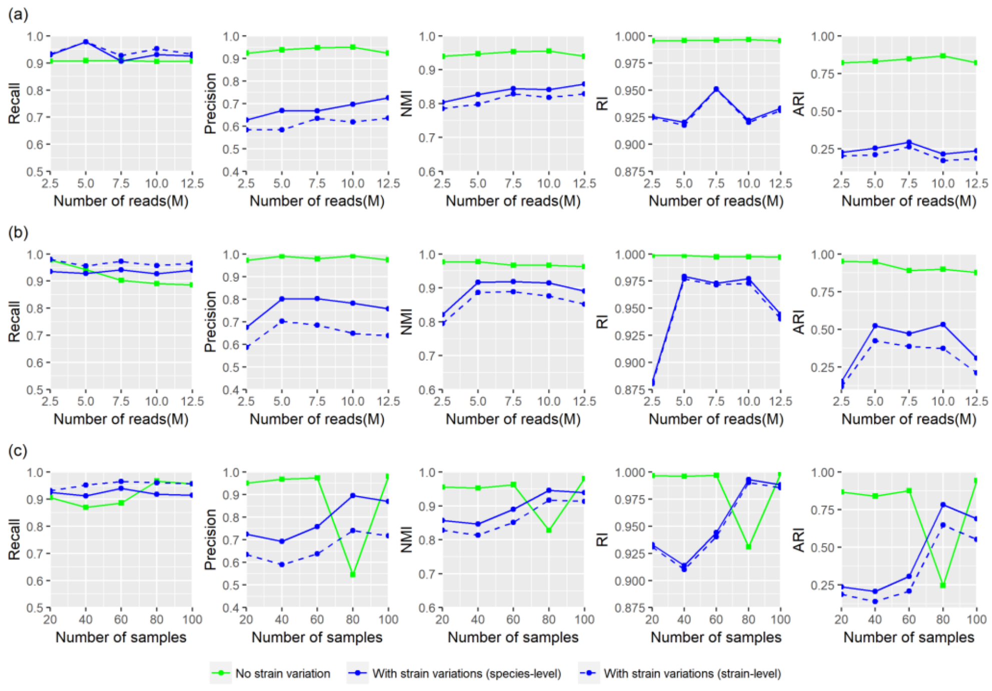


Figure 33. The performance of MyCC

Figure 33 (a) and (b) shows the values of performance measures for MyCC against the number of reads in millions where the number of samples is 20 and 60 respectively. Figure 33 (c) shows the performance scores for MyCC against the number of samples where the number of reads for each sample is 12.5 million.

The performance of MyCC is relatively high for the datasets with multiple strain variations, though it has a poor performance for Species_80S_12.5R. As shown by the green lines in Figure 33 (a), when the number of samples is 20, for the datasets with no strain variations, values of performance increase as the depth of coverage increases. By comparing the blue solid lines and the blue dashed lines in Figure 33, we can see that the performance for the datasets with no strain variation tends to be higher than the performance for the datasets with multiple strain variations except for the recall values and for the datasets with 80 samples. By comparing the blue solid lines and the blue dashed lines in Figure 33, we can see that for the datasets with multiple strain variations, the difference between the species-level scores and the strain-level scores is not big except for the precision values.

Overall, the performance scores of MyCC for the datasets with no strain variation are high except for Species_80S_12.5R. For Species_80S_12.5R, by observing the heatmap of MyCC plotted in Figure A.24 of Appendix, we can observe that a some of the clusters each contain contigs from various species. Though the other clusters each corresponds to a species relatively well, clusters containing multiple species mainly caused the drop in the performance.

2.3.8. The performance of GATTACA

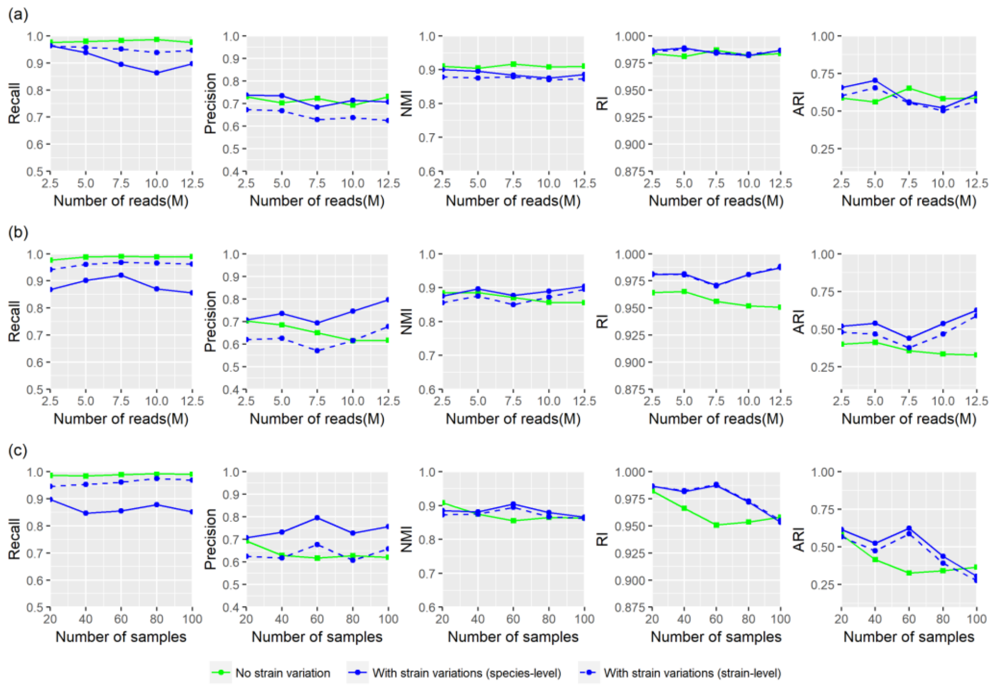


Figure 34. The performance of GATTACA

Figure 34 (a) and (b) shows the values of performance measures for GATTACA against the number of reads in millions where the number of samples is 20 and 60 respectively. Figure 34 (c) shows the performance scores for GATTACA against the number of samples when the number of reads for each sample is 12.5 million.

Since the clustering algorithm for CONCOCT and GATTACA are similar, the patterns of the performance plotted in Figure 34 are similar to the patterns of the performance of CONCOCT shown in Figure 27. By comparing the green solid lines and the blue solid lines in Figure 34, we can see that the presence of multiple strains in a dataset does not seem to negatively affect the performance of GATTACA. By comparing the blue solid lines and the blue dashed lines in Figure 34, we can also observe that for the datasets with multiple strains, the species-level scores and the strain-level scores are similar except for the recall and the precision values.

The overall performance of GATTACA is relatively low. The performance may have been affected by the mean coverage profile used in this study, whereas in the original paper, the coverage profile was generated by the kmer counts for each contig in each sample so that the read mapping process is not included.

Chapter 3. Conclusion

Metagenomics, sparked by the next generation sequencing, led to the need of the taxonomic classification of microbial genomes in the samples obtained from the nature. We have compared the eight taxonomy−independent contig binning methods that have the potential to discover novel genomes from the sequence data.

We have generated 26 distinct in silico datasets, half with no strain variation and the other half with multiple strain variations. After implementing each of the eight methods on each in−silico datasets and calculating the performance scores for each method, we observed that the binning methods with high performances vary by the datasets.

We first illustrate the results from the 13 datasets with no strain variation. The performance of CONCOCT was high for the datasets relatively small in sample size. Hence even with small datasets which are cost effective and time efficient to obtain, CONCOCT can provide reliable results.

COCACOLA, MetaBAT, MaxBin2, and MyCC are the methods with high performances for the datasets with larger number of the samples or the higher depths of coverage, implying that these methods utilize a large amount of information effectively. Therefore, researchers can be more confident in increasing the number of samples or the depth of coverage to have increased performance of these tools.

We now illustrate the results of the datasets with multiple strain variations. It is observed that the performance of CONCOCT is high for the datasets relatively small in the sample size or the depth of coverage. The performances of BMC3C, MaxBin2 and MetaBAT were relatively high for the larger datasets. For BMC3C, the number of clusters inferred was closer to the number of strains, 210, than the number of species, 100. Also, the performance scores in strain-level is high, indicating it is possible to use BMC3C for clustering contigs into strains with no reference genomes.

The overall performances of GroopM and GATTACA were low. GroopM was conducted without the use of the optional refining steps, discarding the chimeric set of contigs identified by the algorithm. This may have led to the lower performance of GroopM. Manual inspections may lead to the higher performance. GATTACA was implemented with the input mean coverage profile. However, the algorithm is originally built to use the coverage profile generated by the k-mer counts for each contig in each sample. Using the coverage profile generated by the k-mer counts may have positive effects on the performance of GATTACA.

It should be taken into account that there are some limitations in this study regarding the method of comparing performances of the binning tools. First of all, we compared the performance of the binning methods mainly based on the values of the score metrics. Since CONCOCT, COCACOLA, MyCC, and GATTACA binned every contig while MetaBAT, MaxBin2, GroopM, and BMC3C had some contigs unbinned, evaluating the performance solely on the value of performance measures can produce biased results.

In this study, we have only conducted one round of binning for each method and each dataset. Repeated rounds of each binning method may have given us the mean performance values. The level of variability of the resulted bins may also have been obtained, which may have enabled us to conduct hypothesis tests on the values of performance for more confidence in the performance evaluations.

Since we have generated the datasets within a pool of 419 known strains, it could have resulted in biased results and the datasets may have had limited variability in the complexity of the datasets. Each dataset is not truly independent, and with the unexplored sequence of novel strains in the samples from the real world, different performance results may be obtained. Also, the distributions used by StrainMetaSim, may not always reflect the aspect of the real samples.

Taking these limitation into account, further research on the evaluation of the contig binning methods will give more insights. Along with the eight methods implemented in this article, the performance of other automatic taxonomy independent binning methods such as BinSanity [65], CoMet [66], and IFCM [67], can also be evaluated. The software such as AMBER [68] and CheckM [69] are also available besides Validate.pl, used in our study, for evaluation of the inferred bins. Performance measures such as F1 score are also available, which is the harmonic mean of the recall and precision [38].

Reference

[1] National Research Council (US) Committee on Metagenomics (2007). *The New Science of Metagenomics: Revealing the Secrets of Our Microbial Planet*. Washington, DC: The National Academies Press

[2] Quince C, Walker AW, Simpson JT, Loman NJ, & Segata N (2017). Shotgun metagenomics, from sampling to analysis. *Nat Biotechnol.,* 35, 833–844.

[3] Breitwieser FP, Lu J, & Salzberg SL (2017). A review of methods and databases for metagenomic classification and assembly. *Brief Bioinform.*

[4] Baker G.C., Smith J.J., & Cowan D (2003). Review and Re-Analysis of Domain-Specific 16S Primers. *J Microbiol Methods.*, 55:541–555.

[5] Wang Y, Leung HC, Yiu SM, & Chin FY (2012). MetaCluster 5.0: a Two Round Binning Spproach for Metagenomic Data for Low Abundance Species in a Noisy Sample. *Bioinformatics.,* 28:i356–62.

[6] DeSantis TZ, Hugenholtz P, Larsen N, et al. (2006). Greengenes, a chimera-checked 16S rRNA gene database and workbench
compatible with ARB. *Appl Environ Microbiol*, 72:5069–72.

[7] Cole JR, Chai B, Farris RJ, et al. (2005). The Ribosomal

Database

Project (RDP-II): sequences and tools for high-throughput rRNA analysis. *Nucleic Acids Res.*, 33:D294-6.

[8] Carlton JM, Angiuoli SV, Suh BB, et al. (2002). Genome sequence and comparative analysis of the model rodent malaria parasite Plasmodium yoelii yoelii. *Nature*, 419:512-19.

[9] Yarza P., Yilmaz P., Pruesse E., Glöckner F. O., Ludwig W., Schleifer K.-H., et al. (2014). Uniting the classification of cultured and uncultured bacteria and archaea using 16S rRNA gene sequences. *Nat Rev Microbiol.,* 12:635-645.

[10] Pei AY, Oberdorf WE, and Nossa CW, et al (2010). Diversity of 16S rRNA genes within individual prokaryotic genomes *Appl Environ Microbiol.,* 76:3886-3897.

[11] Benjamin HM, Laura SO, Julian RM, & Julie AKM (2018). The implementation of omics technologies in cancer microbiome research. *Ecancermedicalscience.,* 12:864.

[12] Eloe-Fadrosh EA, Ivanova NN, Woyke T, et al. (2016). Metagenomics uncovers gaps in amplicon-based detection of microbial diversity. *Nat Microbiol.,* 1:15032.

[13] Bentley DR, Balasubramanian S, Swerdlow HP, Smith GP, Milton J, Brown CG., et al. (2008). Accurate whole human genome sequencing using reversible terminator chemistry. *Nature*, 456:53-59.

[14] Alneberg J, Bjarnason BS, De Bruijn I, Schirmer M, Quick J, Ijaz UZ, Lahti L, Loman NJ, Andersson AF, & Quince C (2014). Binning metagenomics contigs by coverage and composition. *Nat Methods.,* 11:1144 -1146.
[15] Sedlar K, Kupkova K, Provaznik I. (2017). Bioinformatics strategies for taxonomy independent binning and visualization of sequences in shotgun metagenomics. *Comput Struct Biotechnol J.,* 15:48-55.
[16] Sandberg R, Bränden CI, Ernberg I, Cöster J. (2003). Quantifying the species-specificity in genomic signatures, synonymous codon choice, amino acid usage and G+C content. *Gene.,* 311:35-42.
[17] Dick GJ, Andersson AF, Baker BJ, Simmons SL, Thomas BC, Yelton AP, & Banfield JF (2009). Community-wide analysis of microbial genome sequence signatures. *Genome Biol.* 10:R85.
[18] Hugenholtz P (2002). Exploring prokaryotic diversity in the genomic era. Genome Biol., 3(2):reviews0003
[19] Calusinska M, Marynowska M, Goux X, Lentzen E, & Delfosse P (2016). Analysis of dsDNA and RNA viromes in methanogenic digesters reveals novel viral genetic diversity. *Environ. Microbiol.* 18:1162-1175.
[20] Labonte JM, Suttle CA (2015). Corrigendum: metagenomic and whole genome analysis reveals new lineages of Gokushoviruses and

biogeographic separation in the sea. *Front Microbiol.,* 6:114.

[21] Ge X, Li Y, Yang X, Zhang H, Zhou P, Zhang Y, Shi Z (2012). Metagenomic analysis of viruses from bat fecal samples reveals many novel viruses in insectivorous bats in China. *J Virol.,* 86:4620-4630.

[22] Sharon I, Morowitz MJ, Thomas BC, Costello EK, Relman DA, & Banfield JF (2013). Time series community genomics analysis reveals rapid shifts in bacterial species, strains, and phage during infant gut colonization. *Genome Res.,* 23:111-120.

[23] Nayfach S, Rodriguez-Mueller B, Garud N, Pollard KS. An integrated metagenomics pipeline for strain profiling reveals novel patterns of bacterial transmission and biogeography. Genome Res. 2016;26:1612-25.

[24] Luo C, Knight R, Siljander H, Knip M, Xavier RJ, & Gevers D. (2015). ConStrains identifies microbial strains in metagenomic datasets. *Nat Biotechnol.,*33(10):1045-52.

[25] Quince C, Delmont TO, Raguideau S, Alneberg J, Darling AE, Collins G, Eren AM (2017). DESMAN: a new tool for de novo extraction of strains from metagenomes. *Genome Biol.,* 18:181.

[26] O' Brien JD, Didelot X, Iqbal Z, Amenga-Etego L, Ahiska B, Falush D (2014). A Bayesian approach to inferring the phylogenetic structure of communities from metagenomic data. *Genetics.,* 3:925-37.

[27] Zinicola M., Higgins H., Lima S., Machado V., Guard C., Bicalho R (2015). Shotgun metagenomic sequencing reveals functional genes and microbiome associated with bovine digital dermatitis. *PLoS ONE.,* 10:e0133674.

[28] Guo J, Li J, Chen H, Bond PL, Yuan Z (2017). Metagenomic analysis reveals wastewater treatment plants as hotspots of antibiotic resistance genes and mobile genetic elements. *Water Res.,* 123:468–478.

[29] Subirats J., Sanchez–Melsio A., Borrego C. M., Balcazar J. L., Simonet P. (2016). Metagenomic analysis reveals that bacteriophages are reservoirs of antibiotic resistance genes. *Int. J. Antimicrob Agents.*, 48:163–167.

[30] Amos G. C., Zhang L., Hawkey P. M., Gaze W. H., Wellington E. (2014). Functional metagenomic analysis reveals rivers are a reservoir for diverse antibiotic resistance genes. *Vet Microbiol.,* 171:441–447.

[31] Schmeisser C., Steele H., Streit W.R. (2007). Metagenomics, biotechnology with non–culturable microbes. *Appl Microbiol Biotechnol.* 75:955–962.

[32] Donia MS, Cimermancic P, Schulze CJ, Wieland Brown LC, Martin J, Miltreva M, et al. (2014). A systematic analysis of biosynthetic gene clusters in the human microbiome reveals a common family of antibiotics. *Cell.,* 1758:1402–1414.

[33] Lu YY, Chen T, Fuhrman, JA, and Sun F (2017). COCACOLA: binning metagenomic contigs using sequence COmposition, read CoverAge, CO-alignment and paired-end read LinkAge. *Bioinformatics*, 33(6):791-798.
[34] Kang DD, Froula J, Egan R, & Wang Z (2015). MetaBAT, an efficient tool for accurately reconstructing single genomes from complex microbial communities. *PeerJ*, 3:e1165.
[35] Wu YW., Simmons BA, & Singer SW (2016). MaxBin 2.0: an automated binning algorithm to recover genomes from multiple metagenomic datasets. *Bioinformatics*, 32:605-607.
[36] Imelfort M, Parks D, Woodcroft BJ, Dennis P, Hugenholtz P, &Tyson GW (2014). GroopM: an automated tool for the recovery of population genomes from related metagenomes. *PeerJ*, 2:e603.
[37] Yu G, Jiang Y, Wang J, Zhang H. & Luo H (2018). BMC3C: Binning Metagenomic Contigs using Codon usage, sequence Composition and read Coverage. *Bioinformatics,* 34(24):4172-4179.
[38] Lin HH & Liao YC (2016). Accurate binning of metagenomic contigs via automated clustering sequences using information of genomic signatures and marker genes. *Sci Rep.*, 6:24175.
[39] Popic V, Kuleshov V, Snyder M, et al. (2018). Fast Metagenomic Binning via Hashing and Bayesian Clustering. *J Comput Biol.,* 25(7): 677–688.

[40] Huang W, Li L, Myers JR, & Marth GT (2012). Art: a next-generation sequencing read simulator. *Bioinformatics,* 28:593-4.

[41] Dinghua L, Chi-Man L, Luo R, Sadakane K, Lam TW (2015). MEGAHIT: an ultra-fast single-node solution for large and complex metagenomics assembly via succinct de Bruijn graph. *Bioinformatics,* 31:1674-6.

[42] Li H & Durbin R (2010).. Fast and accurate long-read alignment with Burrows-Wheeler transform. *Bioinformatics*, 26(5):589-95.

[43] Li H (2011). A statistical framework for SNP calling, mutation discovery, association mapping and population genetical parameter estimation from sequencing data. *Bioinformatics,* 27(21):2987-93.

[44] Corduneanu A & Bishop CM (2001). Variational Bayesain model selection for mixture distributions. *In Artif Intell Stat.*

[45] Kim J & Park H (2008). Sparse nonnegative matrix factorization for clustering. *Technical Report,* GT-CSE-08-01, Georgia Institute of Technology, Atlanta, Georgia, USA.

[46] Cai, D. et al. (2011) Graph regularized nonnegative matrix factorization for data representation. *IEEE Trans Pattern Anal Mach Intell.*, 33:1548-1560.

[47] Kaufman L, Rousseeuw P (1987). Clustering by means of medoids. Amsterdam: North-Holland, 405-416.

[48]Dupont CL, Rusch DB, Yooseph S, Lombardo MJ, Richter RA,

Valas R, et al. (2012). Genomic insights to SAR86, an abundant and uncultivated marine bacterial lineage. *ISME J.*,6:1186-1199.

[49] Hough PVC. (1962). Method and means for recognizing complex patterns. *Google Patents*.

[50] Kohonen T (1982). Self-organized formation of topologically correct feature maps. *Biological Cybernetics.*, 43:59-69.

[51] Hughes,A.L. and Langley,K.J. (2007) Nucleotide usage, synonymous substitution pattern, and past recombination in genomes of Streptococcus pyogenes. *Infect Genet Evol.,* 7:188-196.

[52] MacQueen J (1967). Some methods for classification and analysis of multivariate observations. In Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability, pp. 281-297.

[53] Shi,J. and Malik,J (2000). Normalized cuts and image segmentation. *IEEE Trans Patt Anal Mach Intell.*, 22, 888-905.

[54] Yu,S.X and Shi,J. (2003) Multiclass spectral clustering. In Ninth IEEE International Conference on Computer Vision, Vol. 1. pp.313-319.

[55] Boisvert, S., Raymond, F., Godzaridis, E., Laviolette, F. & Corbeil, J. Ray (2012). Meta: scalable de novo metagenome assembly and profiling. *Genome Biol.,* 13:R122.

[56] Kultima, J. R. et al. (2012). MOCAT: a metagenomics assembly and gene prediction toolkit. *Plos One.,* 7:e47656.

[57] Sunagawa, S. et al. (2013). Metagenomic species profiling using universal phylogenetic marker genes. *Nat Methods.*, 10:1196-1199.

[58] Edgar RC (2010). Search and clustering orders of magnitude faster than BLAST. Bioinformatics, 26:2460-2461.

[59] Laczny, C. C., Pinel, N., Vlassis, N. & Wilmes, P (2014). Alignment-free visualization of metagenomic data by nonlinear dimension reduction. *Sci Rep.,* 4:4516.

[60] Aitchison, J. The statistical analysis of compositional data. (Blackburn Press, 2003).

[61] Maaten, L. v. d. Barnes-Hut-SNE. arXiv abs/1301.3342 (2013).

[62] Frey BJ & Dueck D (2007). Clustering by passing messages between data points. *Science*, 315:972-976.

[63] Von Luxburg, U (2007). A tutorial on spectral clustering. *Statistics and computing*, 17:395-416.

[64] Cichelli, RJ (1980) Minimal perfect hash functions made simple. *Communications of the ACM*, 23(1):17-19.

[65]Graham, E. D., Heidelberg, J. F., and Tully, B. J. (2016). BinSanity: unsupervised clustering of environmental microbial assemblies using coverage and affinity propagation. *PeerJ*, 5, e3035.

[66]Herath D, Tang S-L, Tandon K, Ackland D, Halgamuge S (2017). CoMet: a workflow using contig coverage and composition for binning a metagenomic sample with high precision. *BMC*

Bioinformatics. 18(Suppl 16):S14.

[67]Liu Y, Hou T, Kang B, Liu F (2017). Unsupervised binning of metagenomics assembled contigs using improved fuzzy c-means method., IEEE/ACM Trans *Comput Biol Bioinform.*, 14(6):1459-1467.

[68]Meyer F, Hofmann P, Belmann P, Garrido-Oter R, Fritz A, Sczyrba A, & McHardy AC (2018). AMBER: Assessment of metagenome binners. *Gigascience.*, 7(6).

[69]Parks DH, Imelfort M, Skennerton CT, Hugenholtz P, Tyson GW (2015). CheckM: assessing the quality of microbial genomes recovered from isolates, single cells, and metagenomes*. Genome Res.*, 25(7):1043-55.

Appendix

	Time elapsed (h:m:s)	Maximum memory usage(KB)	Number of contigs	Number of clusters	Recall	Precision	NMI	RI	ARI
CONCOCT	0:07:33	498,460	42,054	89	0.9708	0.8833	0.9523	0.9964	0.8724
COCACOLA	0:47:38	609,820	42,054	79	0.8034	0.6975	0.7904	0.9882	0.6294
MetaBAT	1:35:49	481,824	39,618	61	0.9134	0.6282	0.8314	0.9680	0.4053
MaxBin2	0:42:56	386,016	41,253	93	0.8669	0.8211	0.8881	0.9928	0.7421
GroopM	2:28:54	21,723,748	39,074	121	0.6729	0.5515	0.6874	0.9688	0.3273
BMC3C	0:26:37	844,076	42,036	66	0.9904	0.7337	0.8933	0.9630	0.3969
MyCC	1:14:44	5,316,372	42,054	92	0.9211	0.8459	0.9141	0.9906	0.6993
GATTACA	09:54.2	902,904	42,054	72	0.9661	0.7615	0.9120	0.9869	0.6481

Table A.1. Elapsed time, maximum memory usage, number of clustered contigs, the number of inferred clusters, and performance score of Species_20S_2.5R

	Time elapsed (h:m:s)	Maximum memory usage (KB)	Number of contigs	Number of clusters	Recall	Precision	NMI	RI	ARI
CONCOCT	0:07:32	459,196	37,936	88	0.9777	0.8688	0.9520	0.9938	0.7922
COCACOLA	0:51:46	567,964	37,936	94	0.8832	0.8236	0.8839	0.9932	0.7513
MetaBAT	1:05:04	529,480	35,779	92	0.8961	0.7844	0.8963	0.9896	0.6665
MaxBin2	0:45:36	429,016	37,651	104	0.9044	0.8846	0.9228	0.9955	0.8211
GroopM	9:47:14	98,012,376	35,929	104	0.7691	0.5604	0.7480	0.9601	0.2808
BMC3C	0:27:35	850,652	37,911	64	0.9972	0.6676	0.8662	0.9343	0.2602
MyCC	1:08:07	3,907,160	37,936	118	0.9067	0.9237	0.9397	0.9955	0.8215
GATTACA	0:07:13	812,936	37,936	70	0.9762	0.7310	0.9104	0.9836	0.5879

Table A.2. Elapsed time, maximum memory usage, number of clustered contigs, the number of inferred clusters, and performance score of Species_20S_5.0R

	Time elapsed (h:m:s)	Maximum memory usage (KB)	Number of contigs	Number of clusters	Recall	Precision	NMI	RI	ARI
CONCOCT	0:03:11	421,464	33,788	87	0.9846	0.8351	0.9463	0.9921	0.7575
COCACOLA	0:51:59	526,104	33,788	90	0.9367	0.8701	0.9276	0.9953	0.8300
MetaBAT	0:44:50	517,960	31,895	99	0.9470	0.8552	0.9372	0.9929	0.7647
MaxBin2	0:39:48	428,632	33,575	105	0.9305	0.9241	0.9443	0.9969	0.8773
GroopM	4:44:14	40,575,764	32,935	129	0.7959	0.6980	0.8123	0.9805	0.4735
BMC3C	0:25:26	850,452	33,769	64	0.9966	0.6601	0.8672	0.9328	0.2611
MyCC	1:04:49	4,154,840	33,788	121	0.9086	0.9384	0.9472	0.9957	0.8296
GATTACA	0:05:27	734,916	33,788	67	0.9799	0.7031	0.9046	0.9811	0.5615

Table A.3. Elapsed time, maximum memory usage, number of clustered contigs, the number of inferred clusters, and performance score of Species_20S_7.5R

	Time elapsed (h:m:s)	Maximum memory usage(KB)	Number of contigs	Number of clusters	Recall	Precision	NMI	RI	ARI
CONCOCT	0:04:11	421,908	32,354	87	0.9866	0.8556	0.9525	0.9929	0.7775
COCACOLA	0:52:12	507,280	32,354	89	0.9626	0.9082	0.9552	0.9968	0.8821
MetaBAT	0:39:56	507,156	30,627	103	0.9527	0.9561	0.9618	0.9978	0.9139
MaxBin2	0:41:07	425,436	32,248	103	0.9452	0.9286	0.9509	0.9972	0.8921
GroopM	10:46:33	84,780,256	31,888	139	0.7798	0.6117	0.7900	0.9759	0.4013
BMC3C	0:24:05	851,348	32,331	63	0.9997	0.6305	0.8538	0.9065	0.1990
MyCC	1:08:19	4,795,728	32,354	121	0.9089	0.9473	0.9539	0.9962	0.8475
GATTACA	0:04:55	709,216	32,354	67	0.9826	0.7233	0.9164	0.9868	0.6523

Table A.4. Elapsed time, maximum memory usage, number of clustered contigs, the number of inferred clusters, and performance score of Species_20S_10R

	Time elapsed (h:m:s)	Maximum memory usage(KB)	Number of contigs	Number of clusters	Recall	Precision	NMI	RI	ARI
CONCOCT	0:03:45	439,680	32,417	87	0.9908	0.8620	0.9530	0.9913	0.7425
COCACOLA	0:47:08	510,388	32,417	93	0.9634	0.9265	0.9637	0.9973	0.8969
MetaBAT	0:39:24	506,816	30,601	107	0.9670	0.9748	0.9714	0.9986	0.9457
MaxBin2	0:41:04	424,748	32,342	98	0.9548	0.9257	0.9544	0.9973	0.8971
GroopM	8:59:54	58,097,104	31,507	125	0.7782	0.6765	0.8168	0.9851	0.5388
BMC3C	0:24:11	856,672	32,390	120	0.9710	0.6040	0.8556	0.9406	0.2783
MyCC	1:06:56	5,035,840	32,417	61	0.9060	0.9510	0.9558	0.9967	0.8673
GATTACA	0:05:27	711,132	32,417	67	0.9865	0.6934	0.9082	0.9823	0.5821

Table A.5. Elapsed time, maximum memory usage, number of clustered contigs, the number of inferred clusters, and performance score of Species_20S_12.5R

	Time elapsed (h:m:s)	Maximum memory usage(KB)	Number of contigs	Number of clusters	Recall	Precision	NMI	RI	ARI
CONCOCT	0:03:36	472,012	32,530	78	0.9863	0.7810	0.9280	0.9844	0.6131
COCACOLA	0:46:45	557,732	32,530	91	0.9927	0.9530	0.9828	0.9989	0.9582
MetaBAT	0:39:56	522,228	30,724	103	0.9714	0.9737	0.9743	0.9987	0.9505
MaxBin2	0:56:05	436,864	32,491	100	0.9717	0.9489	0.9686	0.9982	0.9295
GroopM	19:35:05	110,274,824	32,503	165	0.7362	0.6386	0.7854	0.9810	0.4591
BMC3C	0:25:19	857,312	32,510	75	0.9993	0.7797	0.9354	0.9784	0.5380
MyCC	1:09:35	5,022,184	32,530	133	0.8699	0.9686	0.9526	0.9962	0.8408
GATTACA	0:06:17	718,264	32,530	60	0.9842	0.6289	0.8749	0.9662	0.4159

Table A.6. Elapsed time, maximum memory usage, number of clustered contigs, the number of inferred clusters, and performance score of Species_40S_12.5R

