Statistical and Computational Methods for Analysis of Shotgun Metagenomics Sequencing Data

Hongzhe Li and Haotian Zheng

1 Introduction

Microbiome consists of all the microorganisms in and on human body. These microbes play important roles in human health and disease. High-throughput shotgun metagenomic sequencing approaches enable genomic analyses of all microbes in a sample, not just those that are amenable to cultivation. In a typical metagenomic sequencing study, an average of 10 million reads are often obtained for a given sample. Such shotgun sequencing reads can be used to profile taxonomic composition and functional potential of microbial communities and to recover whole-genome sequences. Due to complexity and large volume of the data, analysis of shotgun sequencing reads data is more challenging than the marker-gene-based sequencing such as 16S rRNA sequencing in microbiome studies (Quince et al. [28]).

Metagenomic sequencing has wide applications in many areas of biomedical research, including microbiome and disease association studies, diagnosis and treatment of infection diseases, and studies of human host gene expressions and antimicrobial resistance. Depending on the studies and goals, different important microbial features can be derived from shotgun metagenomic data. For example, in disease association studies, useful features can be species abundance, metagenome single-nucleotide polymorphisms (SNPs), metagenome structural variants, and bacterial growth rates. In studies that integrate microbiome and host metabolome, useful features can be collection of all the biosynthetic gene clusters (BGCs).

H. Li (✉) · H. Zheng
Department of Biostatistics, Epidemiology and Informatics, University of Pennsylvania,
Philadelphia, PA, USA
e-mail: hongzhe@upenn.edu; Haotian.Zheng@upenn.edu

S. Datta, S. Guha (eds.), *Statistical Analysis of Microbiome Data*,
Frontiers in Probability and the Statistical Sciences,
https://doi.org/10.1007/978-3-030-73351-3_2

In infectious disease and antimicrobial resistance research, one is interested in identifying new bacterial species or strains that lead to the infectious disease.

The main computational problems in analysis of such shotgun short read data include: (1) binning problem that assigns taxonomic labels to these short DNA reads using sequencing alignment or machine learning methods; (2) quantifying the relative abundances of species, genes, or pathways; (3) metagenomic sequencing assemblies to discover new species; (4) strain-level analysis; and (5) estimation of metabolomic potentials. These computational problems are big data problems that involve merging hundreds of millions of shot sequencing reads with close to 282,580 complete genome sequences of prokaryotes (https://www.ncbi.nlm.nih.gov/genome/browse#!/prokaryotes/). Breitwieser et al. [3] reviewed the methods and databases for metagenomic classification and assembly. Most of the efficient computational tools and software packages have been developed by computational biologists and computer scientists. In this chapter, we summarize and review some of the most commonly used algorithms in the field of microbiome and metagenomic data analysis, focusing on the statistical and computational aspects of the methods, and also point out possible improvements and areas that require further research.

2 Methods for Species Identification and Quantification of Microorganisms

One basic feature of a microbial community is the relative abundance of different species in the community. Given short reads data from shotgun metagenomic sequencing, the first step of analysis is to identify and quantify the relative abundances of all the species in the study samples. This can be achieved by aligning the sequencing reads to the reference genomes. Many computational methods have been developed for taxonomic classification and quantification, see Ye et al. [34] for a benchmarking comparison of various methods in terms of accuracy and computing resources needed. One challenge is how to assign the ambiguous reads that originate from genomic locations shared among multiple groups of organisms.

There are two general approaches to tackle this challenge. The first approach is the marker-gene-based methods where marker genes with sequences that are unique to a clade are identified and reads are only aligned to these marker genes. This method represents taxonomic clades uniquely by sequences that do not share common regions with other clades of the same taxonomic rank. The marker genes can be clade-specific as used in *MetaPhlAn2* (Truong et al. [31]) or universal marker genes as used in *mOTU* (Sunagawa et al. [30]). By aligning reads only to these clade-specific marker genes, the problem of aligning ambiguous reads is solved. *MetaPhlAn2* pipeline has been used in the Human Microbiome Project and the Integrative Human Microbiome Project and is very widely used. *MetaPhlAn2* outputs the taxonomic relative abundance estimation at various taxonomic levels.

The second approach is based on using the full set of reference sequences available as a database and assigning ambiguous reads to their least common ancestor (LCA) in a taxonomic tree. *Kraken* (Wood and Salzberg [33]), a k-mer-based read binning method, is an example of such an approach. *Kraken* uses a database comprising a hash table of k-mers (k is about 31 and should be large) and their corresponding node in a given taxonomic tree. Then, it assigns reads based on where the majority of its k-mers are located in the tree. Whenever no clear vote by the k-mers of the read exists, *Kraken* assigns that read to its least common ancestor. See Fig. 1 for an illustration of the steps of *Kraken*. *Kraken* is a very fast read binning method, which is also often used for taxonomic profiling. After reads are assigned to the taxonomic tree, further processing is needed to estimate the relative abundance of the species in order to account for the uncertainty of the reads that are assigned to the LCA nodes. *Bracken* (Lu et