	Time elapsed (h:m:s)	Maximum memory usage(KB)	Number of contigs	Number of clusters	Recall	Precision	NMI	RI	ARI
CONCOCT	0:06:45	494,464	32,793	76	0.9853	0.8019	0.9271	0.9808	0.5601
COCACOLA	0:57:29	600,584	32,793	90	0.9622	0.8926	0.9549	0.9963	0.8651
MetaBAT	0:38:45	513,908	29,340	86	0.9211	0.7158	0.8865	0.9853	0.6051
MaxBin2	0:58:14	446,632	32,662	103	0.9522	0.9347	0.9564	0.9976	0.9063
GroopM	6:51:45	49,156,568	32,069	160	0.7202	0.6682	0.7869	0.9858	0.5278
BMC3C	0:27:37	857,304	32,772	68	0.9991	0.7332	0.9091	0.9640	0.4067
MyCC	1:08:01	4,926,760	32,793	104	0.9774	0.9721	0.9762	0.9987	0.9509
GATTACA	0:09:33	767,180	32,793	67	0.9762	0.7030	0.8846	0.9641	0.3986

Table A.7. Elapsed time, maximum memory usage, number of clustered contigs, the number of inferred clusters, and performance score of Species_60S_2.5R

	Time elapsed (h:m:s)	Maximum memory usage(KB)	Number of contigs	Number of clusters	Recall	Precision	NMI	RI	ARI
CONCOCT	0:07:13	523,472	32,503	76	0.9909	0.7918	0.9258	0.9794	0.5455
COCACOLA	0:51:39	607,324	32,503	93	0.9853	0.9450	0.9784	0.9985	0.9429
MetaBAT	0:38:20	520,264	30,373	97	0.9361	0.8501	0.9335	0.9930	0.7631
MaxBin2	1:06:47	446,772	32,452	98	0.9736	0.9614	0.9703	0.9986	0.9466
GroopM	3:12:22	29,588,576	30,513	119	0.7858	0.7262	0.8349	0.9861	0.5701
BMC3C	0:25:30	856,720	32,480	73	0.9651	0.7680	0.9151	0.9746	0.4797
MyCC	1:10:40	5,307,580	32,503	115	0.9423	0.9913	0.9771	0.9987	0.9480
GATTACA	0:08:04	775,904	32,503	65	0.9881	0.6849	0.8860	0.9653	0.4112

Table A.8. Elapsed time, maximum memory usage, number of clustered contigs, the number of inferred clusters, and performance score of Species_60S_5.0R

	Time elapsed (h:m:s)	Maximum memory usage (KB)	Number of contigs	Number of clusters	Recall	Precision	NMI	RI	ARI
CONCOCT	0:02:41	508,468	32,531	74	0.9919	0.7690	0.9196	0.9767	0.5146
COCACOLA	0:58:15	604,668	32,531	95	0.9873	0.9460	0.9810	0.9984	0.9403
MetaBAT	0:41:23	526,200	30,815	103	0.9563	0.9416	0.9607	0.9976	0.9082
MaxBin2	0:34:43	447,004	32,501	97	0.9794	0.9635	0.9751	0.9988	0.9522
GroopM	13:00:57	118,117,588	31,258	112	0.7999	0.6504	0.8109	0.9750	0.4192
BMC3C	0:24:56	861,180	32,501	63	0.9553	0.7178	0.8875	0.9682	0.4280
MyCC	1:08:56	5,205,540	32,531	124	0.9028	0.9786	0.9668	0.9973	0.8903
GATTACA	0:07:18	767,516	32,531	61	0.9909	0.6497	0.8707	0.9561	0.3552

Table A.9. Elapsed time, maximum memory usage, number of clustered contigs, the number of inferred clusters, and performance score of Species_60S_7.5R

	Time elapsed (h:m:s)	Maximum memory usage(KB)	Number of contigs	Number of clusters	Recall	Precision	NMI	RI	ARI
CONCOCT	0:03:11	476,200	32,532	77	0.9891	0.8068	0.9331	0.9847	0.6172
COCACOLA	0:48:35	597,276	32,532	91	0.9939	0.9384	0.9815	0.9985	0.9435
MetaBAT	0:38:30	527,344	30,793	104	0.9738	0.9787	0.9758	0.9988	0.9545
MaxBin2	0:30:05	446,880	32,507	97	0.9817	0.9657	0.9774	0.9989	0.9563
GroopM	9:40:22	75,100,172	31,531	142	0.8248	0.7296	0.8477	0.9846	0.5502
BMC3C	0:25:20	856,736	32,513	75	0.9991	0.7977	0.9440	0.9862	0.6472
MyCC	1:10:10	5,235,516	32,532	129	0.8905	0.9925	0.9669	0.9976	0.8997
GATTACA	0:07:06	766,620	32,532	59	0.9889	0.6147	0.8566	0.9519	0.3334

Table A.10. Elapsed time, maximum memory usage, number of clustered contigs, the number of inferred clusters, and performance score of Species_60S_10.0R

	Time elapsed (h:m:s)	Maximum memory usage(KB)	Number of contigs	Number of clusters	Recall	Precision	NMI	RI	ARI
CONCOCT	0:03:20	526,384	32,543	77	0.9945	0.7779	0.9204	0.9750	0.4977
COCACOLA	0:56:01	591,404	32,543	93	0.9901	0.9342	0.9808	0.9979	0.9237
MetaBAT	0:39:14	526,880	30,831	104	0.9841	0.9879	0.9851	0.9993	0.9747
MaxBin2	0:46:17	447,052	32,530	97	0.9822	0.9637	0.9778	0.9989	0.9569
GroopM	8:47:14	73,535,852	30,561	127	0.8383	0.7720	0.8660	0.9889	0.6471
BMC3C	0:25:22	856,232	32,519	72	0.9446	0.7598	0.9103	0.9796	0.5350
MyCC	1:09:11	5,123,824	32,543	130	0.8862	0.9739	0.9630	0.9970	0.8764
GATTACA	0:07:03	767,324	32,543	58	0.9895	0.6172	0.8557	0.9507	0.3273

Table A.11. Elapsed time, maximum memory usage, number of clustered contigs, the number of inferred clusters, and performance score of Species_60S_12.5R

	Time elapsed (h:m:s)	Maximum memory usage(KB)	Number of contigs	Number of clusters	Recall	Precision	NMI	RI	ARI
CONCOCT	0:03:46	554,460	32,565	74	0.9897	0.7584	0.9148	0.9751	0.4971
COCACOLA	0:46:59	646,684	32,565	96	0.9841	0.9609	0.9851	0.9988	0.9538
MetaBAT	1:05:57	532,448	30,770	107	0.9827	0.9908	0.9855	0.9994	0.9756
MaxBin2	0:59:19	457,968	32,548	98	0.9869	0.9784	0.9840	0.9993	0.9723
GroopM	18:44:00	103,101,036	32,167	152	0.8346	0.7231	0.8504	0.9858	0.5728
BMC3C	0:21:00	859,412	32,539	73	0.9954	0.7595	0.9400	0.9873	0.6644
MyCC	3:26:05	5,615,084	32,565	68	0.9670	0.5456	0.8281	0.9310	0.2470
GATTACA	0:07:41	822,932	32,565	59	0.9929	0.6280	0.8654	0.9535	0.3426

Table A.12. Elapsed time, maximum memory usage, number of clustered contigs, the number of inferred clusters, and performance score of Species_80S_12.5R

	Time elapsed (h:m:s)	Maximum memory usage(KB)	Number of contigs	Number of clusters	Recall	Precision	NMI	RI	ARI
CONCOCT	0:04:22	524,676	32,643	71	0.9925	0.7041	0.8912	0.9602	0.3789
COCACOLA	0:55:16	677,000	32,643	92	0.9886	0.9407	0.9829	0.9982	0.9322
MetaBAT	1:04:23	539,748	30,934	101	0.9862	0.9873	0.9858	0.9993	0.9741
MaxBin2	1:48:04	467,612	32,614	101	0.9834	0.9609	0.9799	0.9987	0.9497
GroopM	16:02:41	106,297,332	31,992	128	0.8198	0.7113	0.8408	0.9831	0.5285
BMC3C	0:21:13	848,636	32,619	71	0.9814	0.7516	0.9251	0.9812	0.5653
MyCC	1:11:32	5,343,348	32,643	113	0.9571	0.9812	0.9808	0.9986	0.9437
GATTACA	0:09:02	836,936	32,643	58	0.9907	0.6210	0.8630	0.9581	0.3664

Table A.13. Elapsed time, maximum memory usage, number of clustered contigs, the number of inferred clusters, and performance score of Species_100S_12.5R

	Time elapsed (h:m:s)	Maximum memory usage(KB)	Number of contigs	Number of clusters	Recall	Precision	NMI	RI	ARI
CONCOCT	0:45:43	584,236	47,741	98	0.9643	0.9025	0.9497	0.9969	0.8909
COCACOLA	0:38:45	655,288	47,741	58	0.7159	0.5769	0.6896	0.9767	0.4357
MetaBAT	2:19:14	393,808	44,872	44	0.9004	0.5116	0.7547	0.9383	0.2564
MaxBin2	0:36:30	308,044	46,910	73	0.8089	0.7308	0.8289	0.9896	0.6581
GroopM	1:13:54	8,950,308	45,254	118	0.6547	0.5403	0.6731	0.9727	0.3774
BMC3C	2:00:26	834,120	47,713	287	0.5077	0.9320	0.8449	0.9902	0.4984
MyCC	3:50:15	8,305,344	47,741	65	0.9298	0.6278	0.8042	0.9256	0.2263
GATTACA	0:15:13	1,019,784	47,741	66	0.9634	0.7377	0.9002	0.9867	0.6566

Table A.14. Elapsed time, maximum memory usage, number of clustered contigs, the number of inferred clusters, and performance score of Strains_20S_2.5R

	Time elapsed (h:m:s)	Maximum memory usage (KB)	Number of contigs	Number of clusters	Recall	Precision	NMI	RI	ARI
CONCOCT	0:27:26	650,572	57,488	98	0.9400	0.9159	0.9480	0.9965	0.8855
COCACOLA	0:54:48	780,952	57,488	66	0.7671	0.6178	0.7300	0.9745	0.4665
MetaBAT	2:33:12	507,228	53,866	57	0.8980	0.6473	0.8150	0.9690	0.4526
MaxBin2	1:25:07	392,724	56,389	90	0.8136	0.7793	0.8520	0.9910	0.7159
GroopM	4:55:16	14,094,272	56,505	135	0.6689	0.5867	0.6943	0.9737	0.4191
BMC3C	2:20:30	837,800	57,453	281	0.5410	0.9620	0.8573	0.9893	0.4865
MyCC	4:20:58	12,365,556	57,488	55	0.9780	0.6700	0.8266	0.9204	0.2532
GATTACA	0:17:58	1,202,168	57,488	65	0.9379	0.7343	0.8957	0.9887	0.7042

Table A.15. Elapsed time, maximum memory usage, number of clustered contigs, the number of inferred clusters, and performance score of Strains_20S_5.0R

	Time elapsed (h:m:s)	Maximum memory usage(KB)	Number of contigs	Number of clusters	Recall	Precision	NMI	RI	ARI
CONCOCT	0:20:44	646,248	56,518	133	0.9038	0.9193	0.9482	0.9960	0.8407
COCACOLA	1:12:11	774,952	56,518	95	0.8025	0.7664	0.8306	0.9891	0.6167
MetaBAT	3:54:05	642,196	53,054	83	0.9130	0.7437	0.8860	0.9836	0.5736
MaxBin2	2:14:15	516,968	55,805	112	0.8197	0.8677	0.8872	0.9936	0.7350
GroopM	6:07:05	53,618,052	51,756	136	0.7060	0.5579	0.7089	0.9690	0.3261
BMC3C	1:55:00	845,784	56,485	195	0.8301	0.9691	0.9404	0.9956	0.8103
MyCC	3:16:18	10,664,600	56,518	88	0.9067	0.6680	0.8441	0.9511	0.2937
GATTACA	0:55:05	1,185,900	56,518	90	0.8948	0.6844	0.8837	0.9840	0.5614

Table A.16. Elapsed time, maximum memory usage, number of clustered contigs, the number of inferred clusters, and performance score of Strains_20S_7.5R

	Time elapsed (h:m:s)	Maximum memory usage (KB)	Number of contigs	Number of clusters	Recall	Precision	NMI	RI	ARI
CONCOCT	0:21:12	714,752	64,387	122	0.8632	0.8571	0.9167	0.9892	0.6501
COCACOLA	1:16:17	841,240	64,387	93	0.7505	0.7361	0.8085	0.9885	0.5898
MetaBAT	0:17:06	683,116	60,134	92	0.8868	0.8130	0.8956	0.9901	0.6941
MaxBin2	2:09:22	549,324	63,868	104	0.8173	0.8446	0.8852	0.9919	0.6930
GroopM	10:24:31	106,192,656	57,829	133	0.6664	0.5633	0.7081	0.9663	0.3012
BMC3C	1:57:35	845,072	64,341	305	0.6449	0.9800	0.8943	0.9926	0.6363
MyCC	3:28:09	15,033,436	64,387	89	0.9310	0.6964	0.8412	0.9220	0.2157
GATTACA	0:20:26	1,331,896	64,387	89	0.8636	0.7149	0.8754	0.9818	0.5213

Table A.17. Elapsed time, maximum memory usage, number of clustered contigs, the number of inferred clusters, and performance score of Strains_20S_10.0R

	Time elapsed (h:m:s)	Maximum memory usage(KB)	Number of contigs	Number of clusters	Recall	Precision	NMI	RI	ARI
CONCOCT	0:20:06	666,972	60,692	117	0.8891	0.9164	0.9411	0.9950	0.8021
COCACOLA	1:09:26	799,620	60,692	89	0.8255	0.7692	0.8518	0.9892	0.6346
MetaBAT	3:49:06	662,560	57,259	88	0.9304	0.7973	0.9014	0.9860	0.6181
MaxBin2	2:10:04	531,856	59,994	112	0.8656	0.8801	0.9044	0.9942	0.7705
GroopM	4:38:34	34,803,972	57,839	116	0.7309	0.5554	0.7218	0.9679	0.3220
BMC3C	1:51:10	843,748	60,663	220	0.7675	0.9830	0.9269	0.9948	0.7562
MyCC	6:06:14	12,723,324	60,692	97	0.9257	0.7255	0.8577	0.9331	0.2377
GATTACA	0:16:35	1,255,980	60,692	83	0.8975	0.7077	0.8857	0.9868	0.6159

Table A.18. Elapsed time, maximum memory usage, number of clustered contigs, the number of inferred clusters, and performance score of Strains_20S_12.5R

	Time elapsed (h:m:s)	Maximum memory usage(KB)	Number of contigs	Number of clusters	Recall	Precision	NMI	RI	ARI
CONCOCT	1:12:00	832,948	68,846	122	0.8219	0.8694	0.9151	0.9884	0.6206
COCACOLA	1:35:14	989,652	68,846	98	0.7662	0.7665	0.8525	0.9885	0.5975
MetaBAT	4:03:01	763,400	63,715	100	0.8944	0.8643	0.9138	0.9926	0.7578
MaxBin2	3:32:43	630,664	68,564	114	0.8375	0.8948	0.9105	0.9940	0.7728
GroopM	6:44:31	36,558,064	65,228	133	0.6743	0.5381	0.6961	0.9702	0.3262
BMC3C	2:21:47	851,828	68,784	249	0.7126	0.9959	0.9215	0.9933	0.6896
MyCC	6:37:14	16,612,512	68,846	102	0.9124	0.6939	0.8464	0.9136	0.2063
GATTACA	0:51:24	1,460,400	68,846	94	0.8465	0.7328	0.8815	0.9817	0.5238

Table A.19. Elapsed time, maximum memory usage, number of clustered contigs, the number of inferred clusters, and performance score of Strains_40S_12.5R

	Time elapsed (h:m:s)	Maximum memory usage(KB)	Number of contigs	Number of clusters	Recall	Precision	NMI	RI	ARI
CONCOCT	2:19:27	787,856	62,556	89	0.8523	0.8176	0.9017	0.9862	0.5981
COCACOLA	2:44:36	982,660	62,556	90	0.7416	0.7338	0.7994	0.9875	0.5783
MetaBAT	4:41:04	660,020	54,741	59	0.8628	0.5214	0.7646	0.9505	0.3051
MaxBin2	1:56:04	575,780	61,439	121	0.8094	0.8753	0.8909	0.9935	0.7503
GroopM	11:23:09	77,905,724	58,173	148	0.6368	0.5430	0.6762	0.9715	0.3482
BMC3C	1:12:25	847,552	62,515	299	0.6127	0.9835	0.8917	0.9918	0.5899
MyCC	5:12:13	15,293,376	62,556	77	0.9353	0.6744	0.8211	0.8820	0.1525
GATTACA	1:48:00	1,433,420	62,556	73	0.8685	0.7066	0.8752	0.9808	0.5188

Table A.20. Elapsed time, maximum memory usage, number of clustered contigs, the number of inferred clusters, and performance score of Strains_60S_2.5R

	Time elapsed (h:m:s)	Maximum memory usage (KB)	Number of contigs	Number of clusters	Recall	Precision	NMI	RI	ARI
CONCOCT	0:24:34	805,348	56,318	105	0.8954	0.8564	0.9282	0.9890	0.6632
COCACOLA	1:23:40	901,980	56,318	97	0.8259	0.7442	0.8653	0.9872	0.5957
MetaBAT	1:57:07	679,108	52,968	90	0.8977	0.7772	0.8942	0.9888	0.6587
MaxBin2	1:49:21	578,544	56,050	110	0.8406	0.8704	0.8989	0.9931	0.7319
GroopM	16:37:33	122,267,376	50,013	117	0.6953	0.5556	0.7208	0.9732	0.3521
BMC3C	1:32:12	847,820	56,278	258	0.7339	0.9960	0.9263	0.9943	0.7252
MyCC	6:17:06	10,540,628	56,318	101	0.9279	0.8014	0.9163	0.9792	0.5237
GATTACA	1:04:32	1,302,920	56,318	86	0.9016	0.7351	0.8958	0.9813	0.5375

Table A.21. Elapsed time, maximum memory usage, number of clustered contigs, the number of inferred clusters, and performance score of Strains_60S_5.0R

	Time elapsed (h:m:s)	Maximum memory usage (KB)	Number of contigs	Number of clusters	Recall	Precision	NMI	RI	ARI
CONCOCT	0:12:26	826,136	58,131	96	0.9092	0.8538	0.9339	0.9892	0.6788
COCACOLA	1:17:38	938,036	58,131	91	0.8171	0.7385	0.8583	0.9827	0.5249
MetaBAT	2:13:06	680,668	54,539	88	0.9026	0.7717	0.8837	0.9856	0.6262
MaxBin2	2:13:06	680,668	57,936	108	0.8605	0.9163	0.9248	0.9945	0.7946
GroopM	14:26:12	87,467,524	52,787	129	0.7205	0.6448	0.7798	0.9766	0.4063
BMC3C	2:20:46	843,652	58,093	251	0.7246	0.9884	0.9223	0.9931	0.6885
MyCC	4:21:59	11,322,780	58,131	94	0.9419	0.8021	0.9186	0.9731	0.4711
GATTACA	0:57:05	1,290,008	58,131	71	0.9215	0.6935	0.8766	0.9709	0.4388

Table A.22. Elapsed time, maximum memory usage, number of clustered contigs, the number of inferred clusters, and performance score of Strains_60S_7.5R

	Time elapsed (h:m:s)	Maximum memory usage (KB)	Number of contigs	Number of clusters	Recall	Precision	NMI	RI	ARI
CONCOCT	0:43:14	872,468	69,490	100	0.8342	0.8346	0.9092	0.9839	0.5578
COCACOLA	2:29:25	1,095,236	69,490	94	0.7558	0.7329	0.8482	0.9836	0.5146
MetaBAT	2:36:16	790,484	64,146	89	0.9058	0.7860	0.8892	0.9874	0.6640
MaxBin2	2:21:18	664,544	69,205	125	0.8276	0.9081	0.9127	0.9933	0.7540
GroopM	17:22:05	112,097,996	63,651	149	0.6687	0.6174	0.7428	0.9792	0.4005
BMC3C	2:34:11	844,604	69,432	191	0.7608	0.9931	0.9394	0.9941	0.7474
MyCC	4:11:27	17,472,192	69,490	104	0.9270	0.7821	0.9147	0.9773	0.5313
GATTACA	1:18:55	1,543,632	69,490	80	0.8708	0.7457	0.8895	0.9809	0.5351

Table A.23. Elapsed time, maximum memory usage, number of clustered contigs, the number of inferred clusters, and performance score of Strains_60S_10.0R

	Time elapsed (h:m:s)	Maximum memory usage (KB)	Number of contigs	Number of clusters	Recall	Precision	NMI	RI	ARI
CONCOCT	0:23:45	860,680	68,181	113	0.8606	0.9035	0.9399	0.9928	0.7483
COCACOLA	2:11:33	1,083,744	68,181	94	0.7682	0.7385	0.8543	0.9862	0.5597
MetaBAT	2:30:45	775,304	63,653	94	0.9337	0.7916	0.9049	0.9843	0.6160
MaxBin2	1:33:17	657,200	67,947	115	0.8502	0.9192	0.9242	0.9942	0.7848
GroopM	13:12:07	109,799,712	61,994	124	0.6727	0.5592	0.7277	0.9747	0.3480
BMC3C	3:22:13	1,083,744	68,134	202	0.7595	0.9970	0.9390	0.9940	0.7411
MyCC	6:26:15	16,938,388	68,181	102	0.9398	0.7579	0.8902	0.9443	0.3076
GATTACA	0:33:32	1,521,388	68,181	94	0.8556	0.7960	0.9044	0.9874	0.6254

Table A.24. Elapsed time, maximum memory usage, number of clustered contigs, the number of inferred clusters, and performance score of Strains_60S_12.5R

	Time elapsed (h:m:s)	Maximum memory usage (KB)	Number of contigs	Number of clusters	Recall	Precision	NMI	RI	ARI
CONCOCT	0:42:40	1,156,616	62,785	104	0.8692	0.8521	0.9184	0.9855	0.5977
COCACOLA	1:37:07	1,060,516	62,785	98	0.7933	0.7473	0.8717	0.9835	0.5351
MetaBAT	2:19:56	753,260	58,589	107	0.8862	0.9079	0.9336	0.9933	0.7733
MaxBin2	2:25:08	652,200	62,710	107	0.8524	0.9222	0.9293	0.9940	0.7799
GroopM	12:25:38	98,253,532	60,550	128	0.7288	0.5814	0.7400	0.9630	0.2930
BMC3C	0:38:26	847,248	62,737	200	0.7783	0.9946	0.9430	0.9943	0.7625
MyCC	2:52:55	13,289,716	62,785	113	0.9180	0.8967	0.9458	0.9930	0.7822
GATTACA	0:34:37	1,442,532	62,785	82	0.8783	0.7287	0.8795	0.9722	0.4379

Table A.25. Elapsed time, maximum memory usage, number of clustered contigs, the number of inferred clusters, and performance score of Strains_80S_12.5R

	Time elapsed (h:m:s)	Maximum memory usage (KB)	Number of contigs	Number of clusters	Recall	Precision	NMI	RI	ARI
CONCOCT	0:20:23	909,860	59,981	91	0.8637	0.8209	0.9085	0.9810	0.5331
COCACOLA	1:34:25	1,076,388	59,981	91	0.8025	0.7501	0.8788	0.9835	0.5372
MetaBAT	0:07:22	750,120	54,681	104	0.8974	0.9127	0.9339	0.9940	0.8072
MaxBin2	4:46:47	647,404	59,866	99	0.8611	0.9290	0.9343	0.9941	0.7926
GroopM	12:55:56	50,478,240	58,512	176	0.6754	0.5906	0.7370	0.9775	0.3838
BMC3C	0:39:42	845,864	59,960	217	0.7344	0.9950	0.9334	0.9932	0.7144
MyCC	5:10:05	11,927,152	59,981	119	0.9149	0.8705	0.9398	0.9883	0.6899
GATTACA	0:34:49	1,440,636	59,981	81	0.8515	0.7568	0.8656	0.9532	0.3048

Table A.26. Elapsed time, maximum memory usage, number of clustered contigs, the number of inferred clusters, and performance score of Strains_100S_12.5R

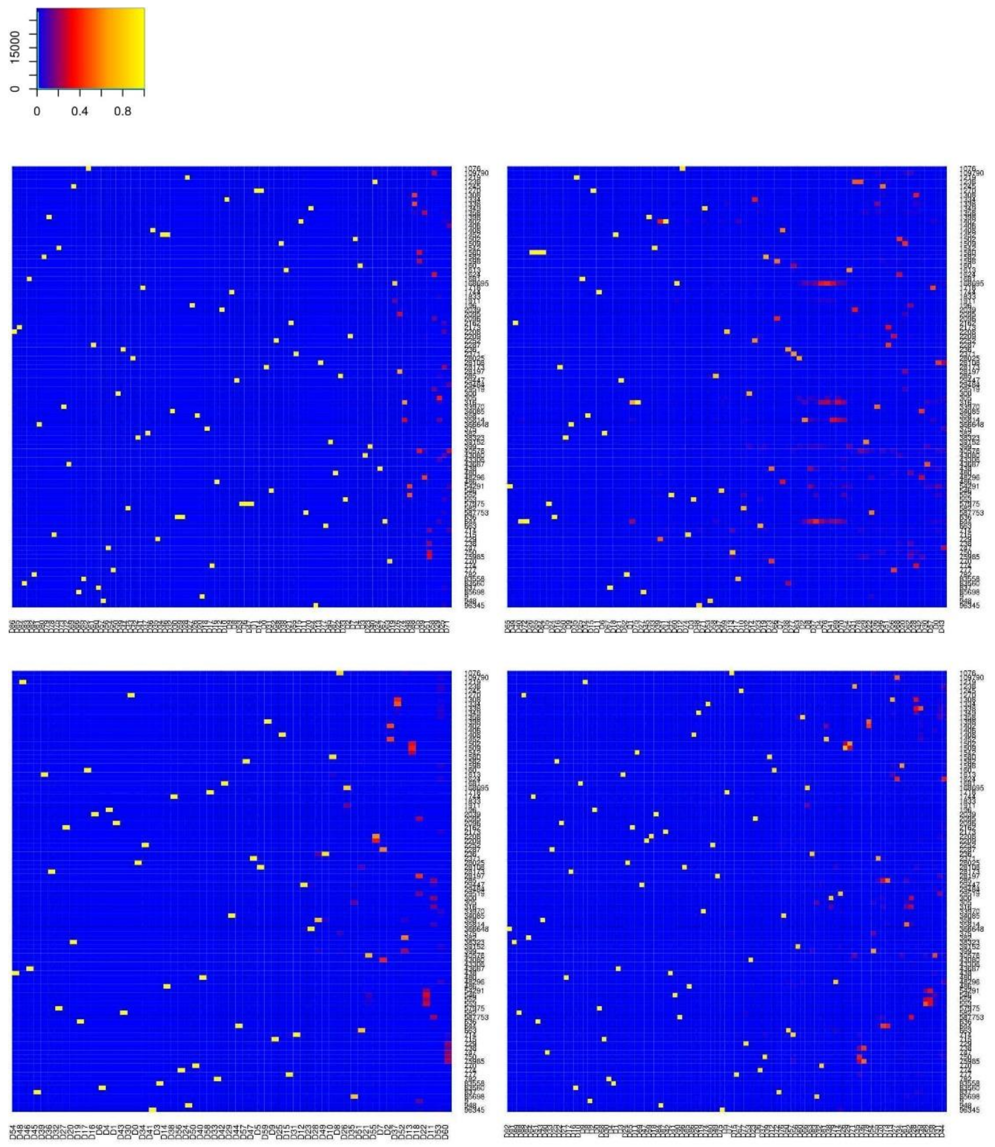


Figure A.1. Heat maps for Speceis_20S_2.5R. The top left is that of CONCOCT, top right of COCACOLA, bottom left of MetaBAT, and bottom right of MaxBin2

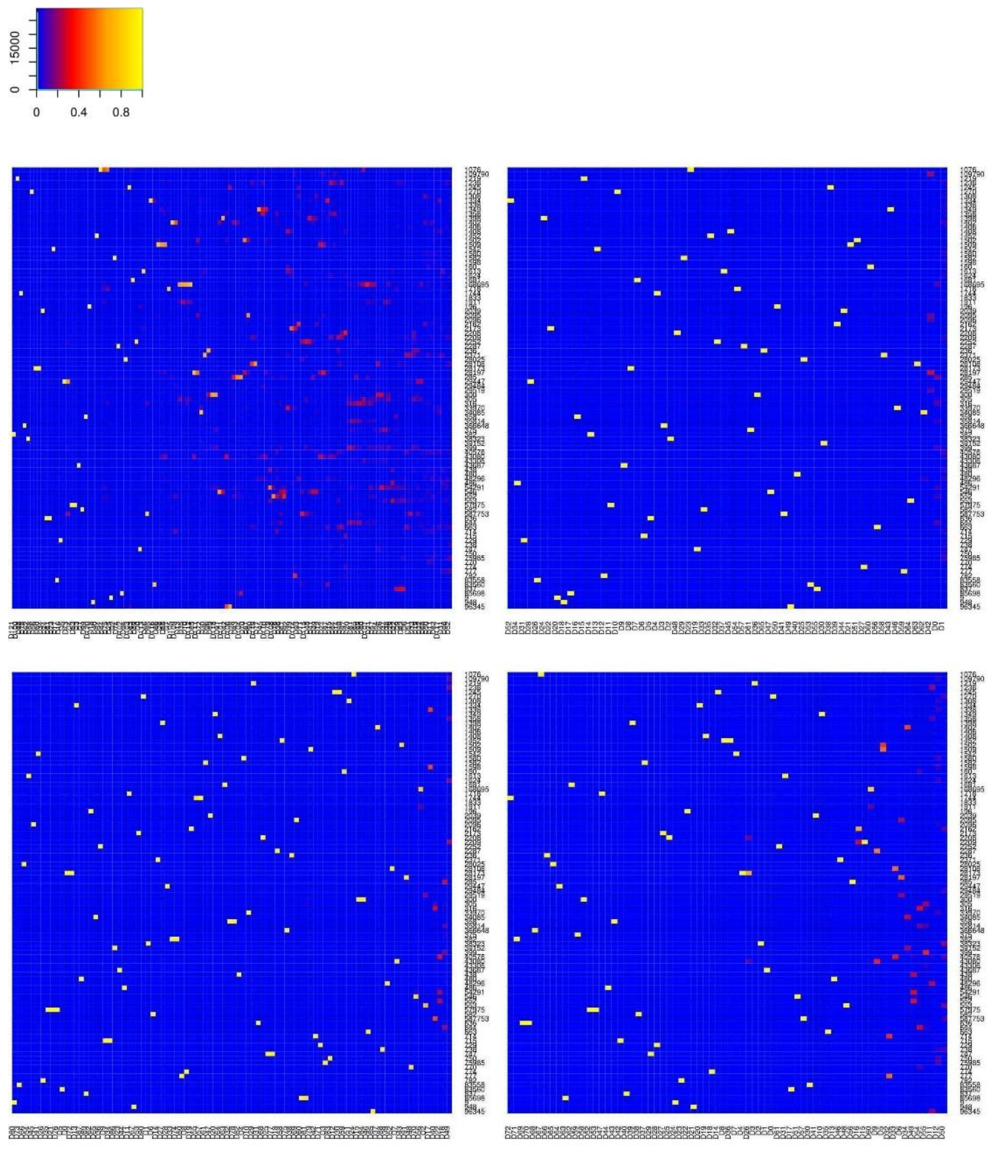


Figure A.2. Heat maps for Speceis_20S_2.5R. The top left is that of GroopM, top right of BMC3C, bottom left of MyCC, and bottom right of GATTACA

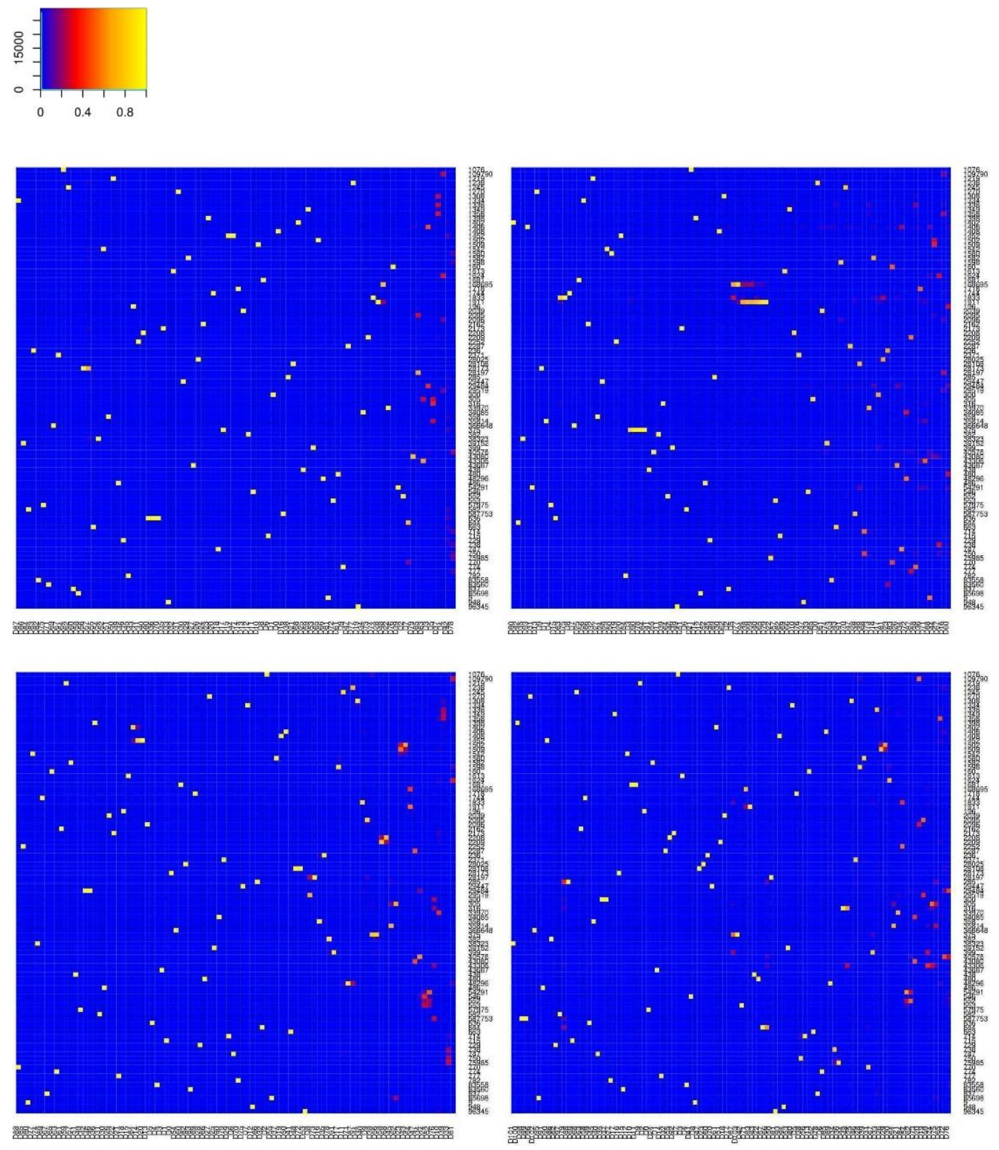


Figure A.3. Heat maps for Speceis_20S_5.0R. The top left is that of CONCOCT, top right of COCACOLA, bottom left of MetaBAT, and bottom right of MaxBin2

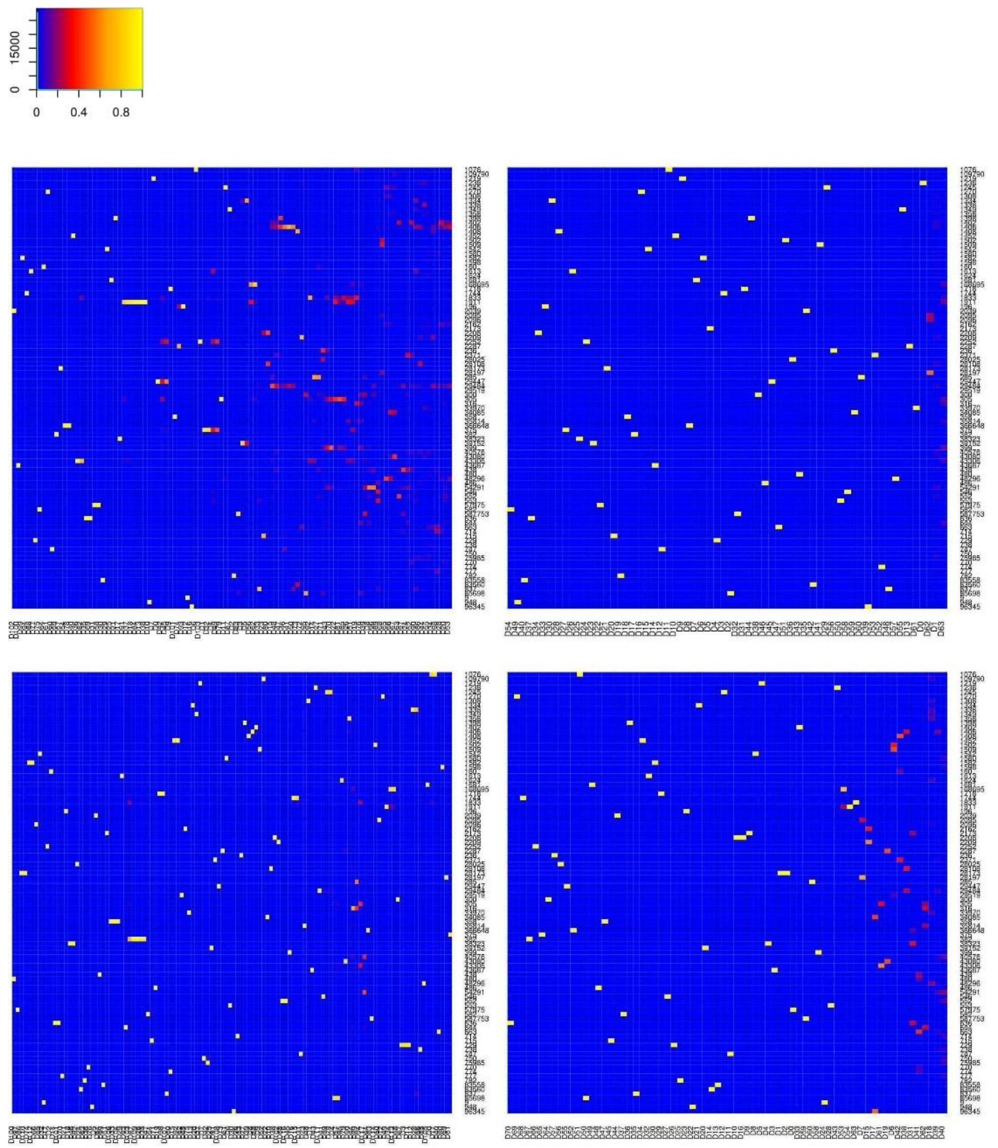


Figure A.4. Heat maps for Speceis_20S_5.0R. The top left is that of GroopM, top right of BMC3C, bottom left of MyCC, and bottom right of GATTACA

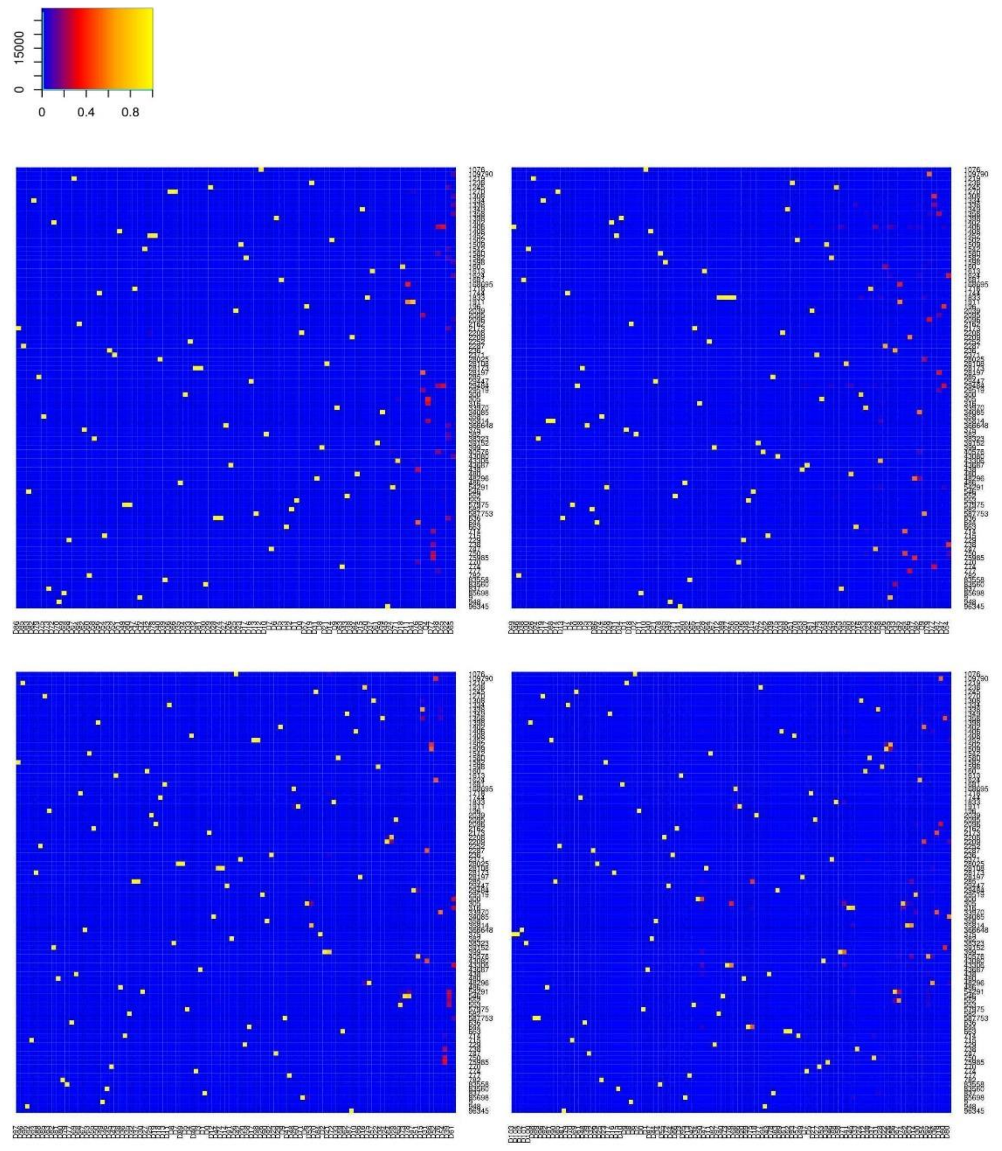


Figure A.5. Heat maps for Speceis_20S_7.5R. The top left is that of CONCOCT, top right of COCACOLA, bottom left of MetaBAT, and bottom right of MaxBin2

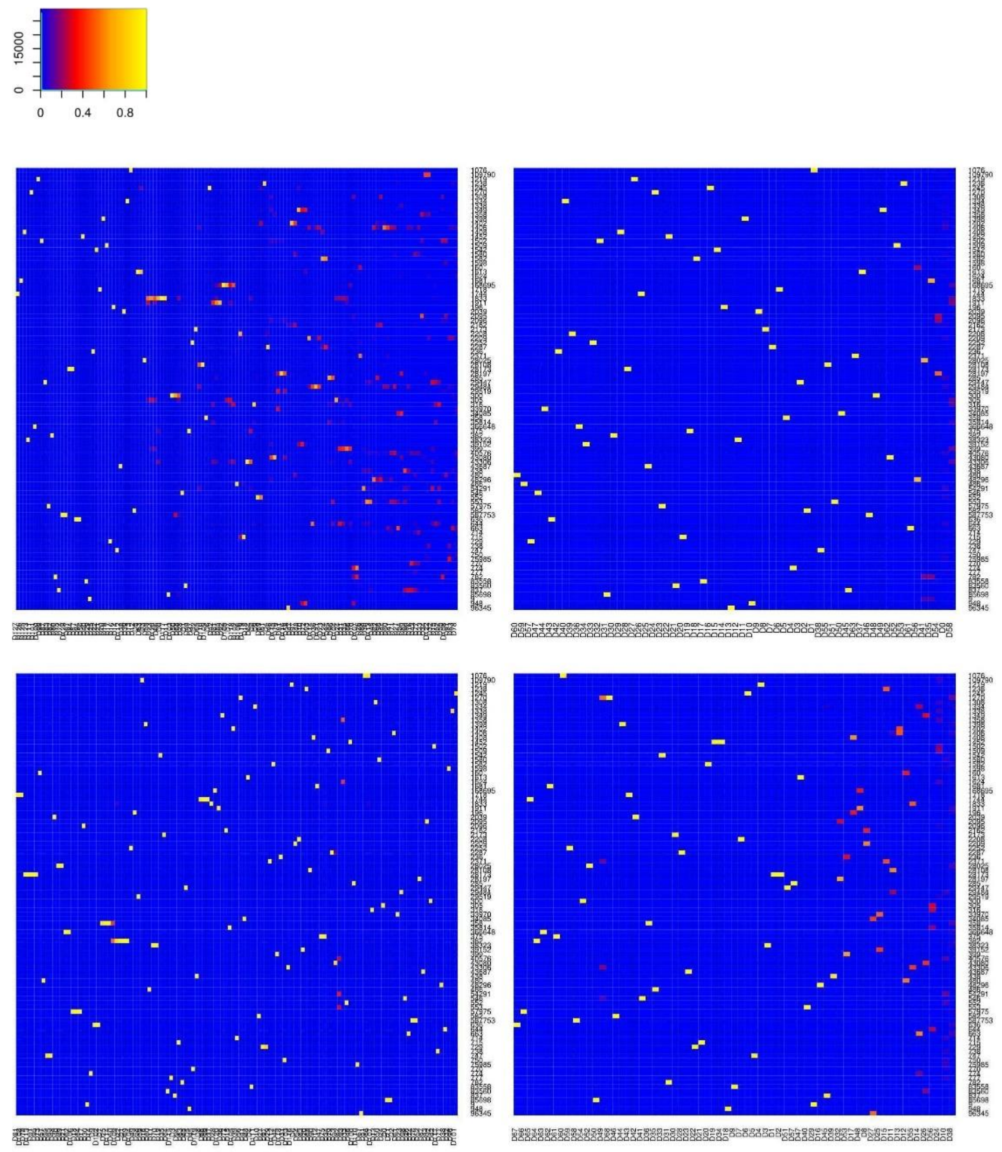
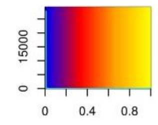


Figure A.6. Heat maps for Speceis_20S_7.5R. The top left is that of GroopM, top right of BMC3C, bottom left of MyCC, and bottom right of GATTACA



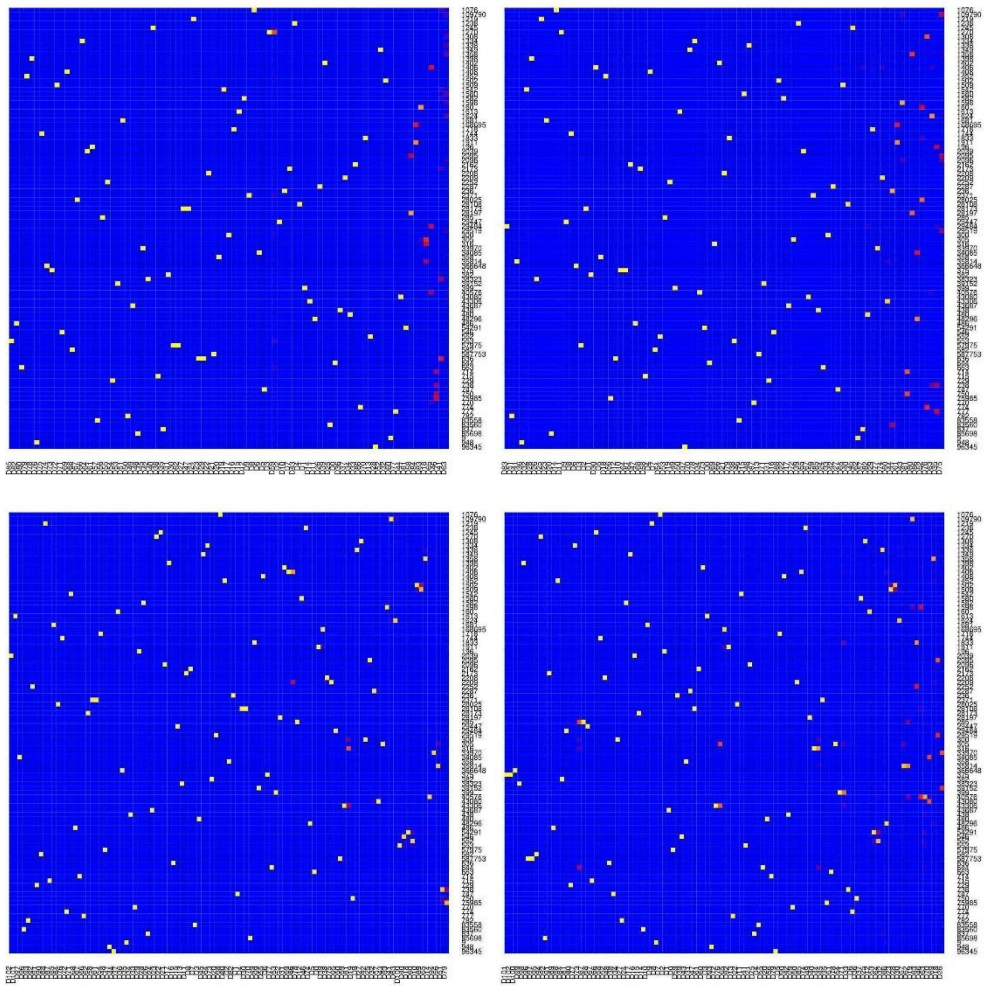


Figure A.7. Heat maps for Speceis_20S_10.0R. The top left is that of CONCOCT, top right of COCACOLA, bottom left of MetaBAT, and bottom right of MaxBin2

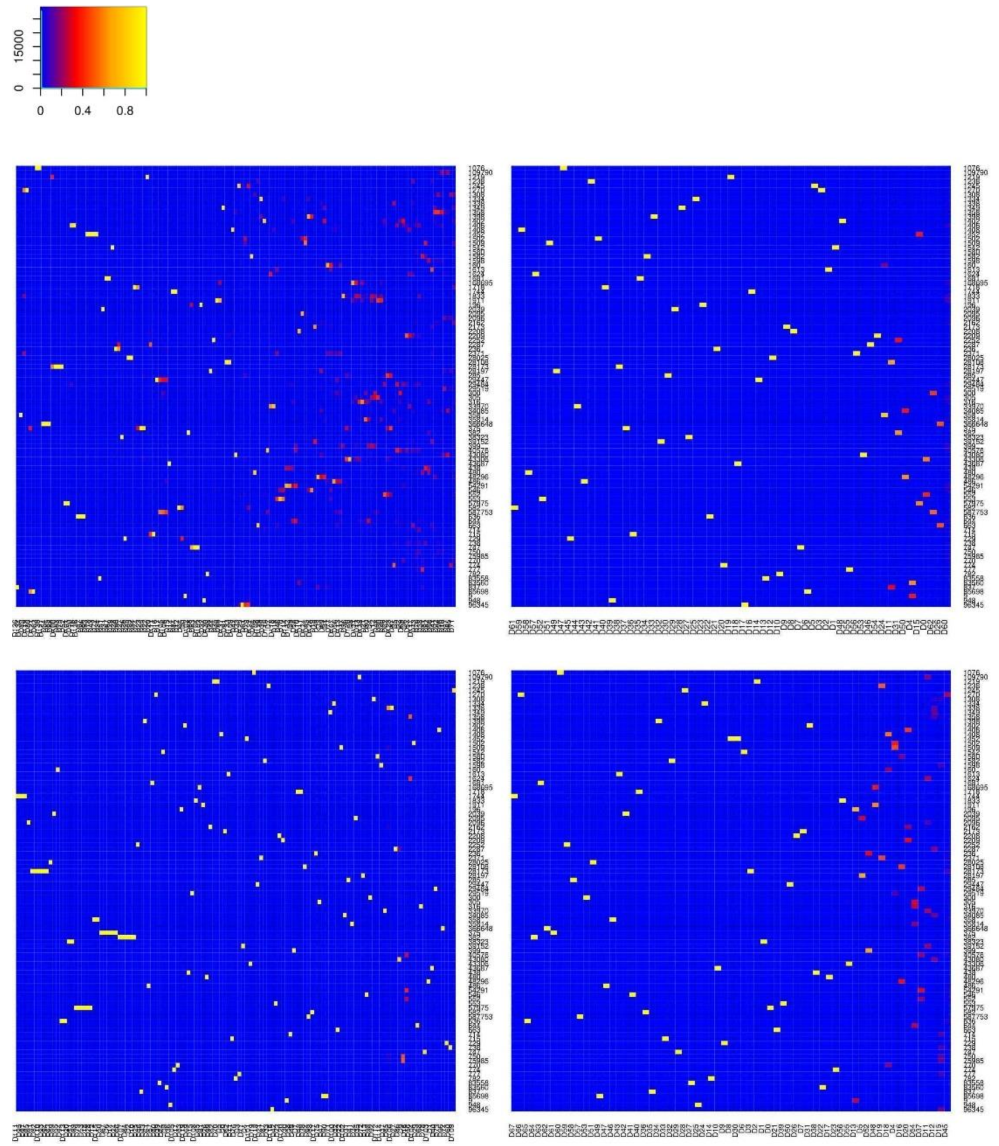


Figure A.8. Heat maps for Speceis_20S_10.0R. The top left is that of GroopM, top right of BMC3C, bottom left of MyCC, and bottom right of GATTACA

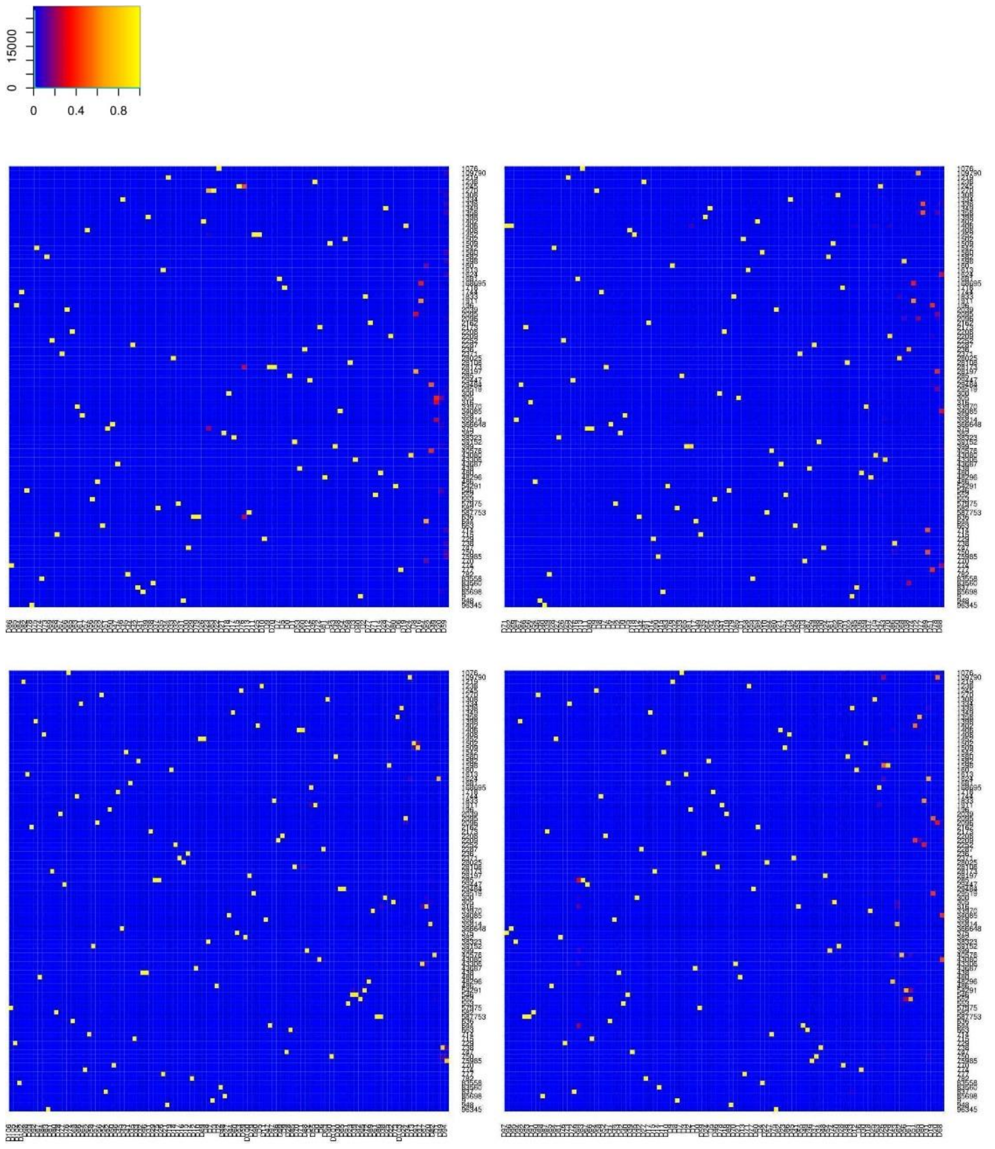
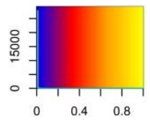


Figure A.9. Heat maps for Speceis_20S_12.5R. The top left is that of CONCOCT, top right of COCACOLA, bottom left of MetaBAT, and bottom right of MaxBin2



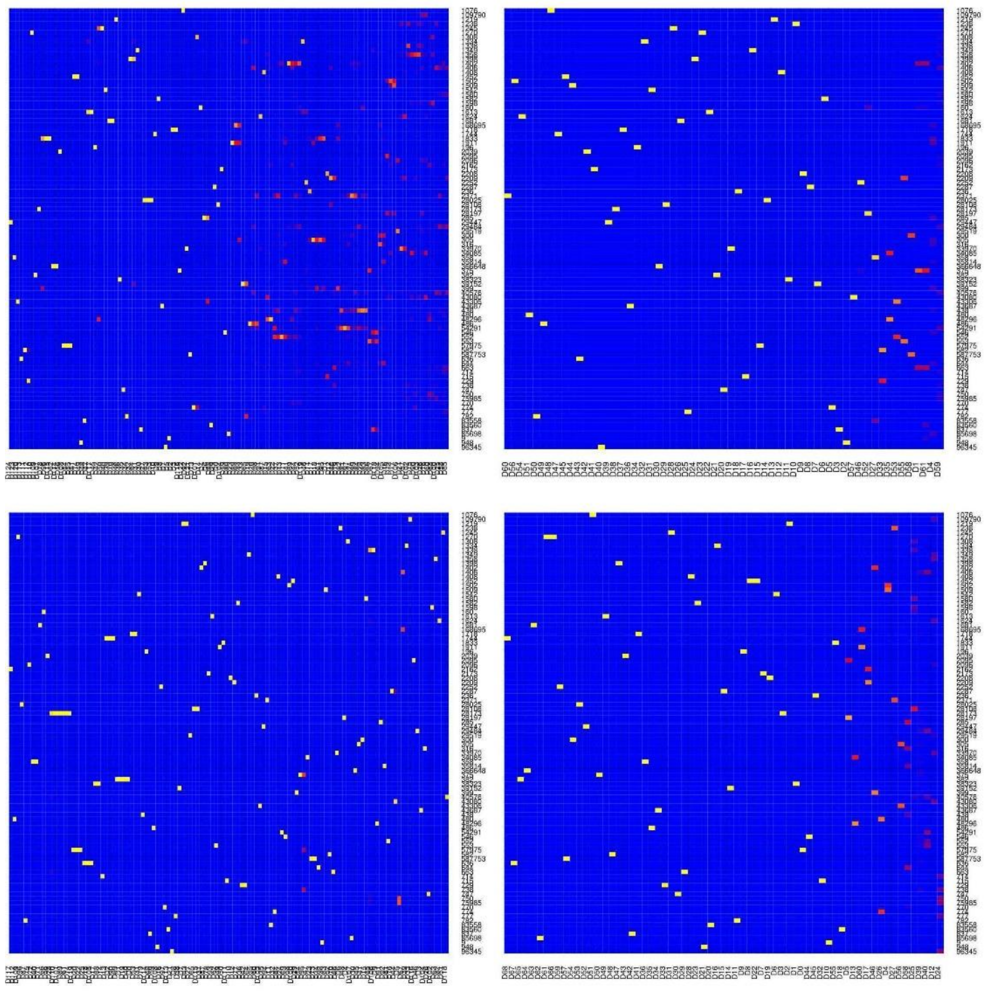


Figure A.10. Heat maps for Speceis_20S_12.5R. The top left is that of GroopM, top right of BMC3C, bottom left of MyCC, and bottom right of GATTACA

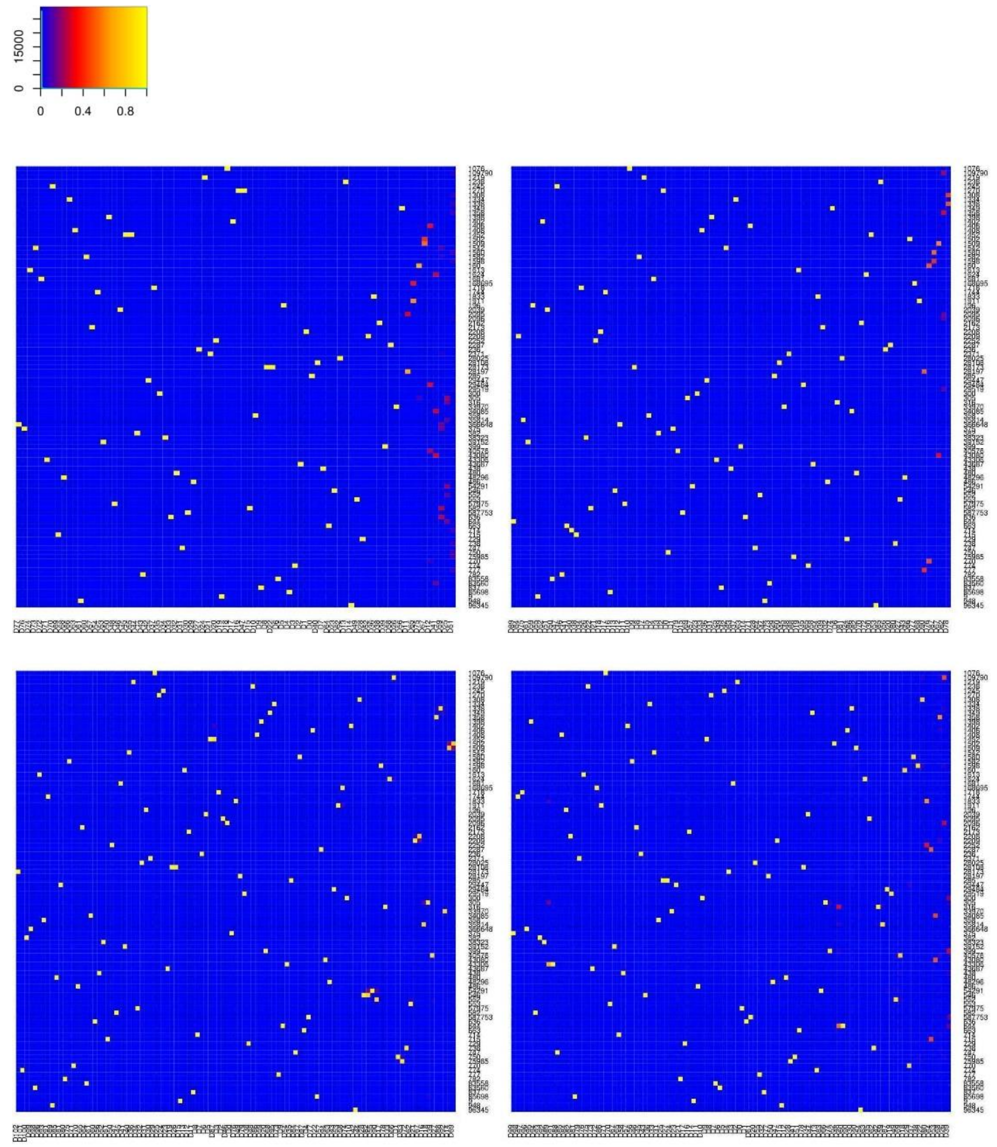


Figure A.11. Heat maps for Speceis_40S_12.5R. The top left is that of CONCOCT, top right of COCACOLA, bottom left of MetaBAT, and bottom right of MaxBin2

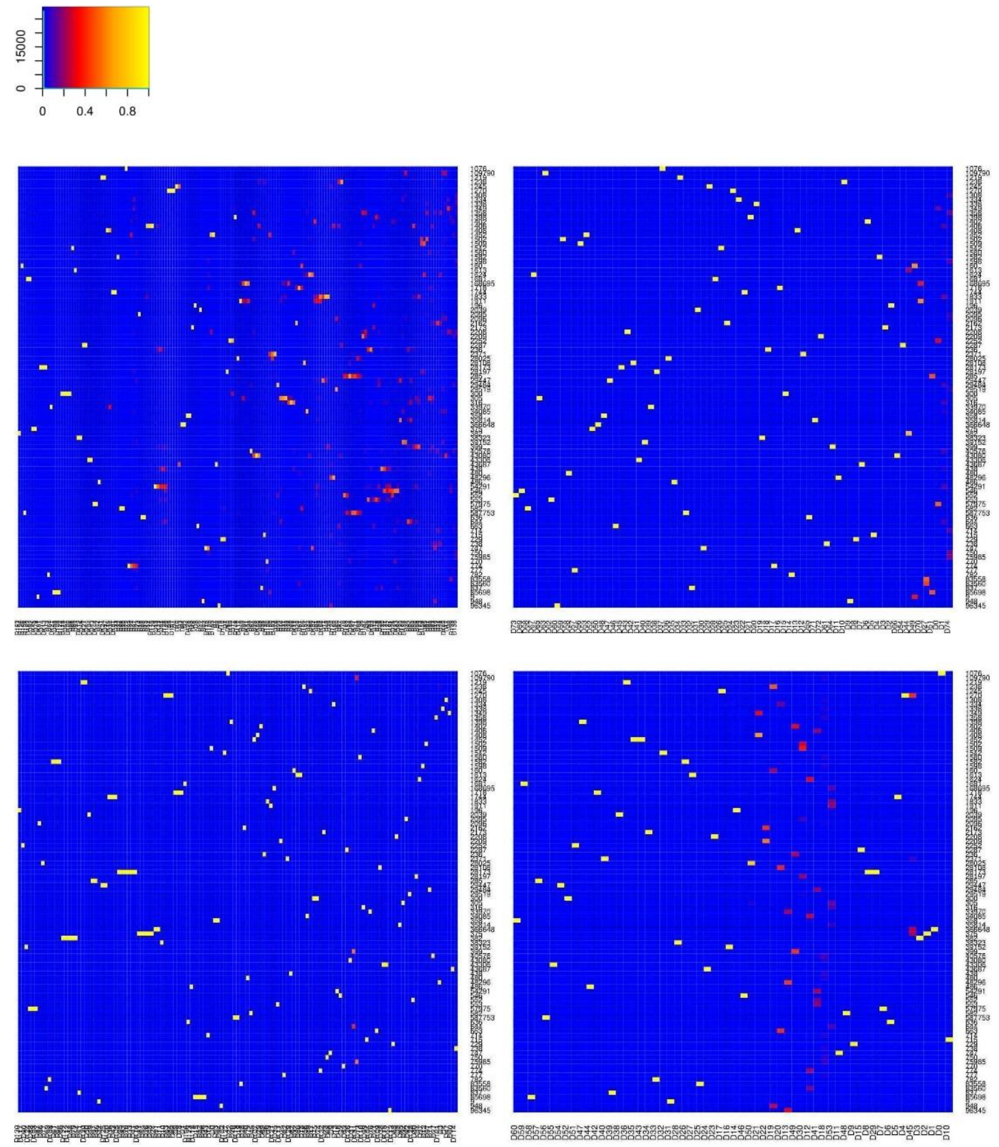


Figure A.12. Heat maps for Speceis_20S_10.0R. The top left is that of GroopM, top right of BMC3C, bottom left of MyCC, and bottom right of GATTACA

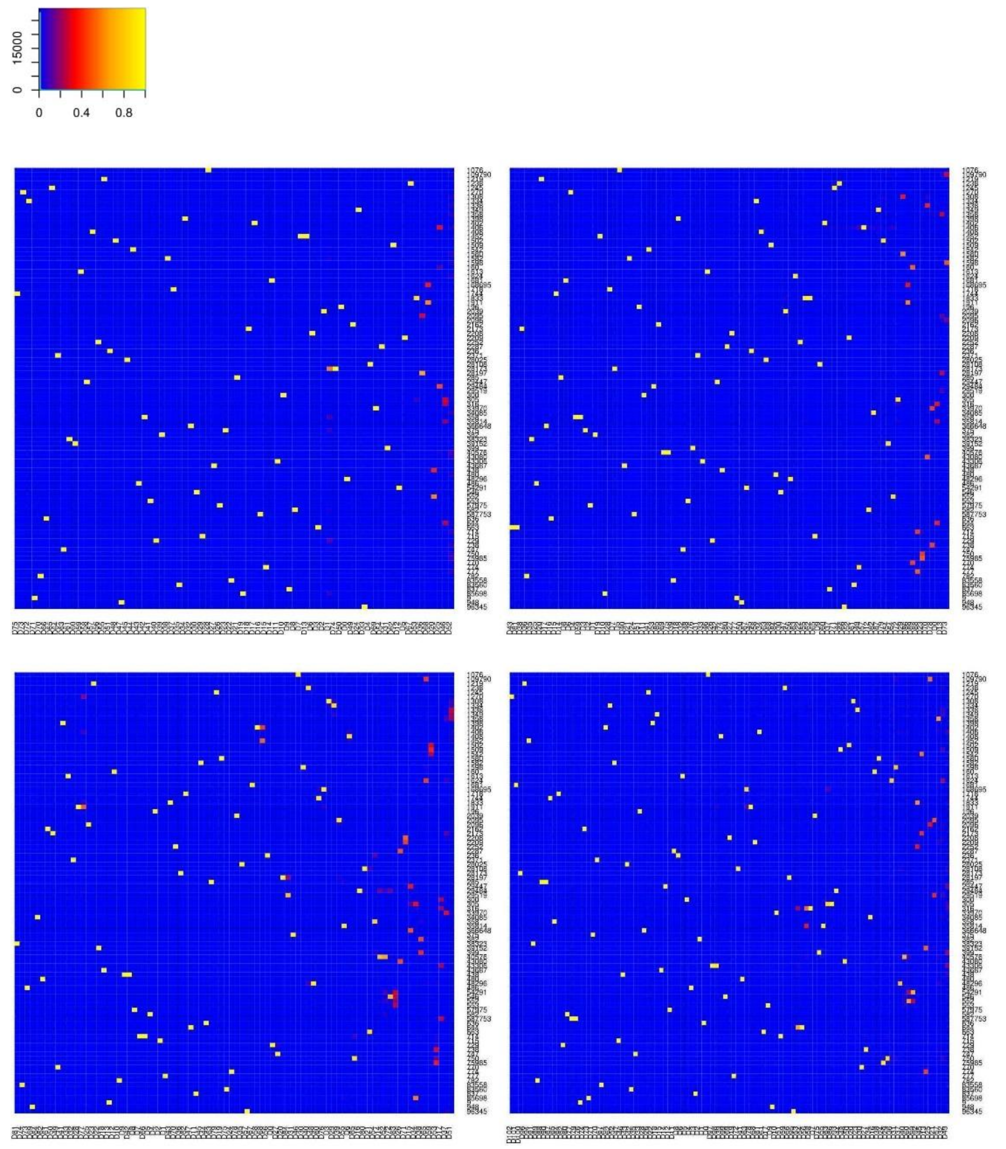
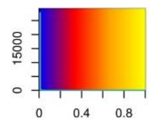


Figure A.13. Heat maps for Speceis_60S_2.5R. The top left is that of CONCOCT, top right of COCACOLA, bottom left of MetaBAT, and bottom right of MaxBin2



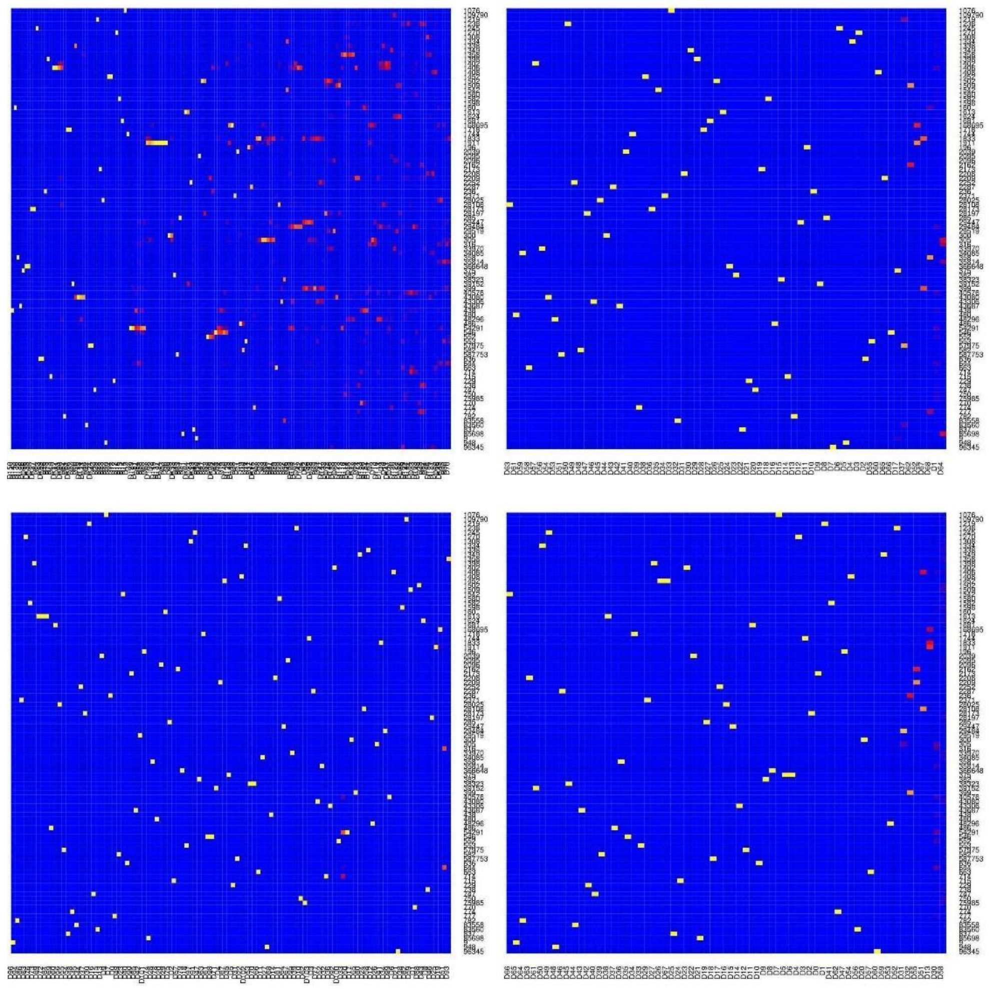


Figure A.14. Heat maps for Speceis_60S_2.5R. The top left is that of GroopM, top right of BMC3C, bottom left of MyCC, and bottom right of GATTACA

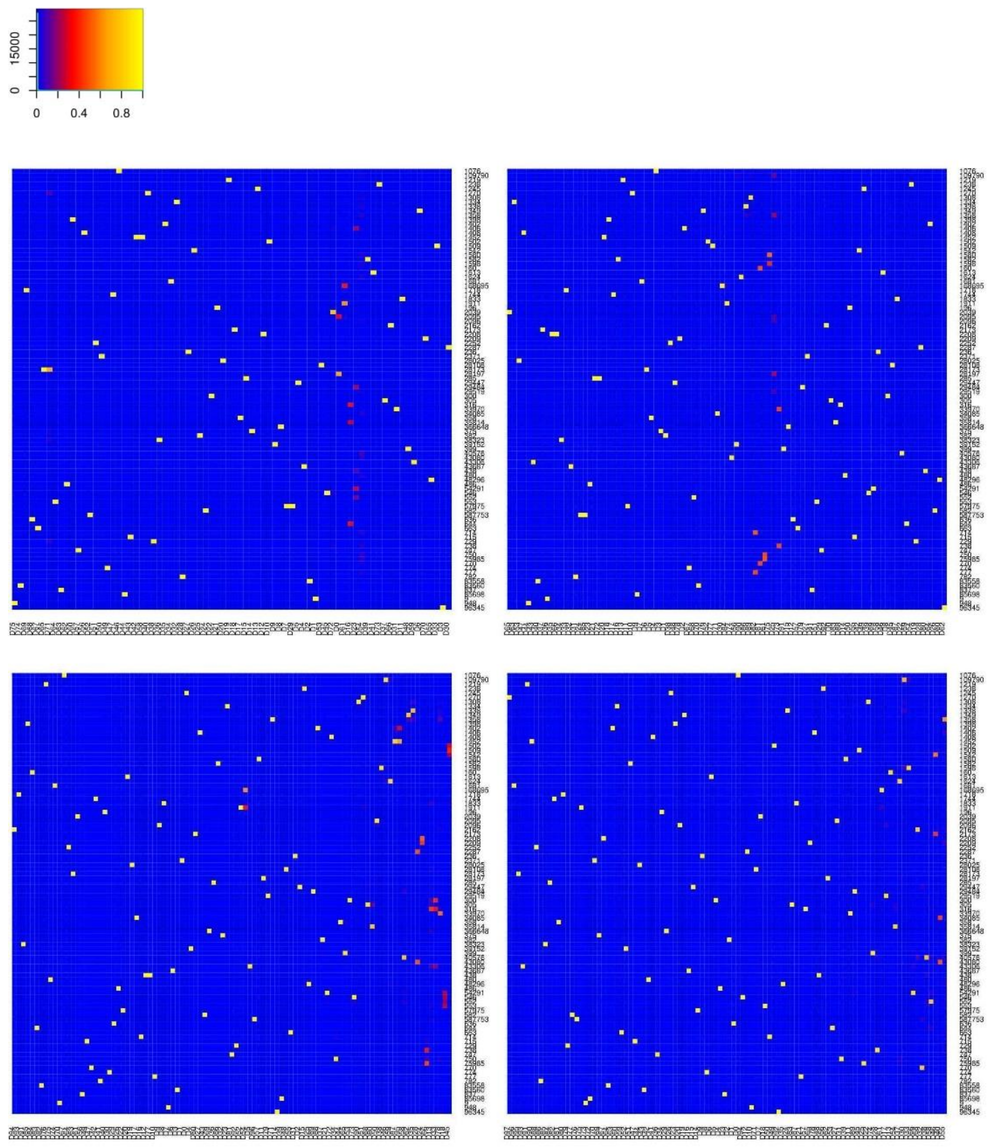
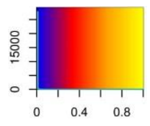


Figure A.15. Heat maps for Speceis_60S_5.0R. The top left is that of CONCOCT, top right of COCACOLA, bottom left of MetaBAT, and bottom right of MaxBin2



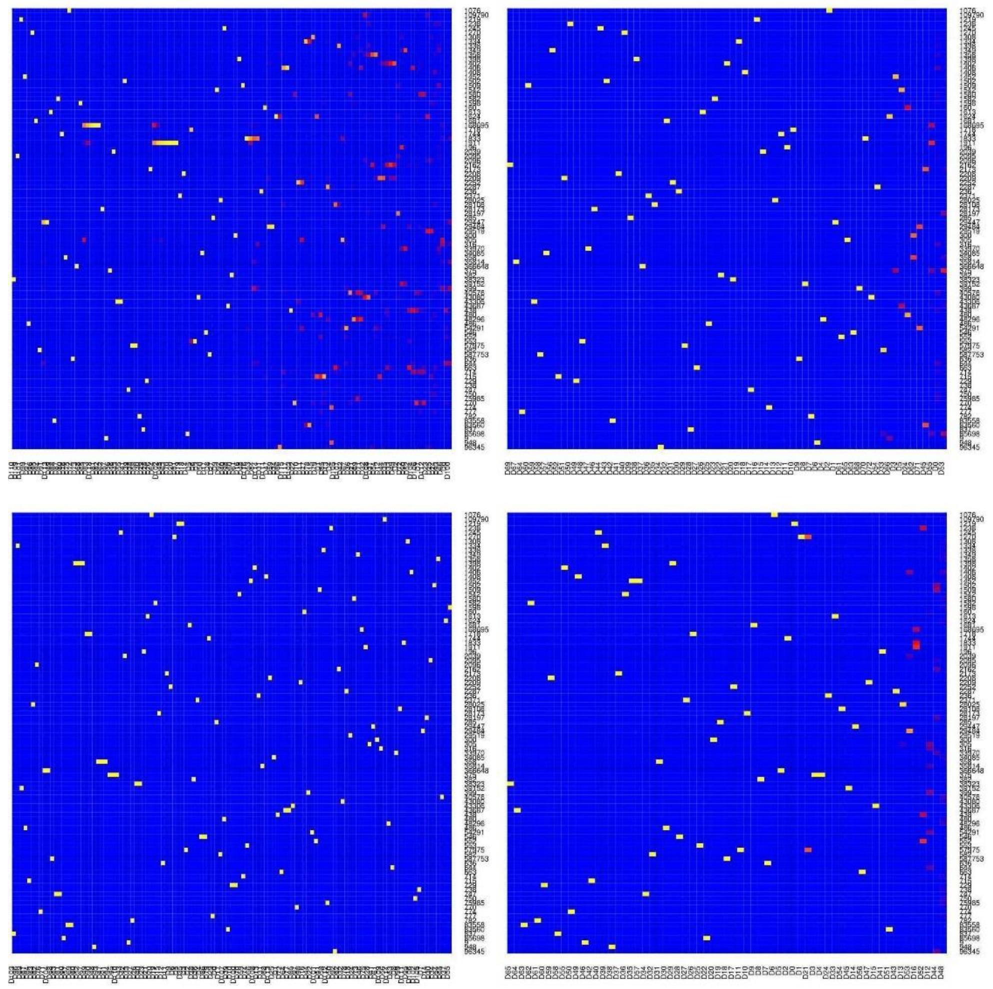


Figure A.16. Heat maps for Speceis_60S_5.0R. The top left is that of GroopM, top right of BMC3C, bottom left of MyCC, and bottom right of GATTACA

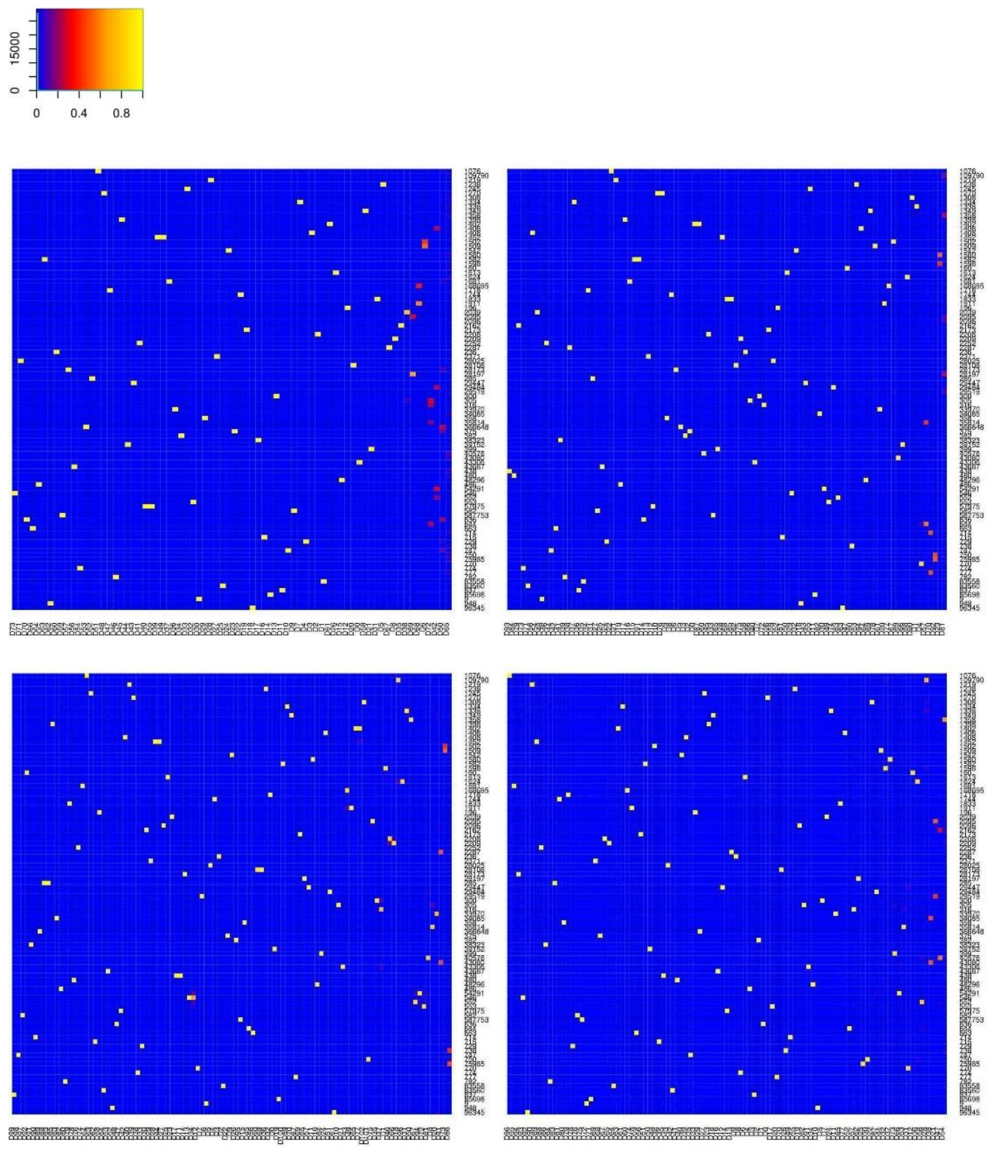


Figure A.17. Heat maps for Speceis_60S_7.5R. The top left is that of CONCOCT, top right of COCACOLA, bottom left of MetaBAT, and bottom right of MaxBin2

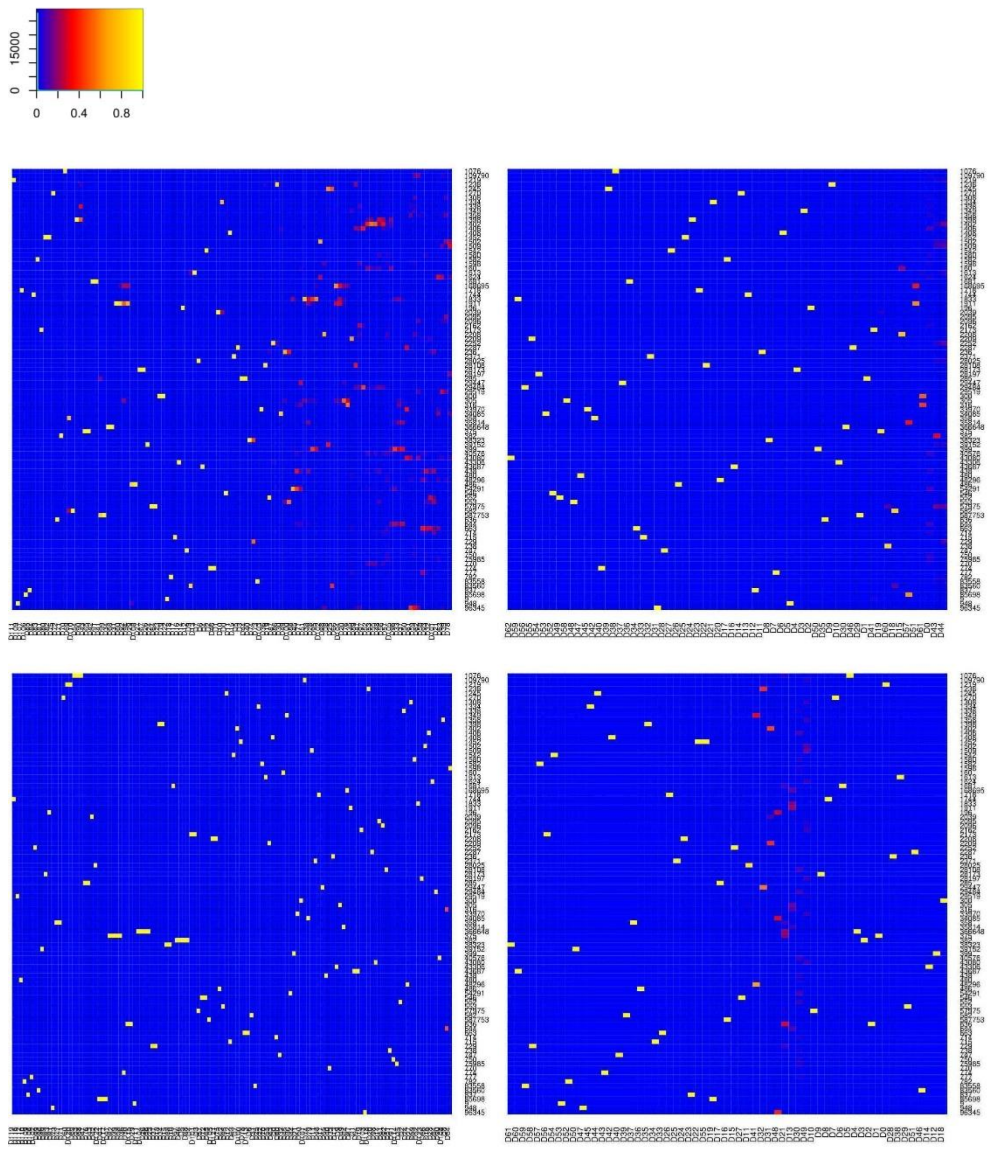


Figure A.18. Heat maps for Speceis_60S_7.5R. The top left is that of GroopM, top right of BMC3C, bottom left of MyCC, and bottom right of GATTACA

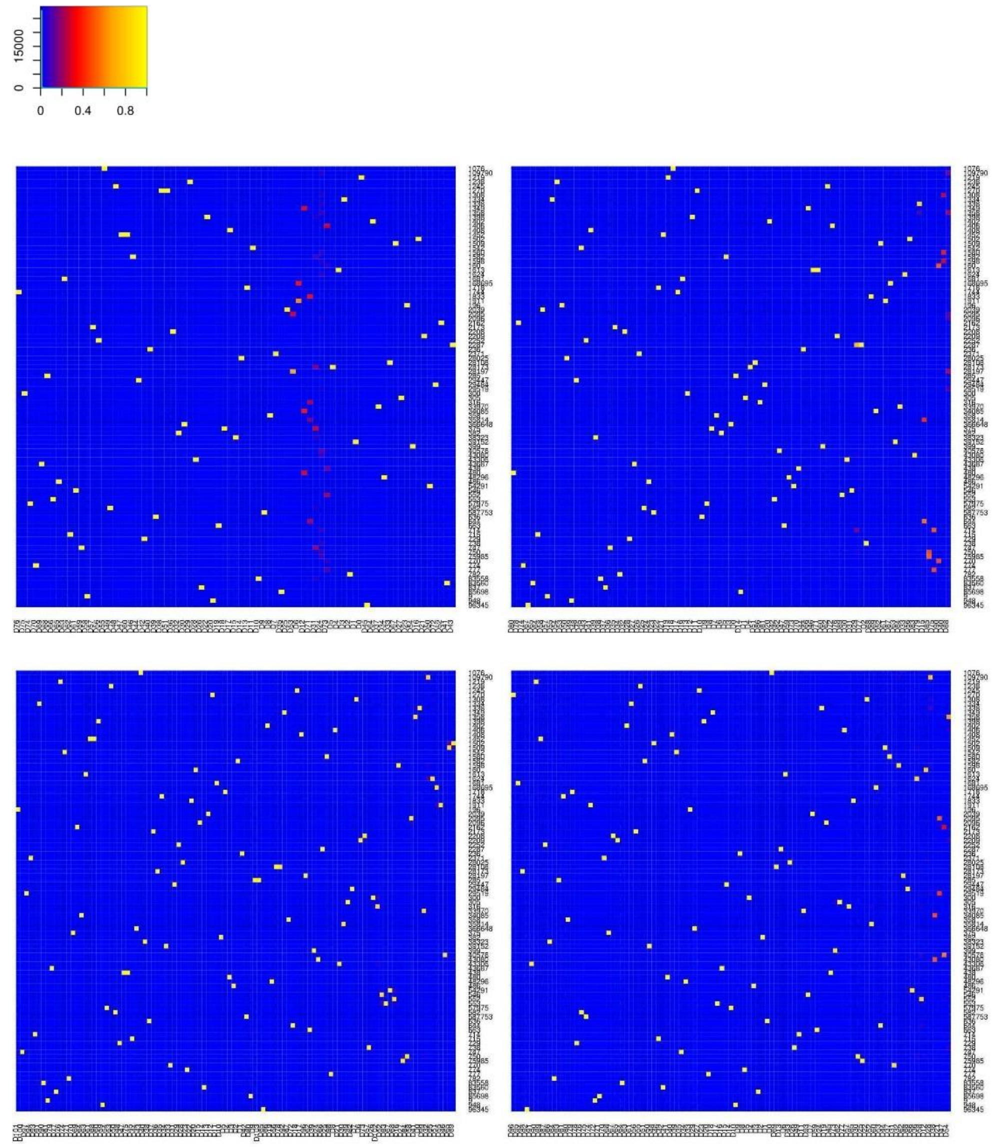
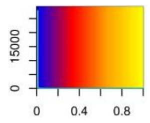


Figure A.19. Heat maps for Speceis_60S_10.0R. The top left is that of CONCOCT, top right of COCACOLA, bottom left of MetaBAT, and bottom right of MaxBin2



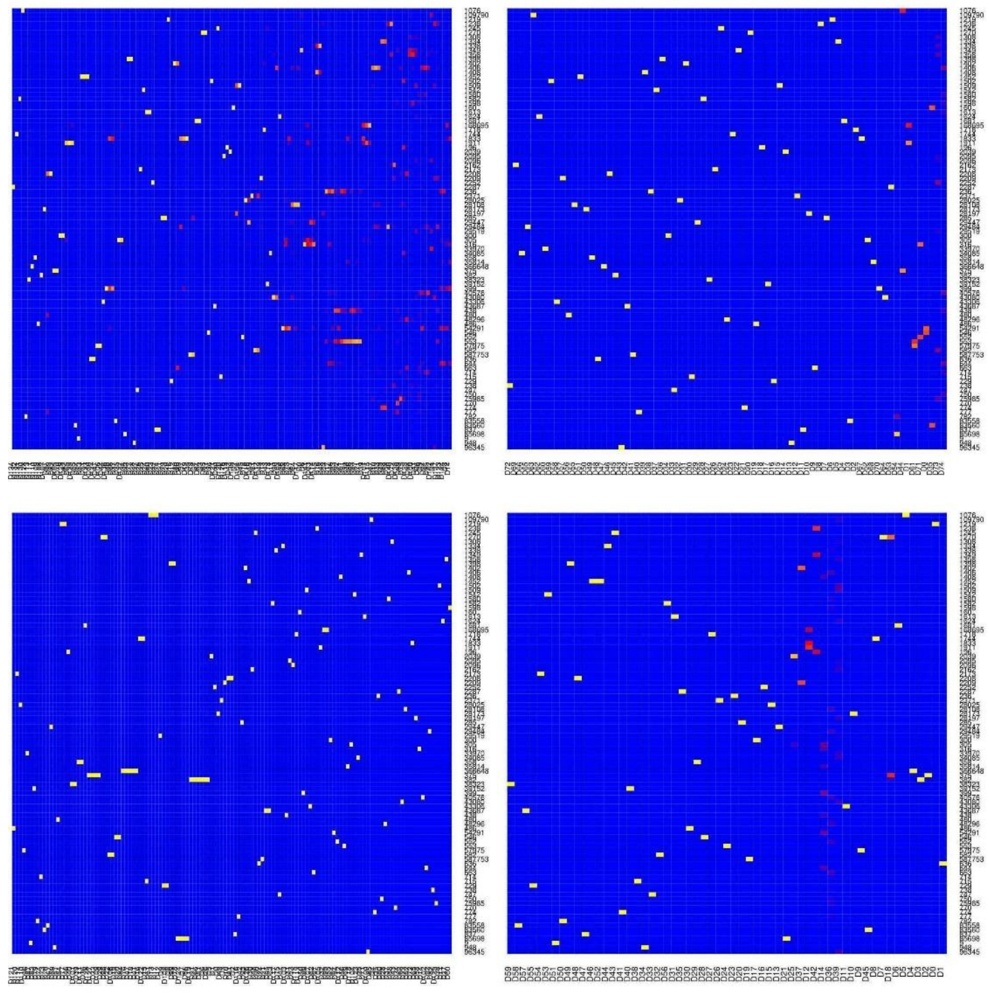


Figure A.20. Heat maps for Speceis_60S_10.0R. The top left is that of GroopM, top right of BMC3C, bottom left of MyCC, and bottom right of GATTACA

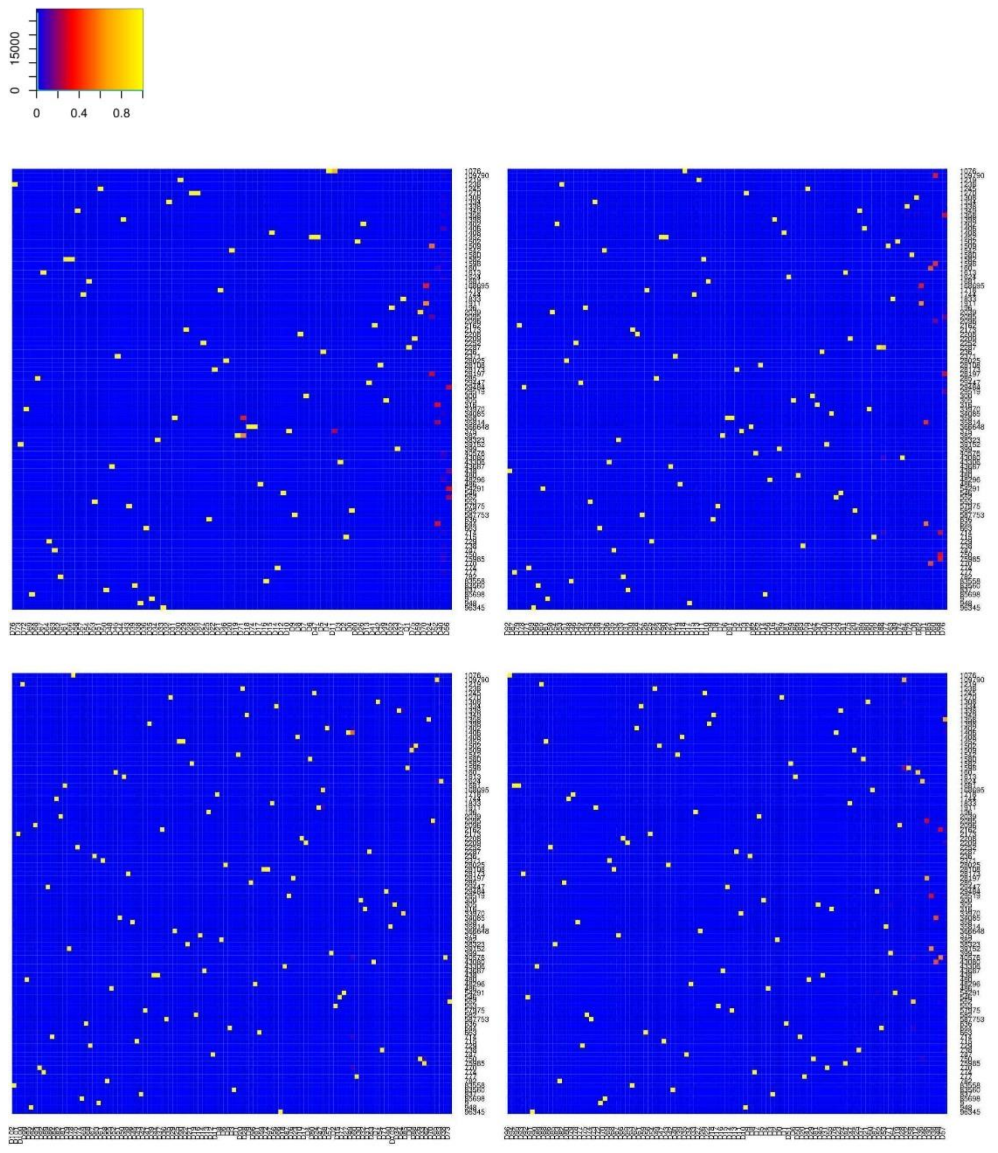


Figure A.21. Heat maps for Speceis_60S_12.5R. The top left is that of CONCOCT, top right of COCACOLA, bottom left of MetaBAT, and bottom right of MaxBin2

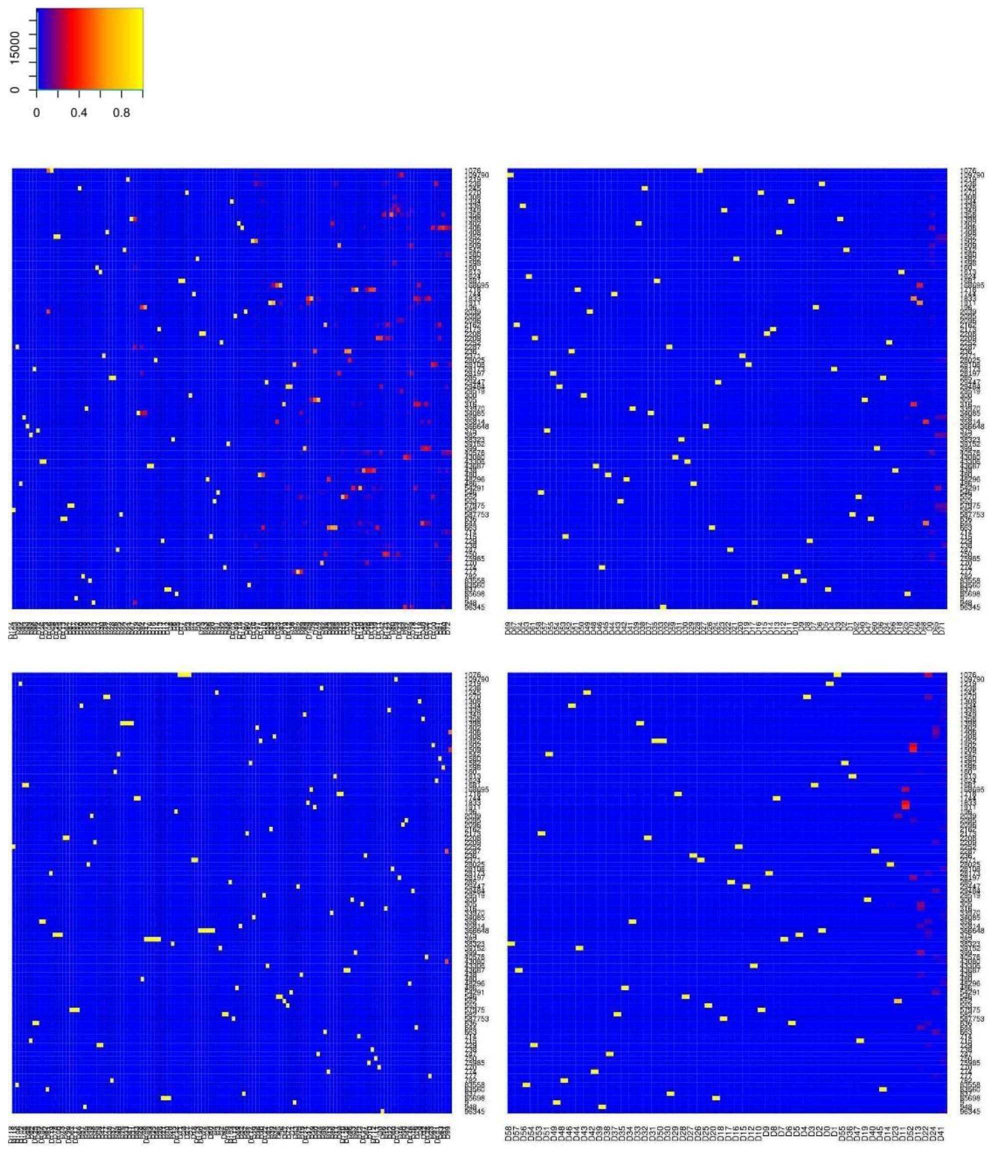


Figure A.22. Heat maps for Speceis_60S_12.5R. The top left is that of GroopM, top right of BMC3C, bottom left of MyCC, and bottom right of GATTACA

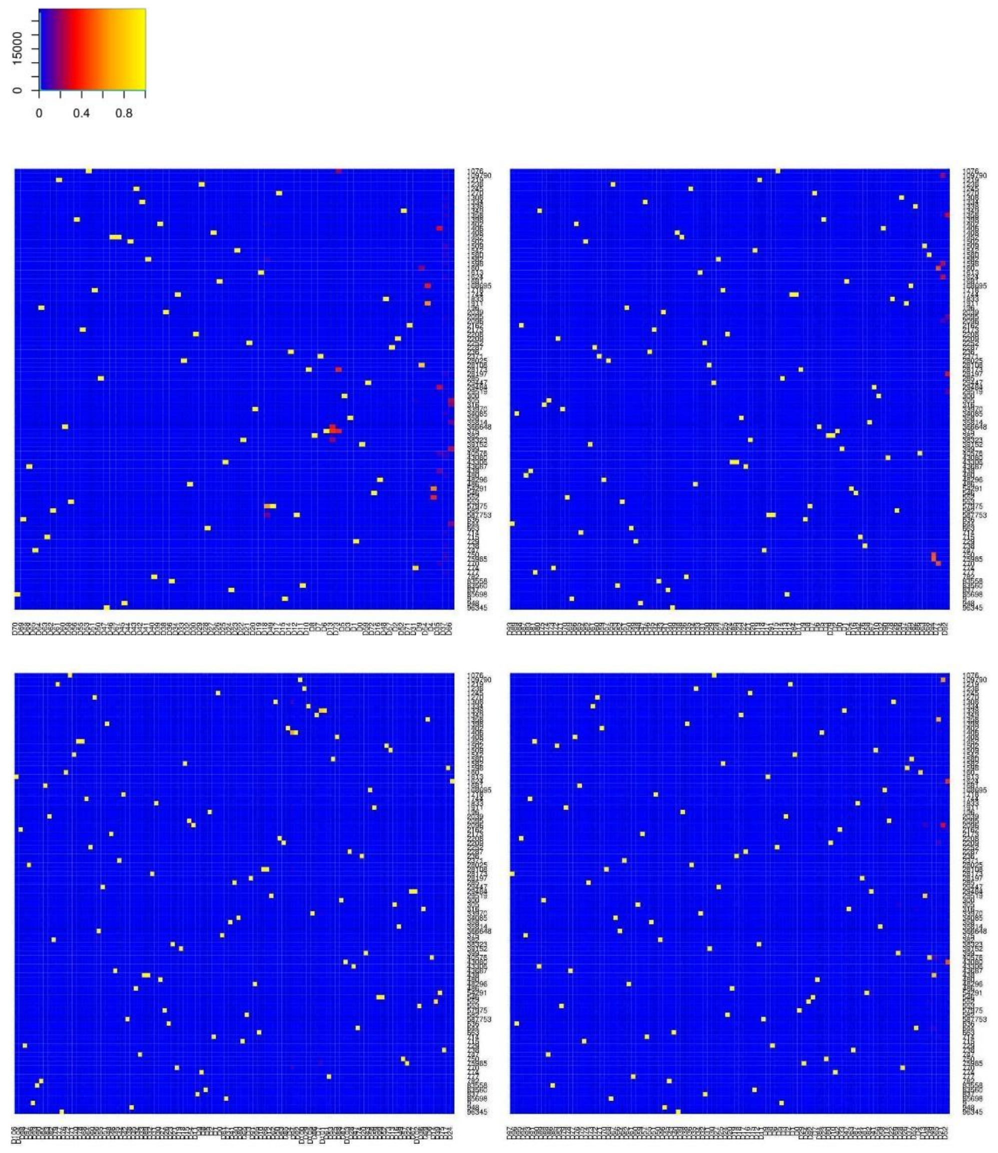
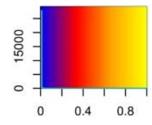


Figure A.23. Heat maps for Speceis_80S_12.5R. The top left is that of CONCOCT, top right of COCACOLA, bottom left of MetaBAT, and bottom right of MaxBin2



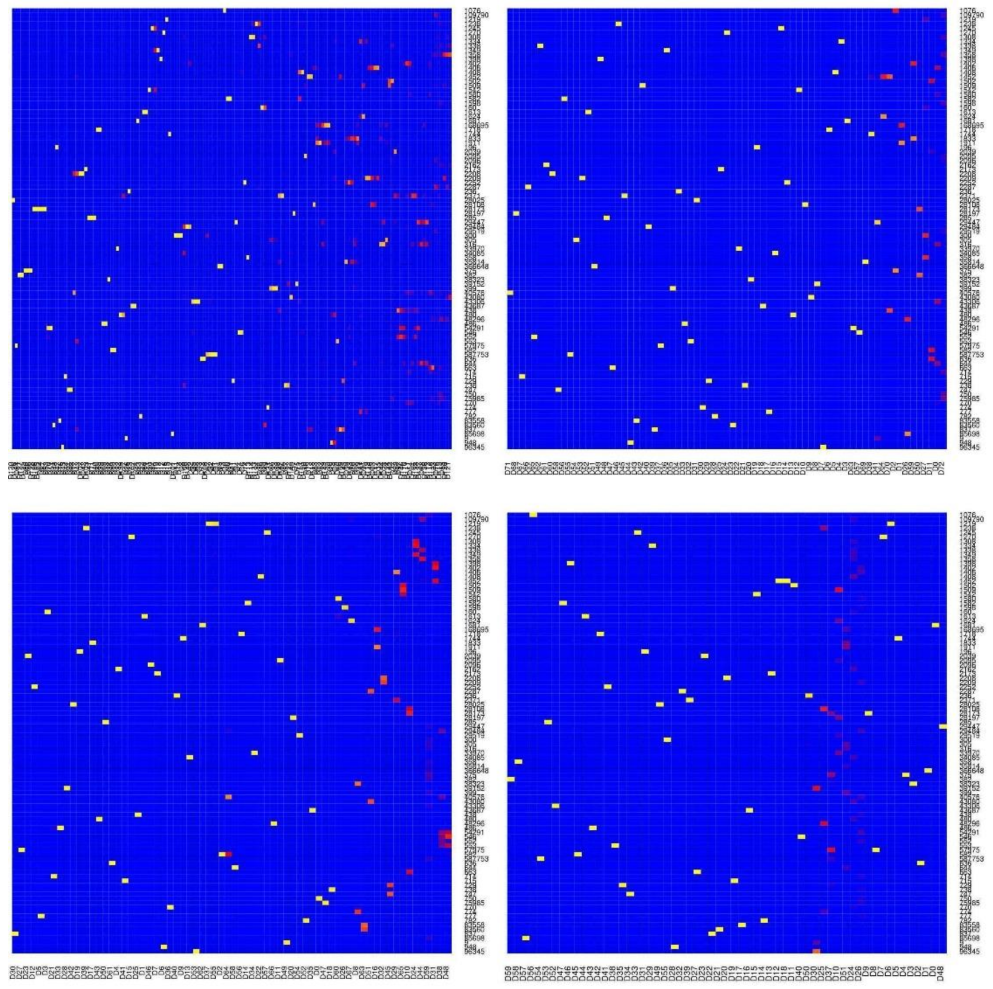
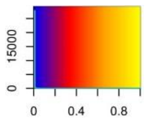


Figure A.24. Heat maps for Speceis_80S_12.5R. The top left is that of GroopM, top right of BMC3C, bottom left of MyCC, and bottom right of GATTACA



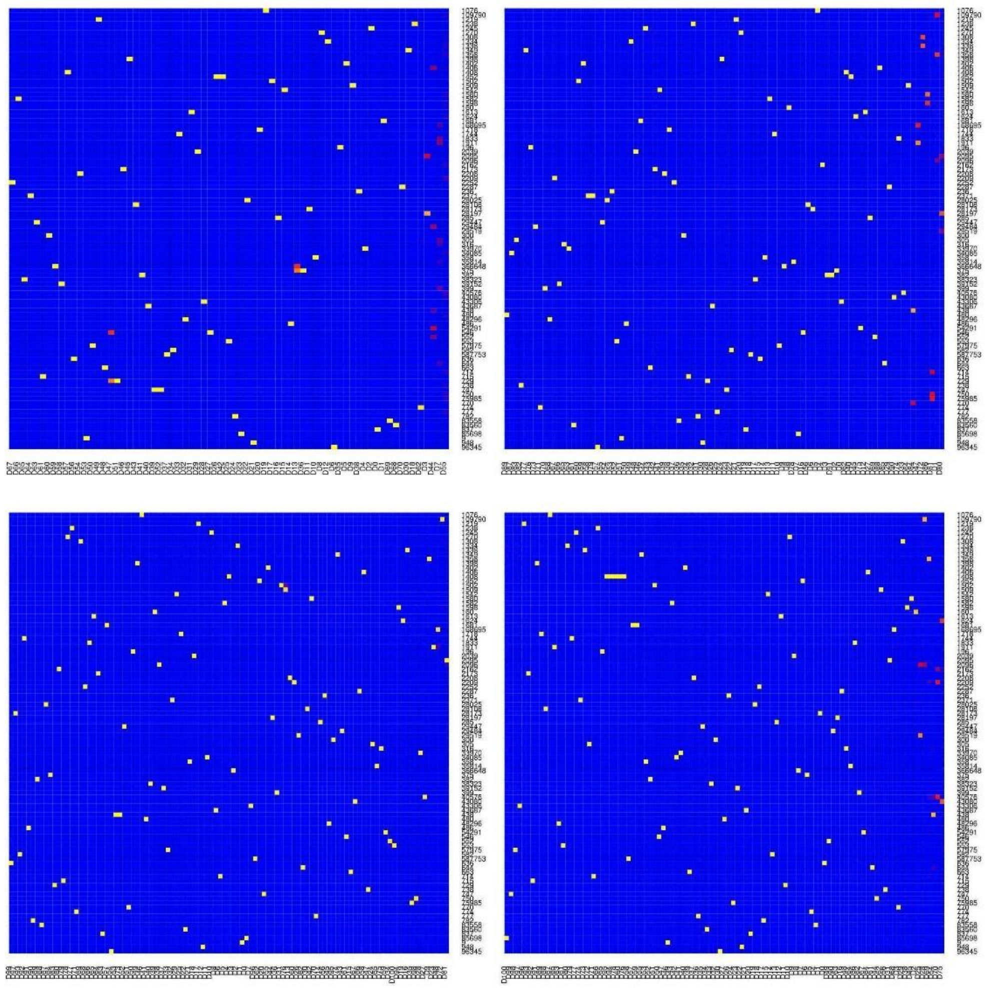


Figure A.25. Heat maps for Speceis_100S_12.5R. The top left is that of CONCOCT, top right of COCACOLA, bottom left of MetaBAT, and bottom right of MaxBin2

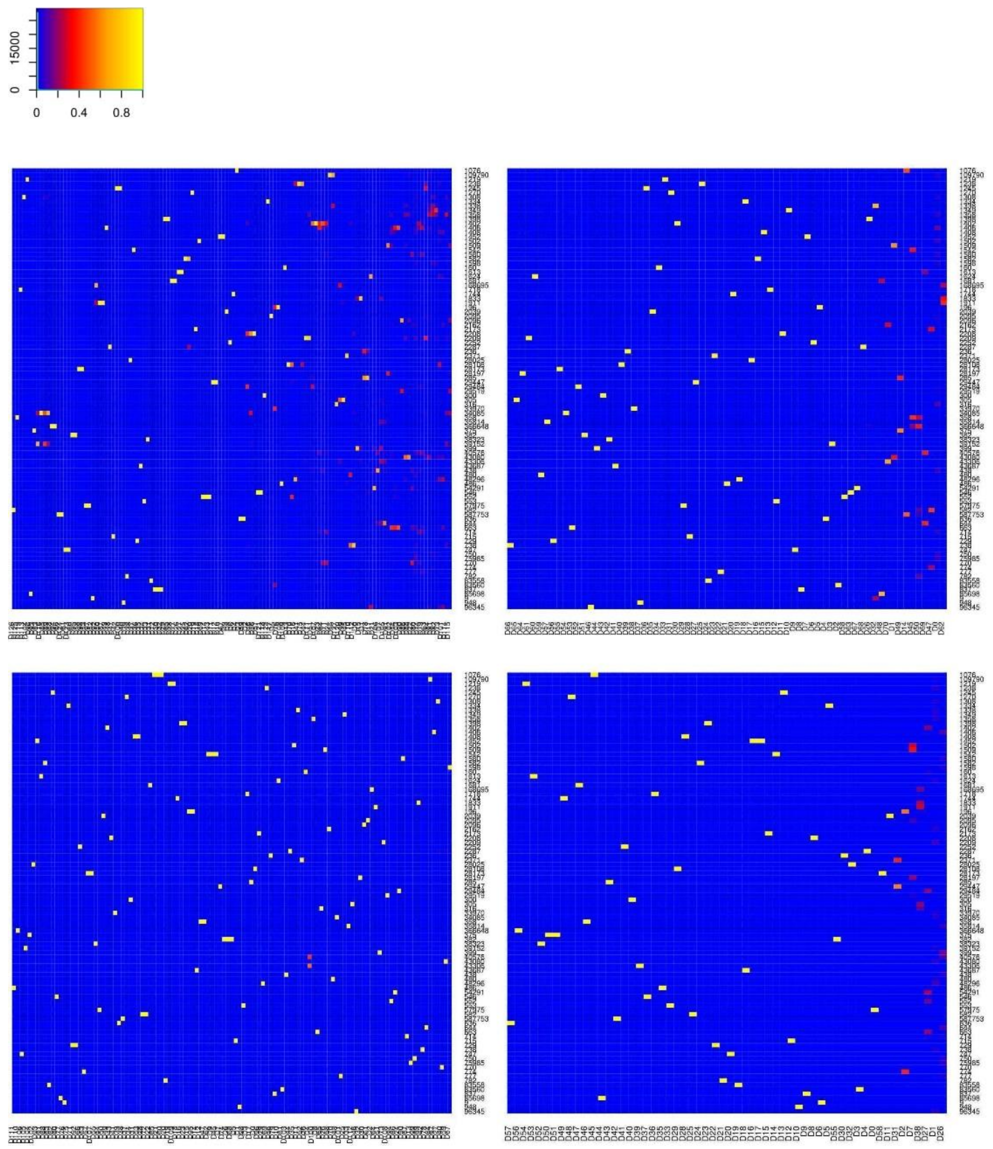


Figure A.26. Heat maps for Speceis_100S_12.5R. The top left is that of GroopM, top right of BMC3C, bottom left of MyCC, and bottom right of GATTACA

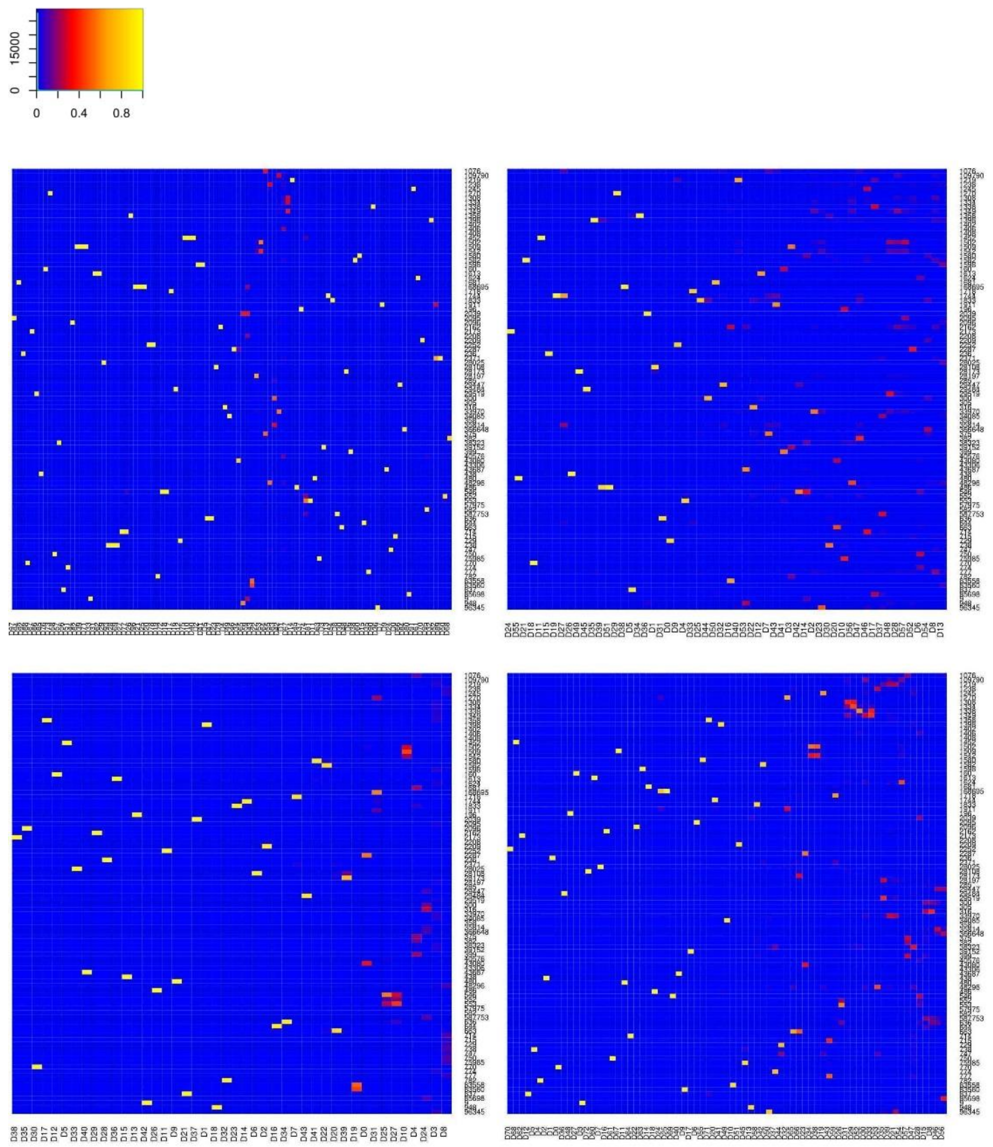


Figure A.27. Heat maps for Strains_20S_2.5R. The top left is that of CONCOCT, top right of COCACOLA, bottom left of MetaBAT, and bottom right of MaxBin2

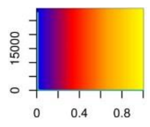
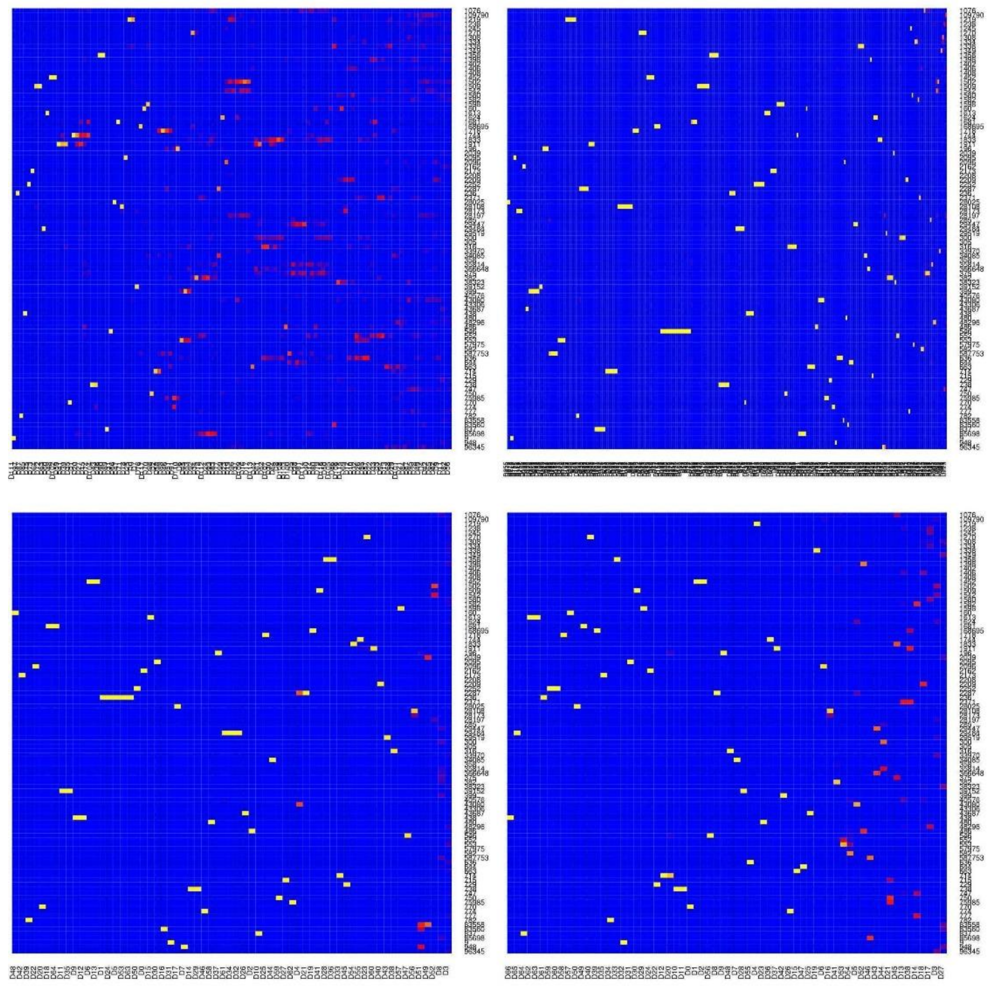
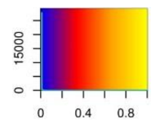



Figure A.28. Heat maps for Strains_20S_2.5R. The top left is that of GroopM, top right of BMC3C, bottom left of MyCC, and bottom right of GATTACA



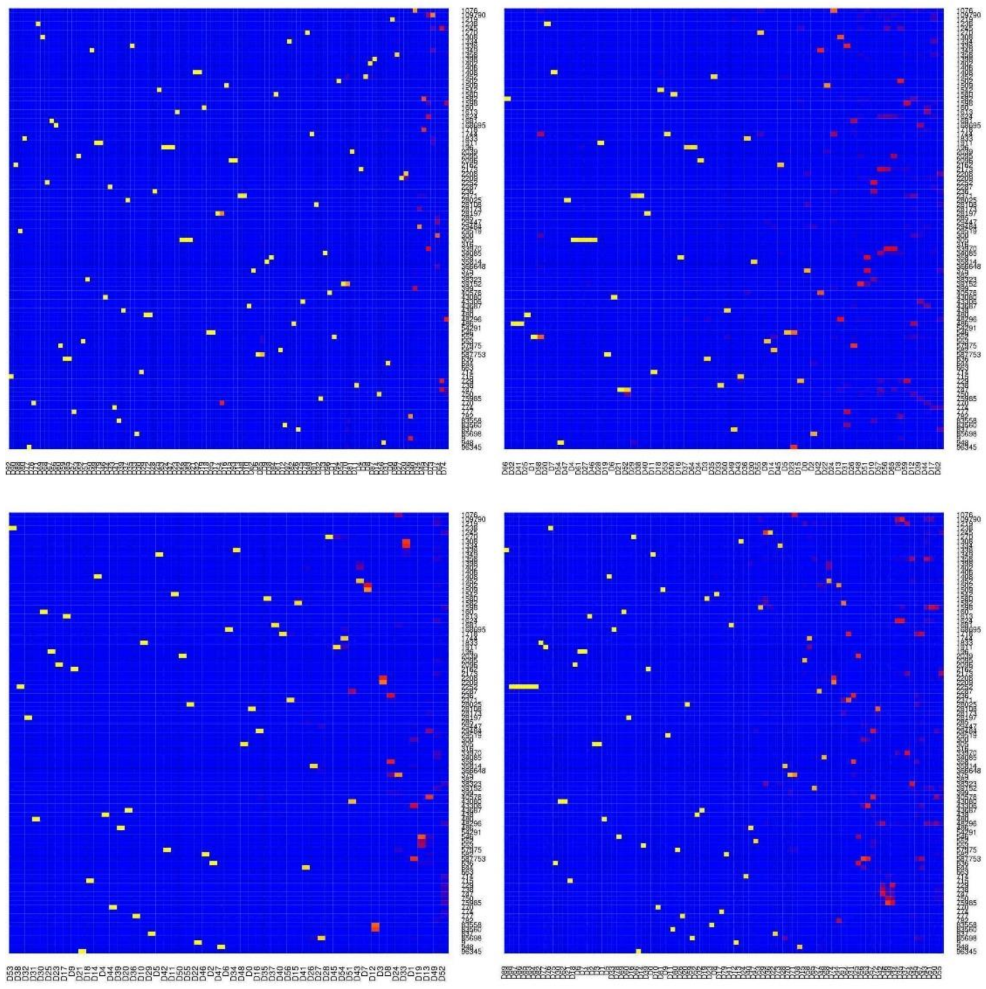
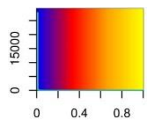


Figure A.29. Heat maps for Strains_20S_5.0R. The top left is that of CONCOCT, top right of COCACOLA, bottom left of MetaBAT, and bottom right of MaxBin2



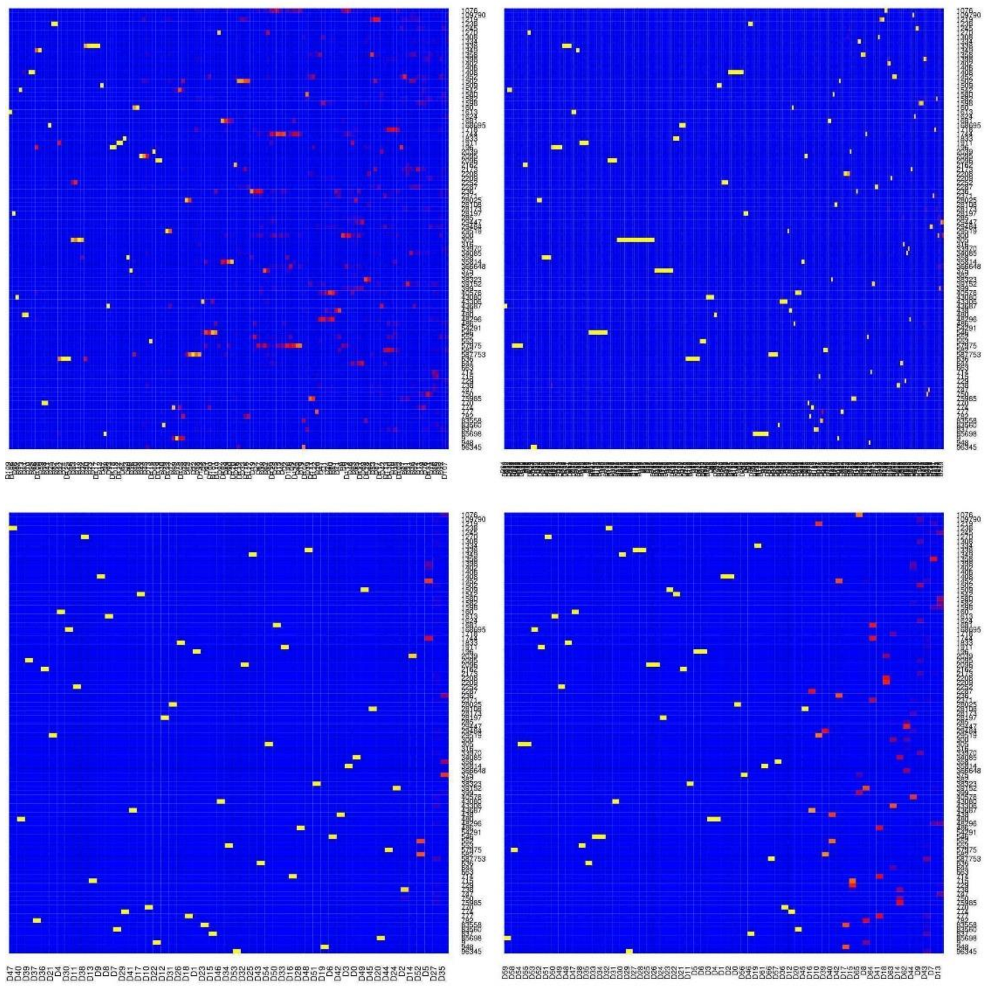


Figure A.30. Heat maps for Strains_20S_5.0R. The top left is that of GroopM, top right of BMC3C, bottom left of MyCC, and bottom right of GATTACA

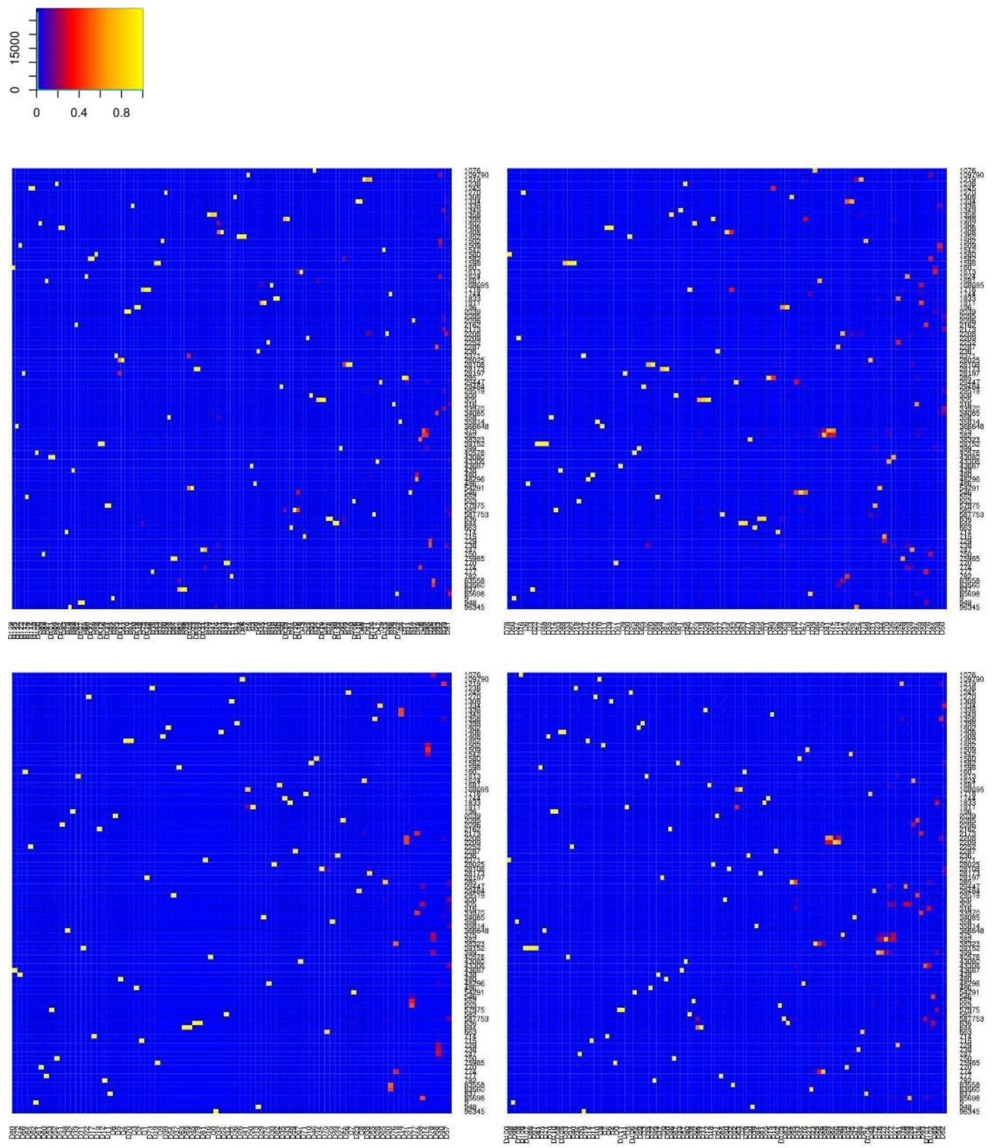


Figure A.31. Heat maps for Strains_20S_7.5R. The top left is that of CONCOCT, top right of COCACOLA, bottom left of MetaBAT, and bottom right of MaxBin2

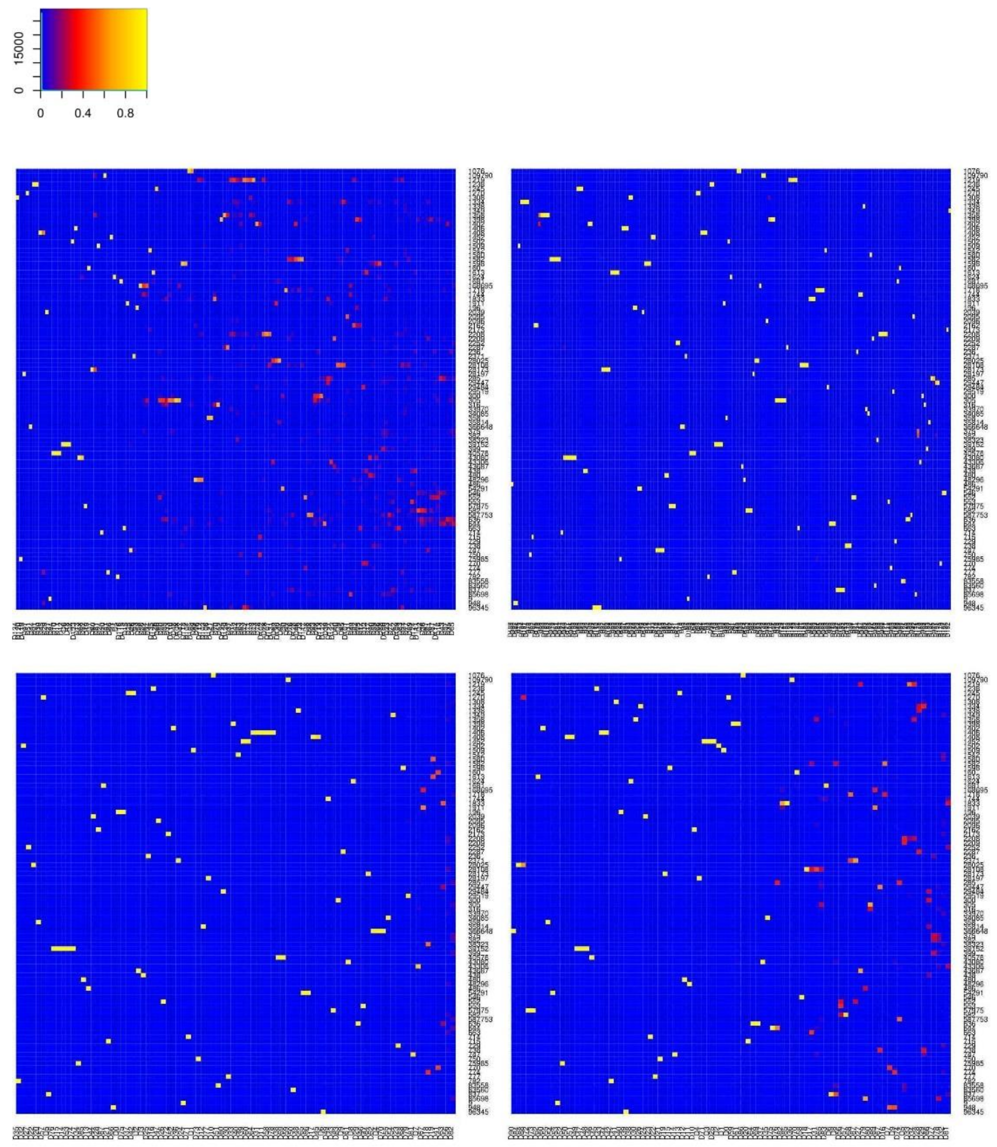


Figure A.32. Heat maps for Strains_20S_7.5R. The top left is that of GroopM, top right of BMC3C, bottom left of MyCC, and bottom right of GATTACA

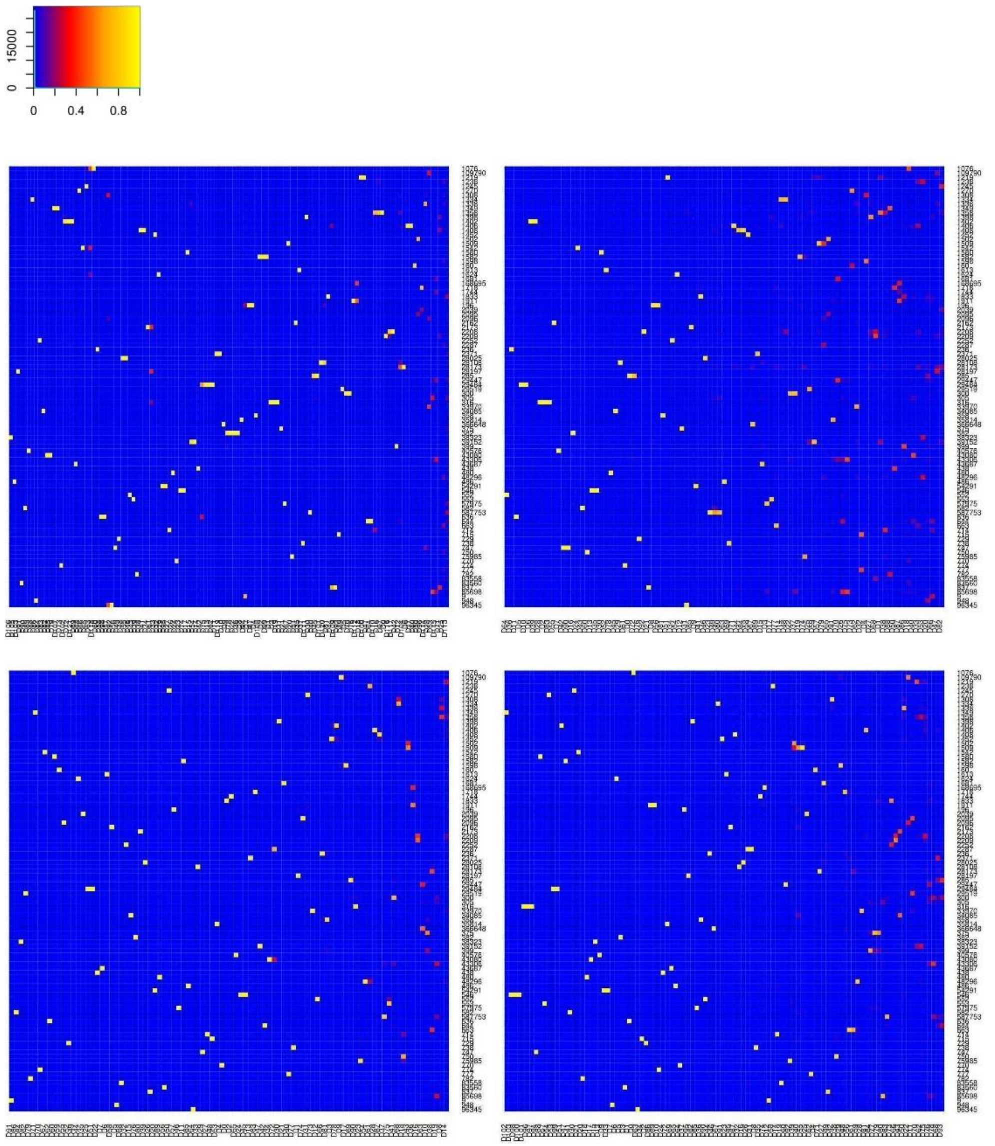


Figure A.33. Heat maps for Strains_20S_10.0R. The top left is that of CONCOCT, top right of COCACOLA, bottom left of MetaBAT, and bottom right of MaxBin2

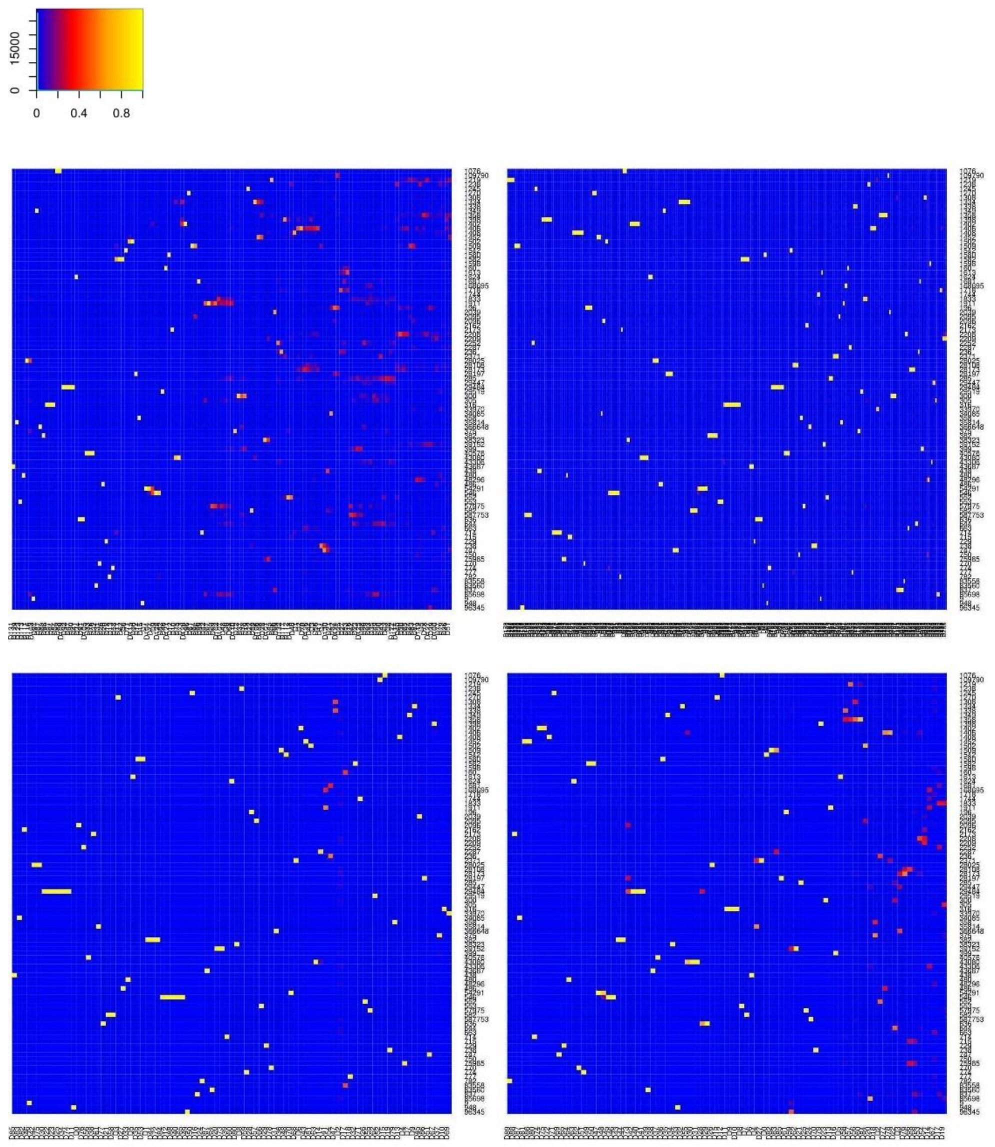


Figure A.34. Heat maps for Strains_20S_10.0R. The top left is that of GroopM, top right of BMC3C, bottom left of MyCC, and bottom right of GATTACA

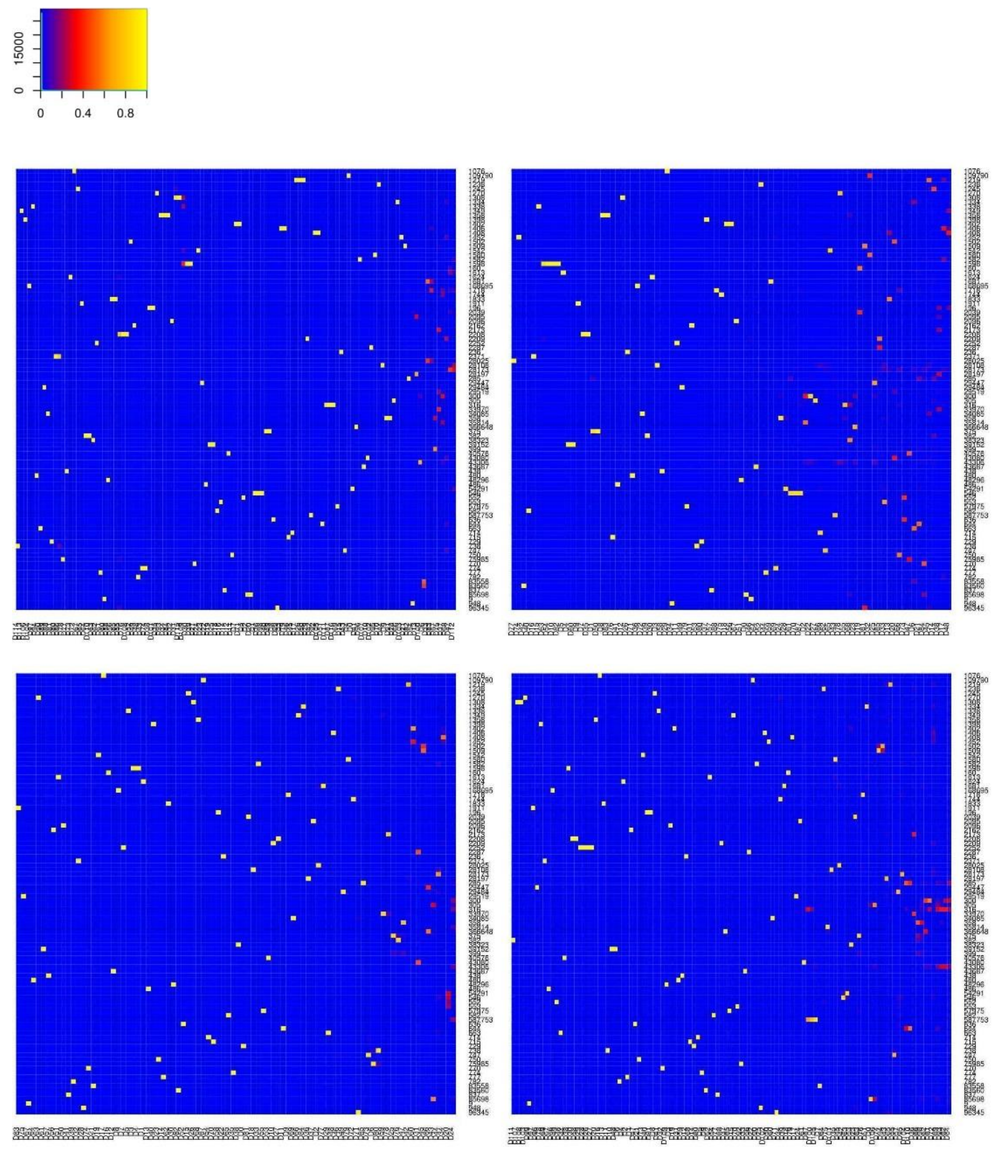


Figure A.35. Heat maps for Strains_20S_12.5R. The top left is that of CONCOCT, top right of COCACOLA, bottom left of MetaBAT, and bottom right of MaxBin2

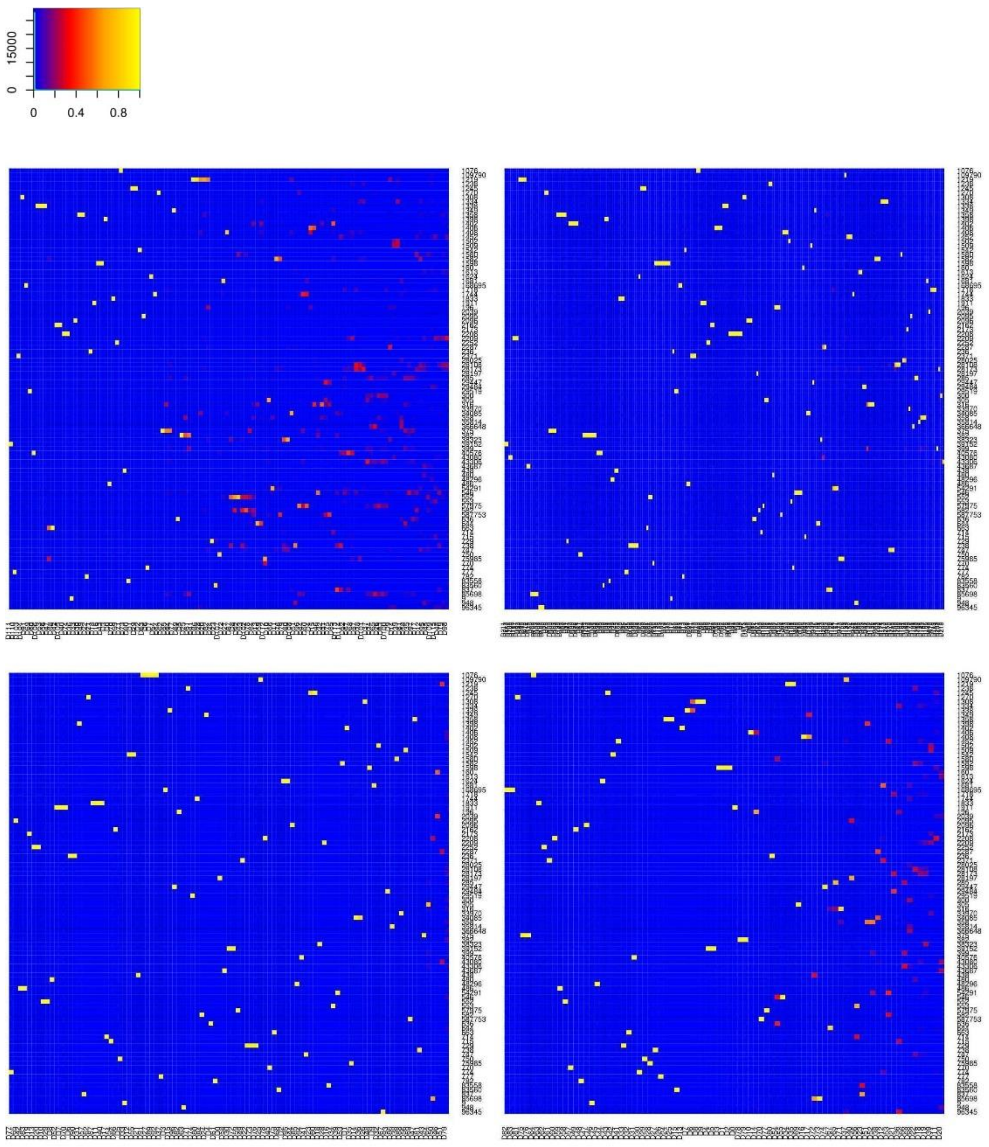
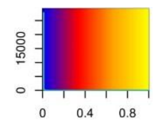


Figure A.36. Heat maps for Strains_20S_12.5R. The top left is that of GroopM, top right of BMC3C, bottom left of MyCC, and bottom right of GATTACA



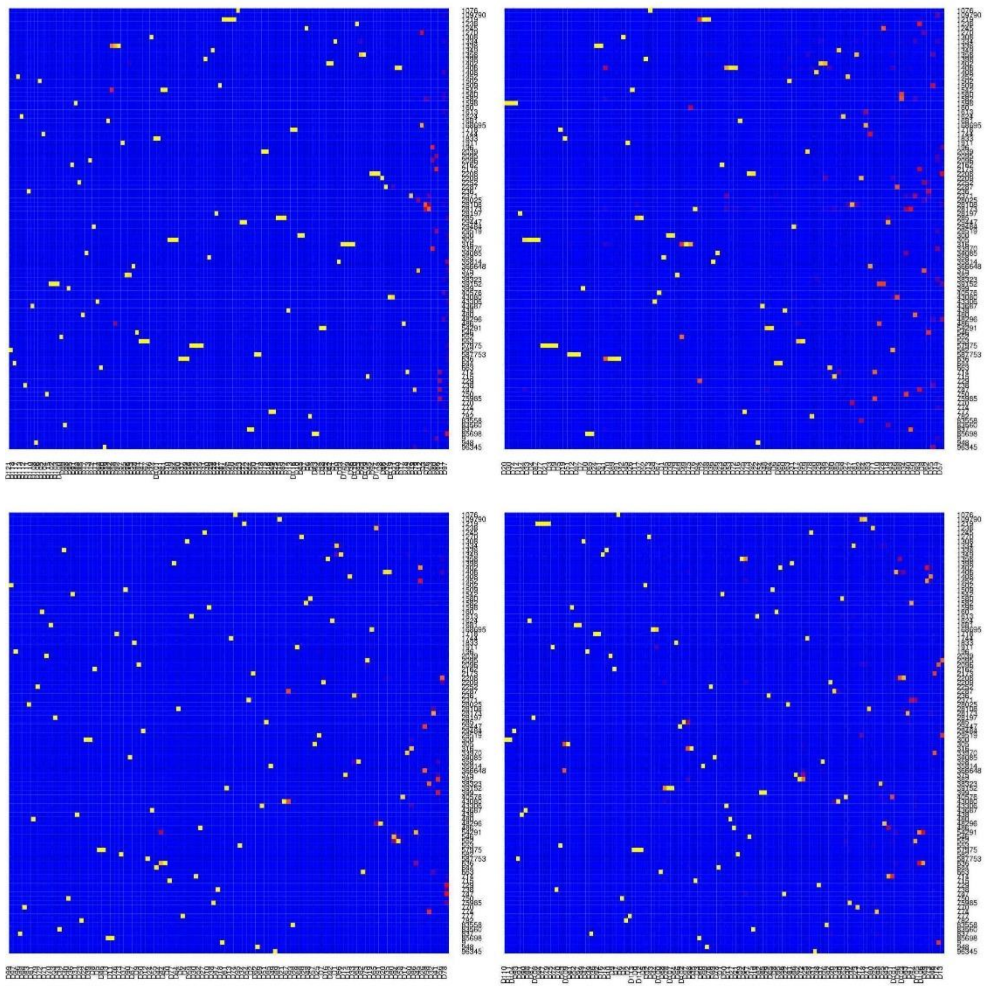


Figure A.37. Heat maps for Strains_40S_12.5R. The top left is that of CONCOCT, top right of COCACOLA, bottom left of MetaBAT, and bottom right of MaxBin2

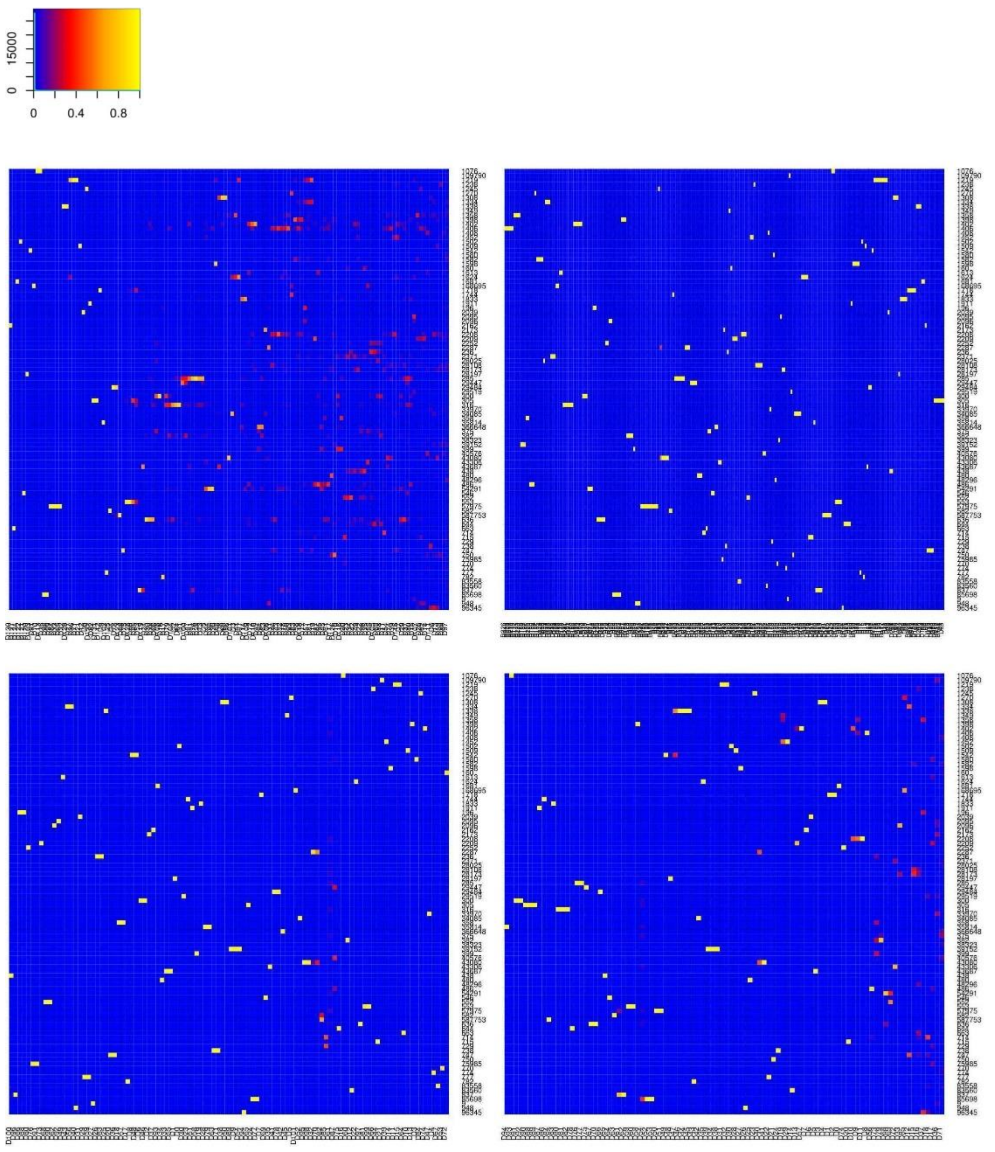


Figure A.38. Heat maps for Strains_40S_12.5R. The top left is that of GroopM, top right of BMC3C, bottom left of MyCC, and bottom right of GATTACA

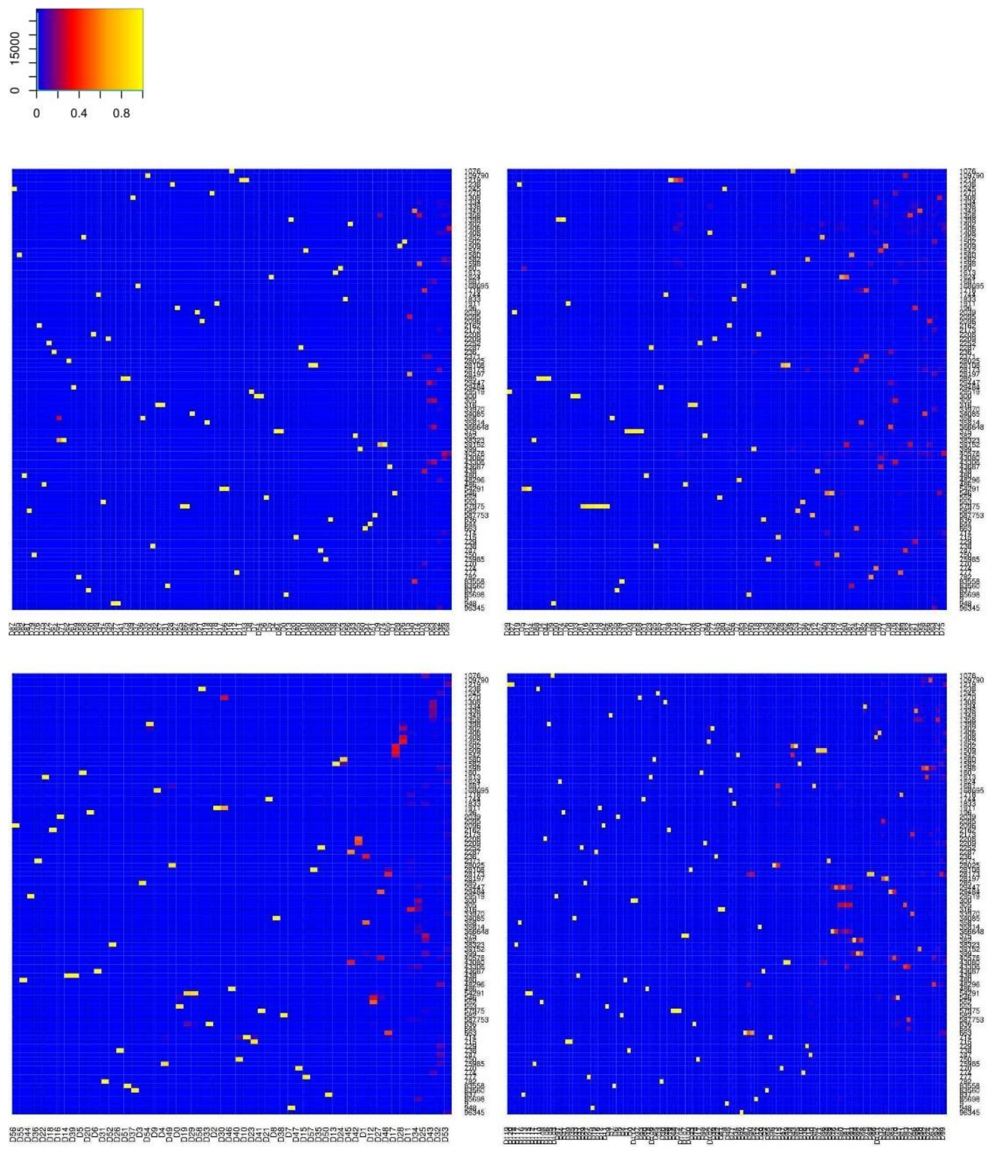
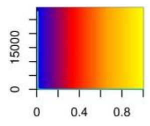


Figure A.39. Heat maps for Strains_60S_2.5R. The top left is that of CONCOCT, top right of COCACOLA, bottom left of MetaBAT, and bottom right of MaxBin2



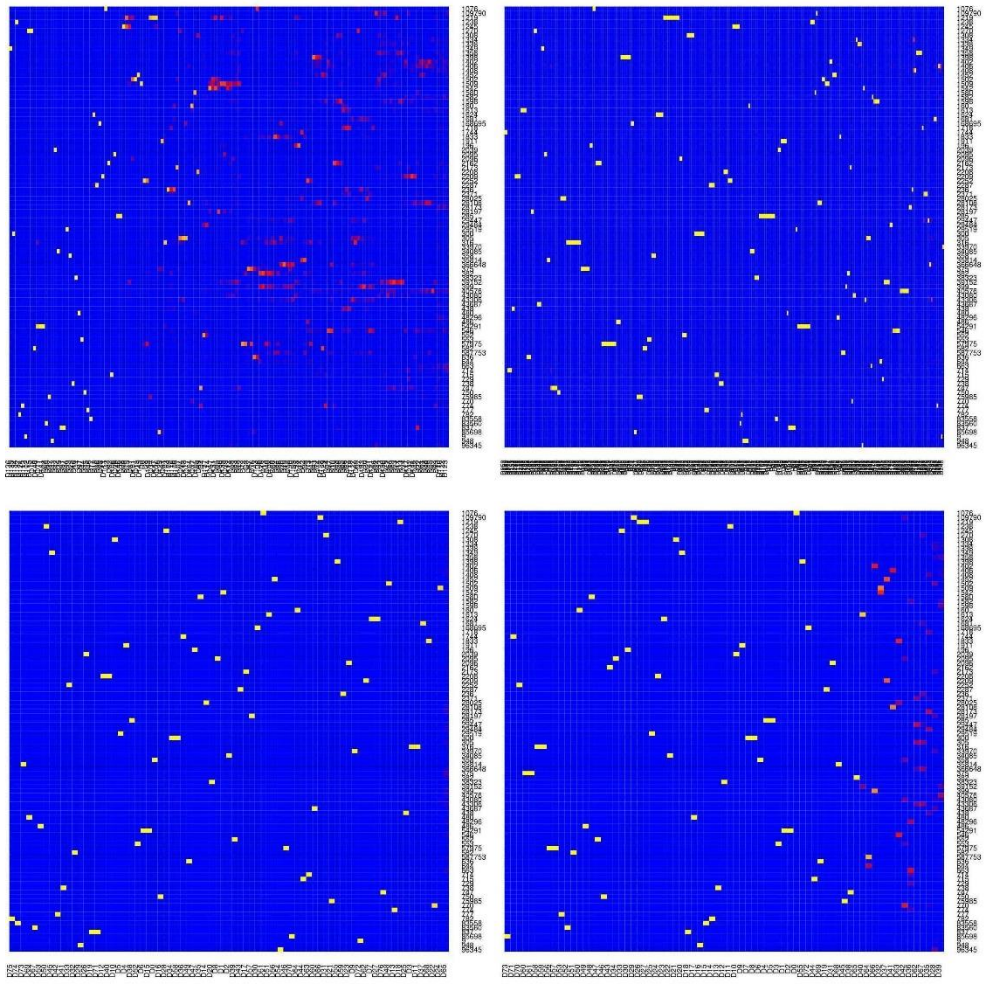


Figure A.40. Heat maps for Strains_60S_2.5R. The top left is that of GroopM, top right of BMC3C, bottom left of MyCC, and bottom right of GATTACA

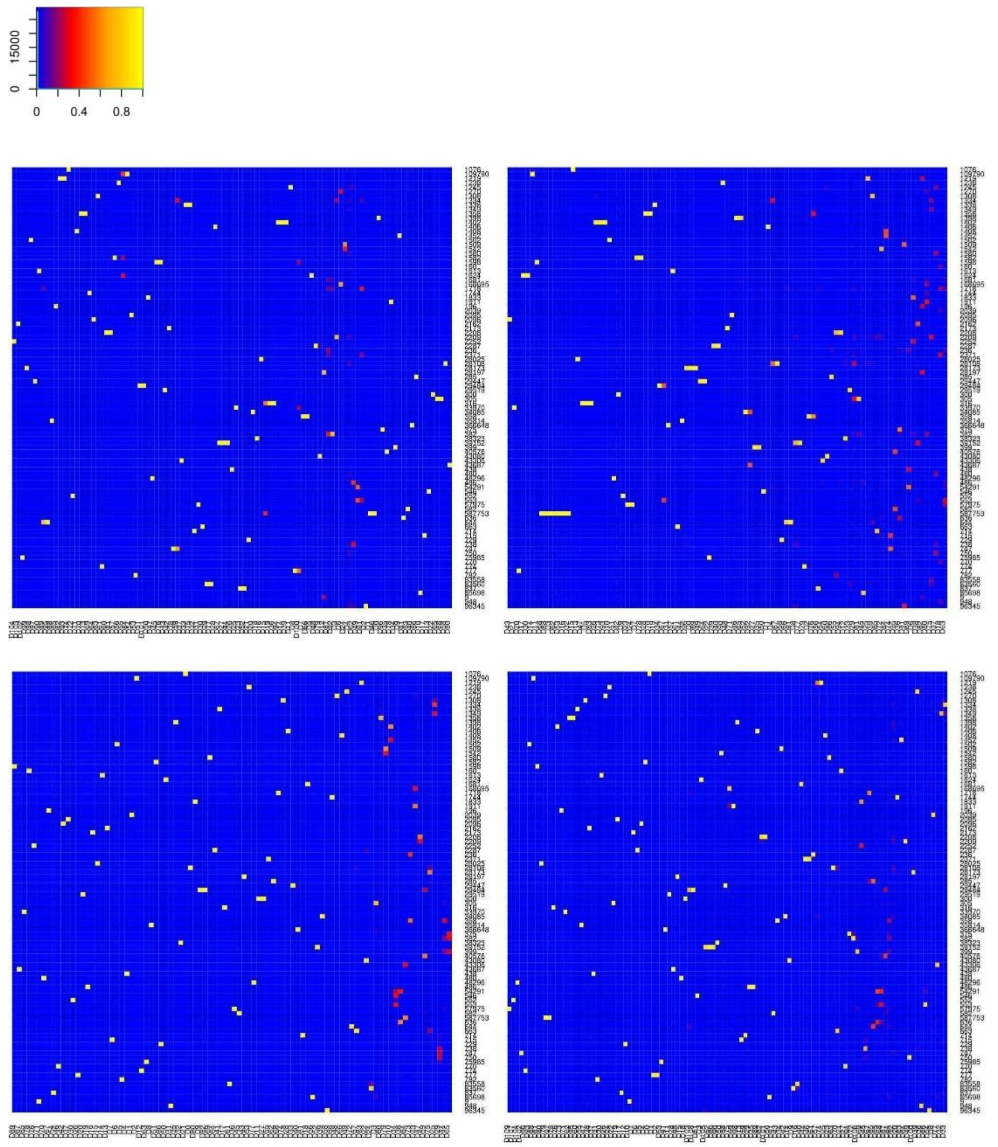
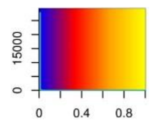


Figure A.41. Heat maps for Strains_60S_5.0R. The top left is that of CONCOCT, top right of COCACOLA, bottom left of MetaBAT, and bottom right of MaxBin2



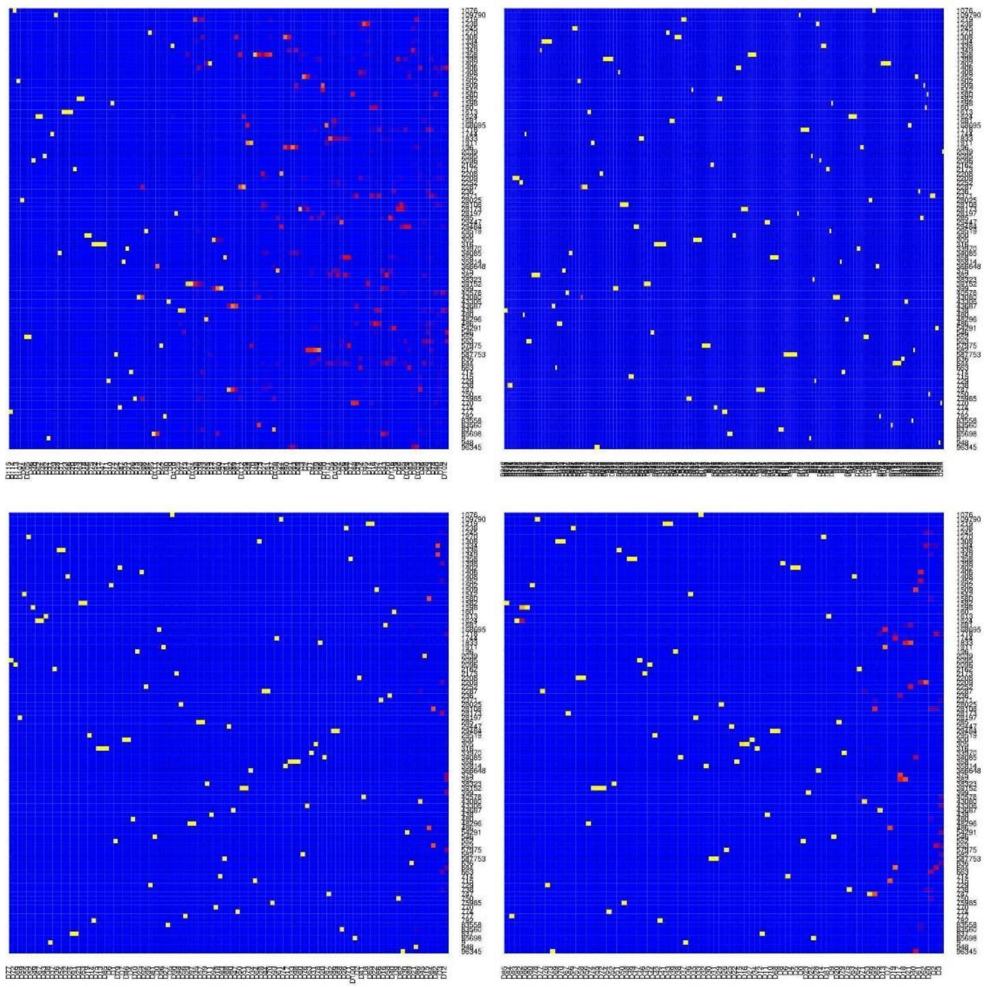


Figure A.42. Heat maps for Strains_60S_5.0R. The top left is that of GroopM, top right of BMC3C, bottom left of MyCC, and bottom right of GATTACA

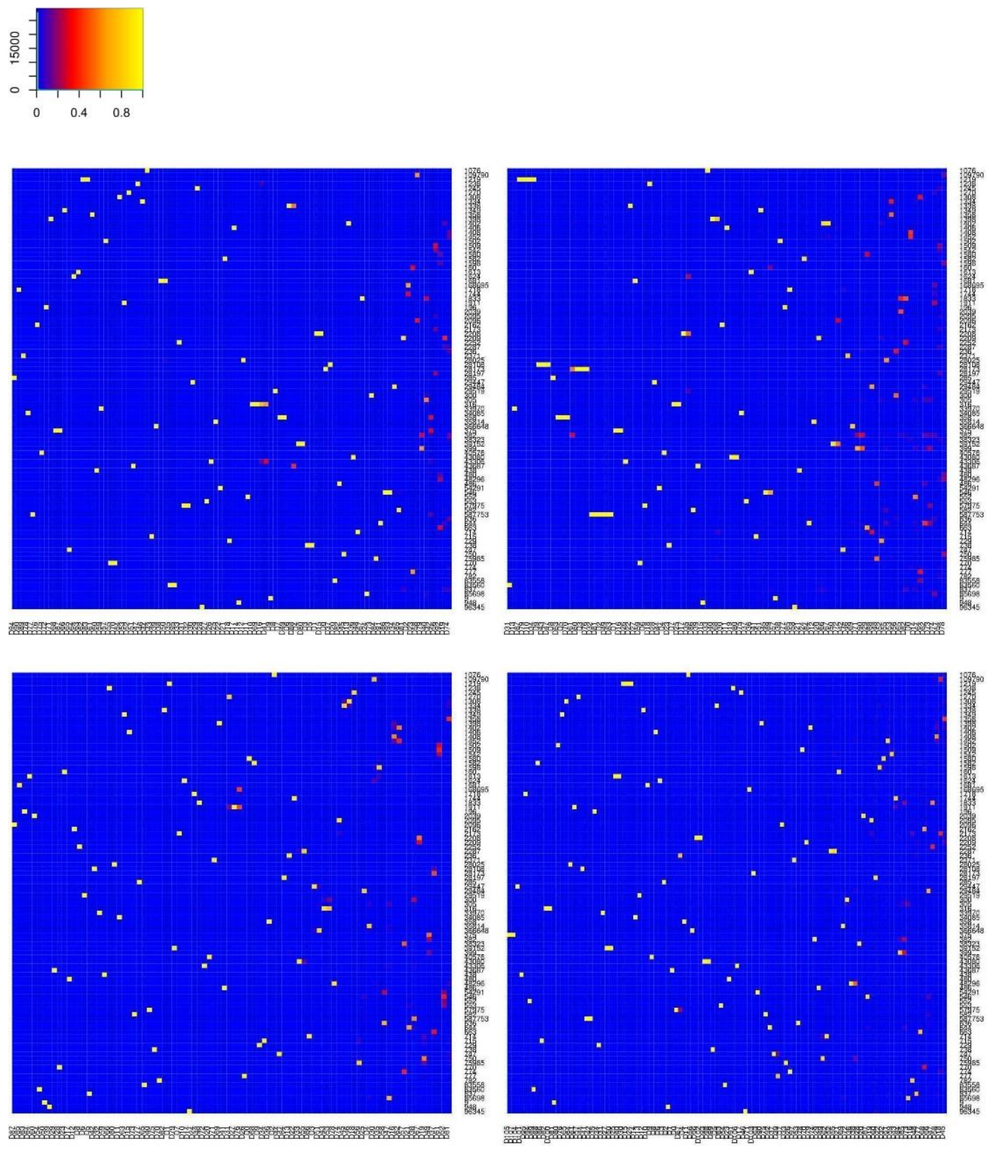
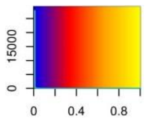


Figure A.43. Heat maps for Strains_60S_7.5R. The top left is that of CONCOCT, top right of COCACOLA, bottom left of MetaBAT, and bottom right of MaxBin2



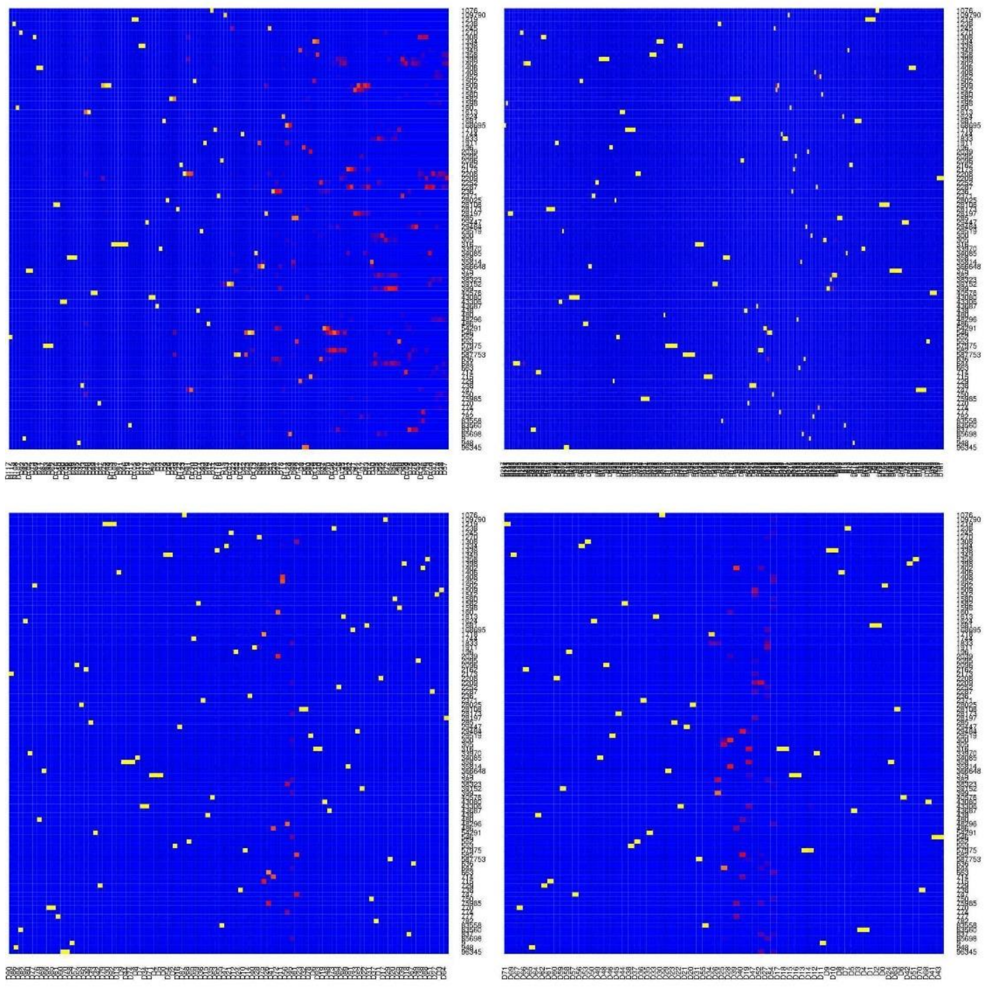
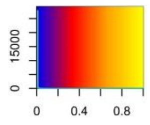


Figure A.44. Heat maps for Strains_60S_7.5R. The top left is that of GroopM, top right of BMC3C, bottom left of MyCC, and bottom right of GATTACA



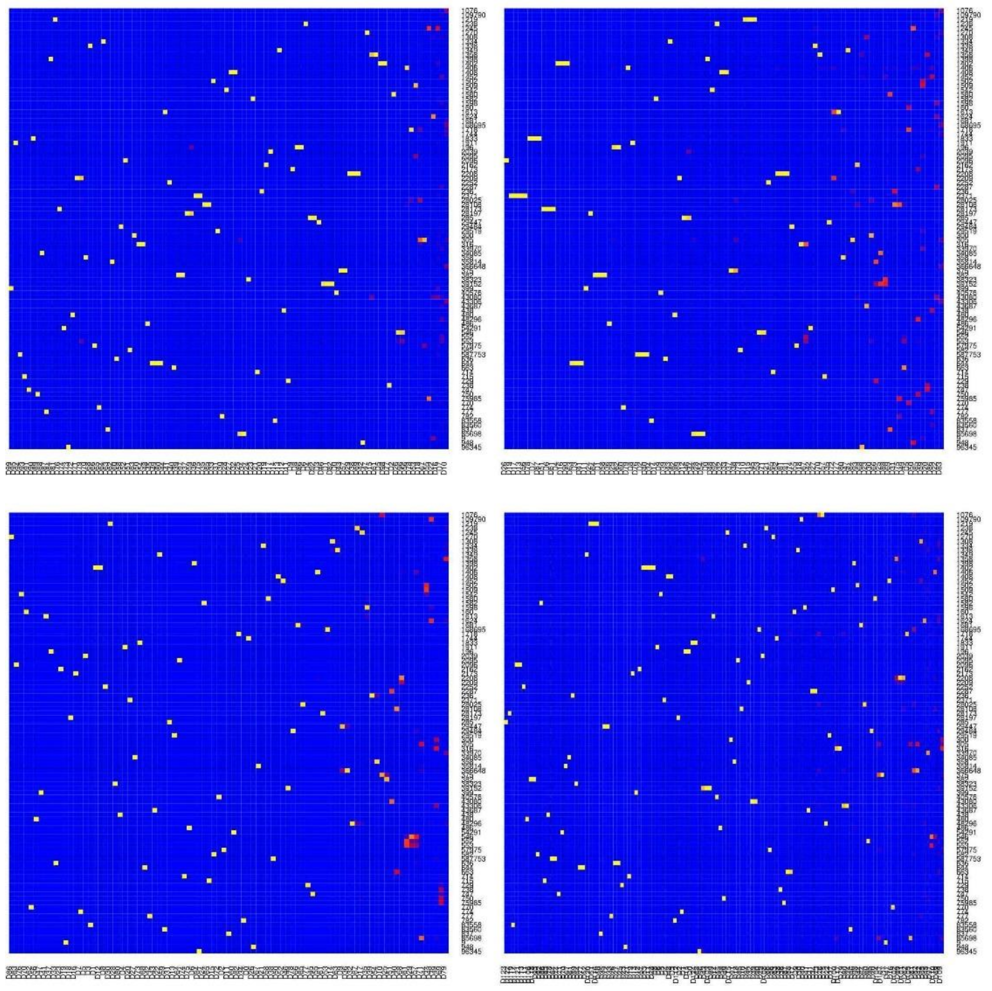
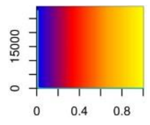


Figure A.45. Heat maps for Strains_60S_10.0R. The top left is that of CONCOCT, top right of COCACOLA, bottom left of MetaBAT, and bottom right of MaxBin2



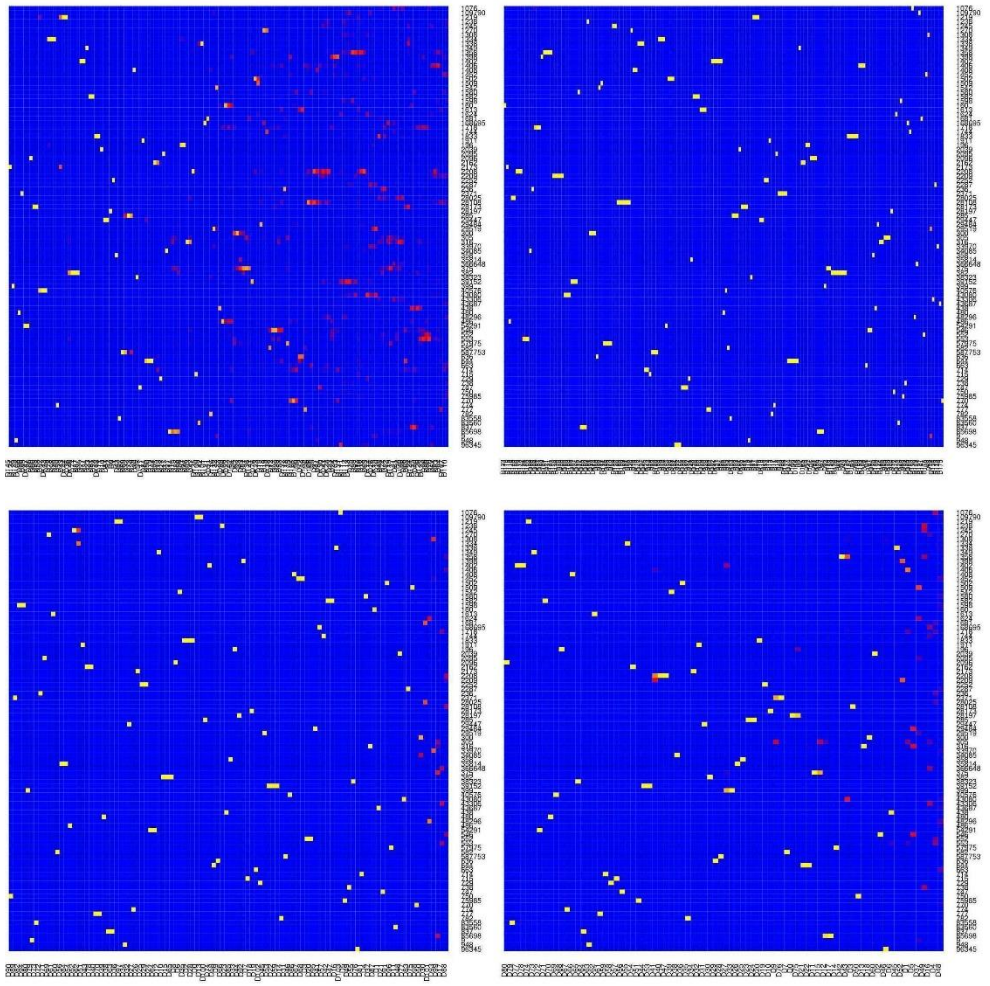


Figure A.46. Heat maps for Strains_60S_10.0R. The top left is that of GroopM, top right of BMC3C, bottom left of MyCC, and bottom right of GATTACA

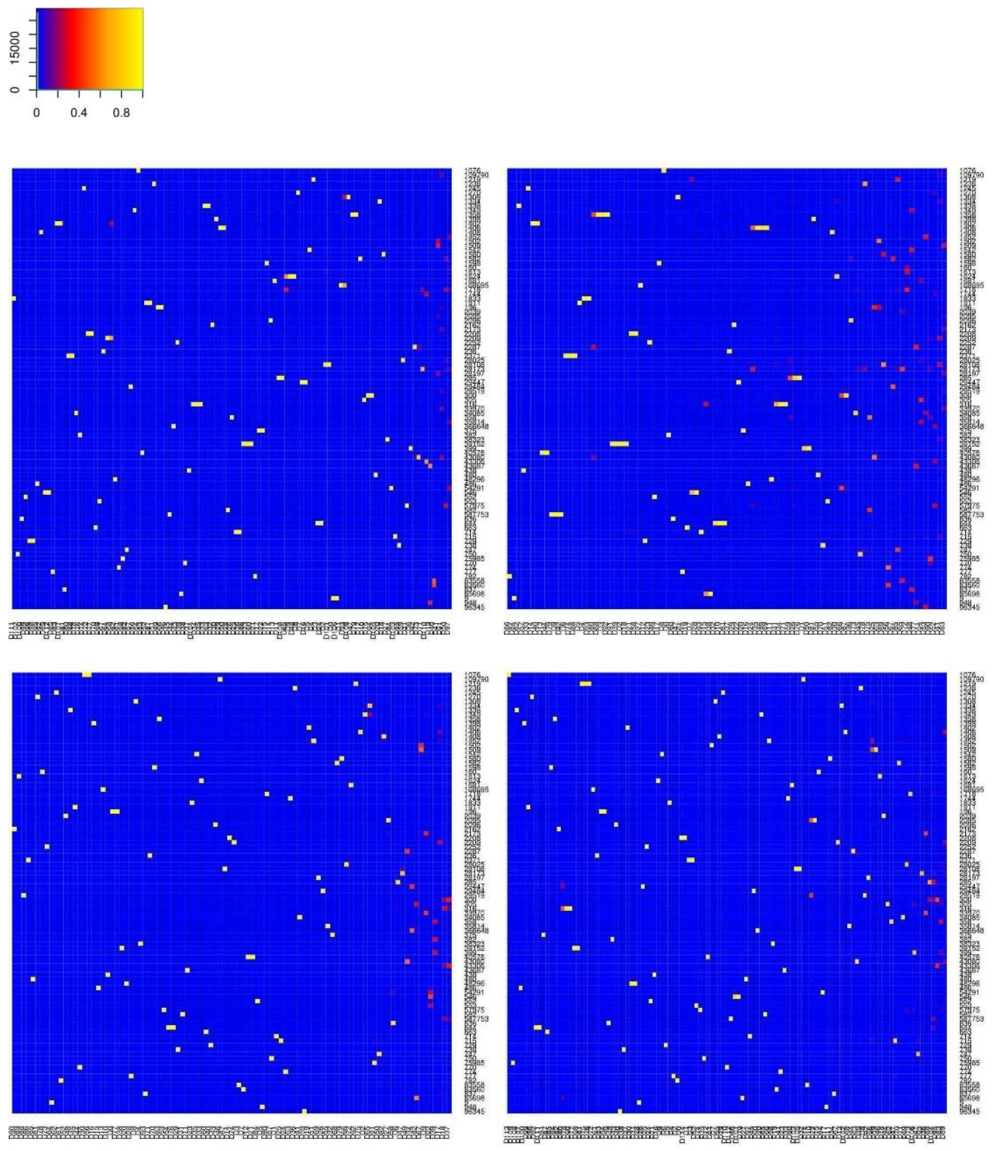


Figure A.47. Heat maps for Strains_60S_12.5R. The top left is that of CONCOCT, top right of COCACOLA, bottom left of MetaBAT, and bottom right of MaxBin2

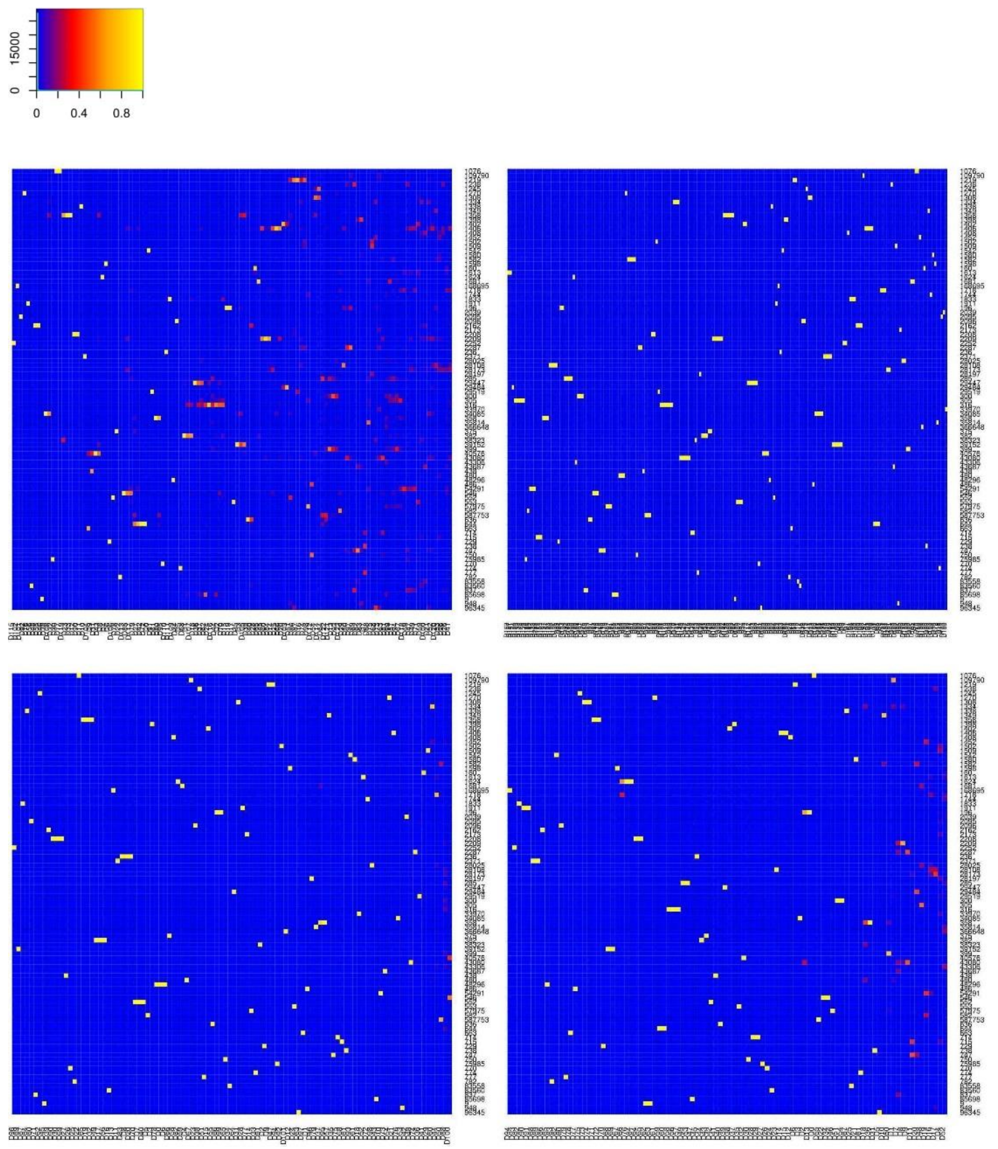


Figure A.48. Heat maps for Strains_60S_12.5R. The top left is that of GroopM, top right of BMC3C, bottom left of MyCC, and bottom right of GATTACA

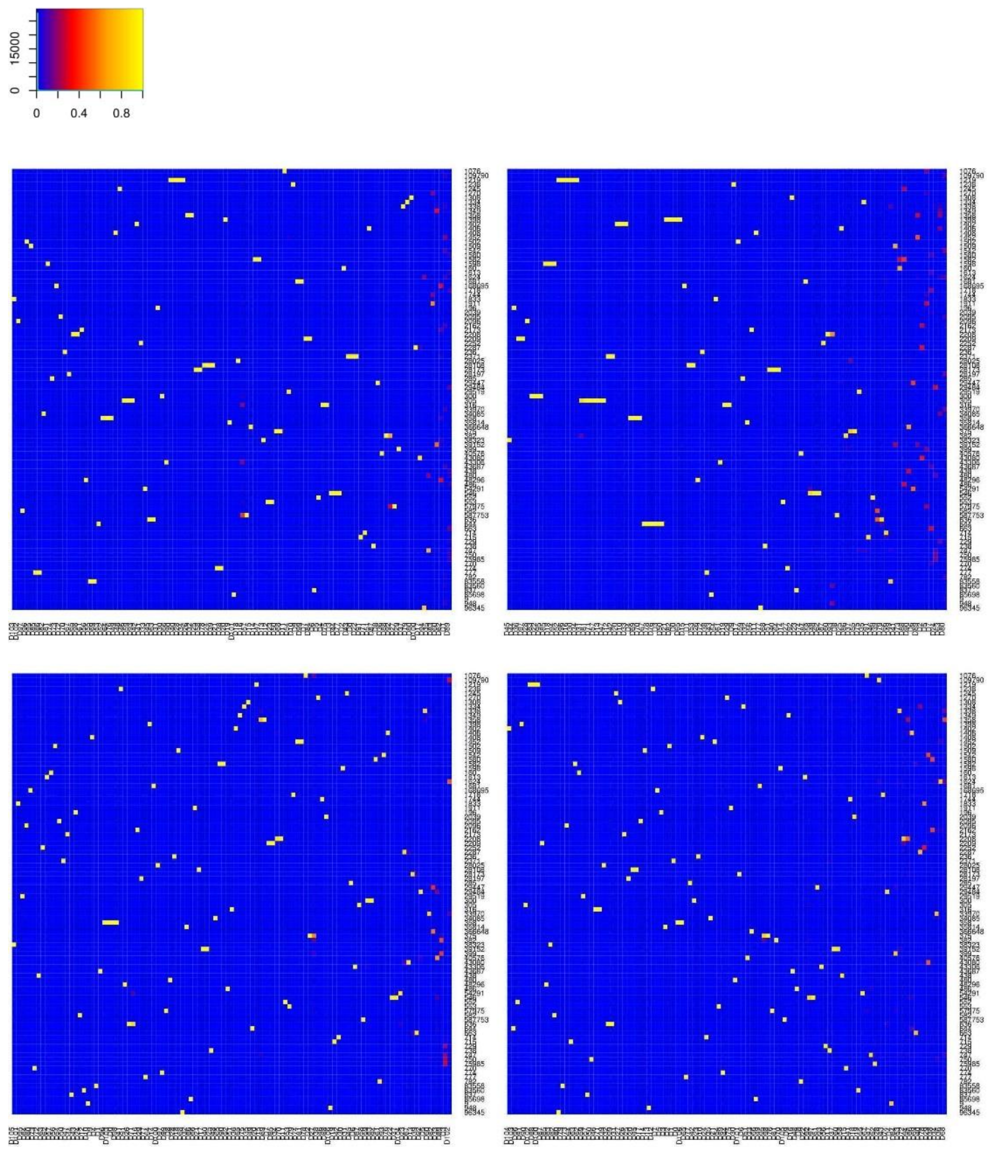


Figure A.49. Heat maps for Strains_80S_12.5R. The top left is that of CONCOCT, top right of COCACOLA, bottom left of MetaBAT, and bottom right of MaxBin2

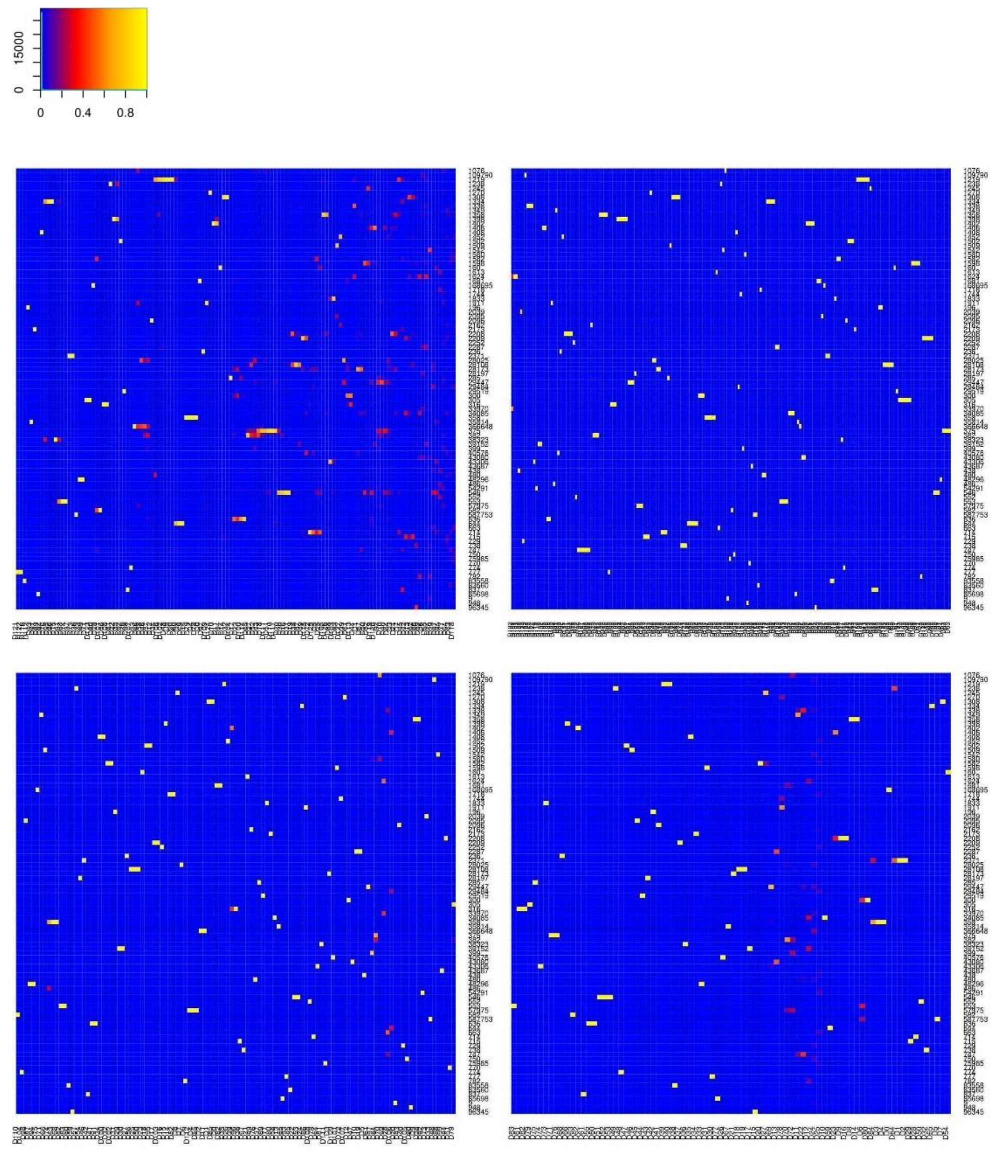


Figure A.50. Heat maps for Strains_80S_12.5R. The top left is that of GroopM, top right of BMC3C, bottom left of MyCC, and bottom right of GATTACA

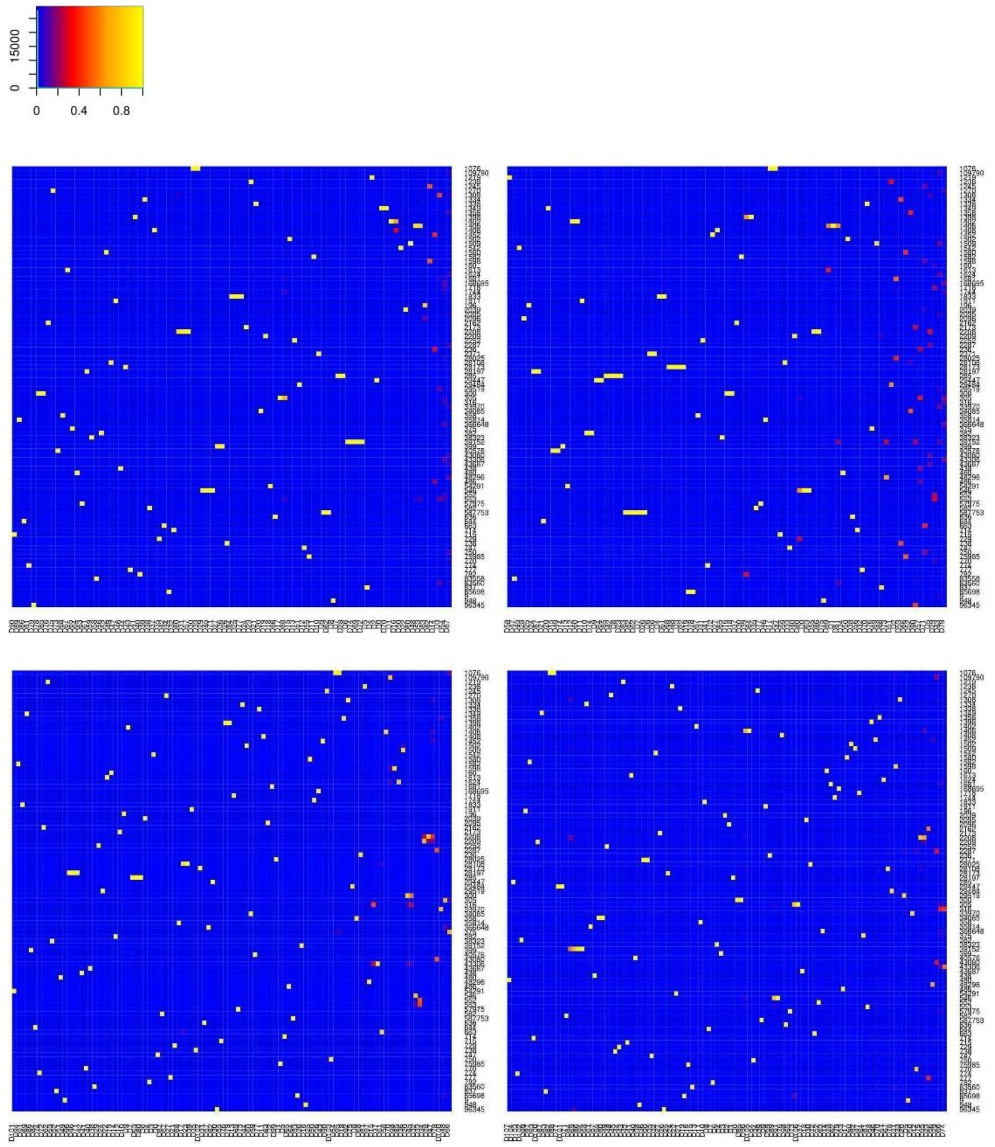
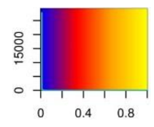


Figure A.51. Heat maps for Strains_100S_12.5R. The top left is that of CONCOCT, top right of COCACOLA, bottom left of MetaBAT, and bottom right of MaxBin2



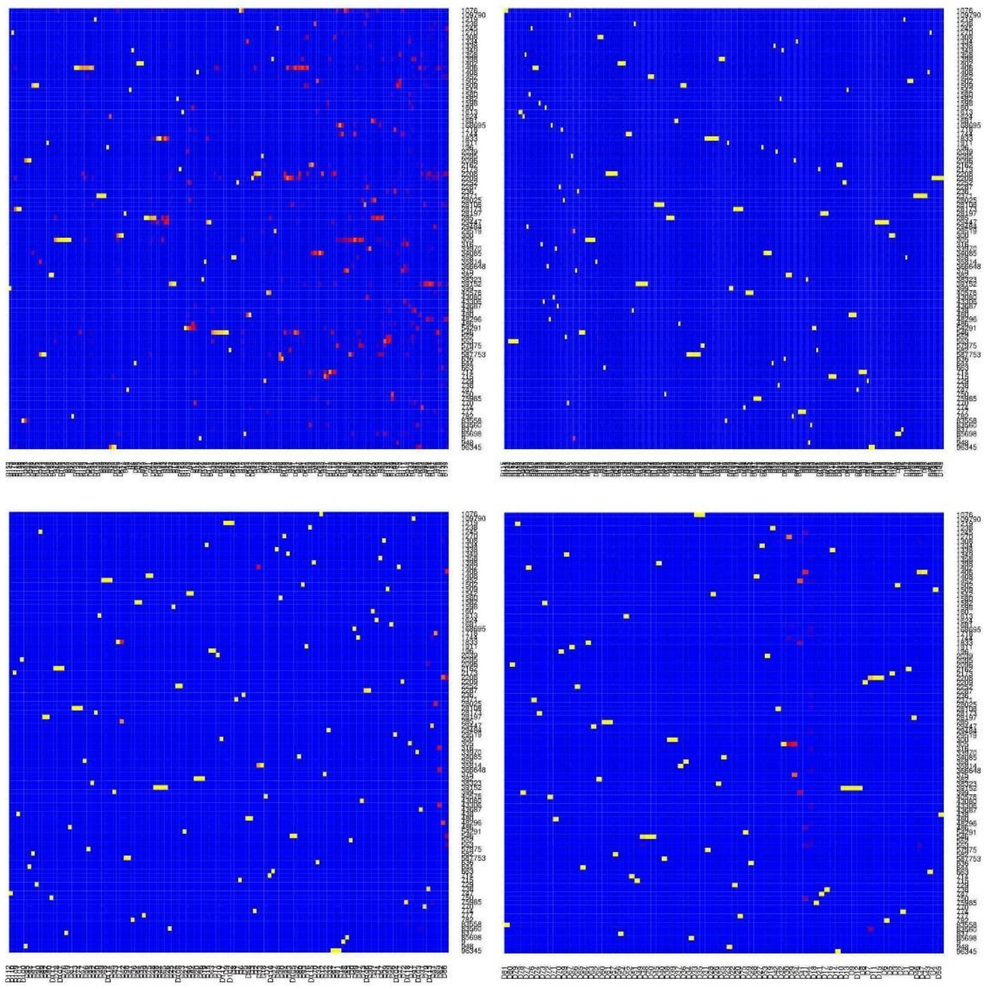


Figure A.52. Heat maps for Strains_100S_12.5R. The top left is that of GroopM, top right of BMC3C, bottom left of MyCC, and bottom right of GATTACA

국문초록

염기서열 결정 기술(시퀀싱)이 발달하면서 자연환경에서 미생물의 염기서열 정보를 직접 얻을 수 있는 메타게놈이 발전했다. 메타게놈을 이용하여 이전에 연구되지 않았던 미생물에 대한 연구를 할 수 있게 되었다. 시퀀싱으로 얻은 미생물 샘플의 염기서열 조각들은 겹치는 구간의 정보를 이용해 긴 가닥의 시퀀스인 콘티그로 합치는 과정인 어셈블리를 거친다. 합치는 어셈블리 과정을 통해 얻는 콘티그들은 시퀀싱으로 얻은 조각들에 비해 긴 길이를 가지지만 게놈 전체의 염기서열을 생성하지는 못한다. 따라서 각 콘티그가 어떤 미생물 종에서 유래한 것인지를 밝히기 위해 콘티그 클러스터링 방법을 사용할 수 있다. 콘티그 클러스터링을 통해 콘티그들을 클러스터에 나눠서 담고, 각 클러스터에 대응하는 미생물 종의 염기서열 정보와 그 종이 샘플 내에서 존재하는 비율을 추정하는 방법이다. 우리는 컨티그 집합의 구성과 커버리지정보를 이용하는 여덟가지의 콘티그 클러스터링 방법들을 비교하였다. 컴퓨터 시뮬레이션을 통해 서로 다른 커버리지와 샘플 수를 가지는 26가지의 데이터를 만든 후, 여덟가지의 방법들을 적용해 보고 각 방법의 성능을 측정하고 분석하였다. 우리는 어떤 방법이 상대적으로 높은 성능을 보이는지, 그리고 각 클러스터링 방법들이 적용되기에 적합한 데이터는 무엇인지 살펴보고자 한다.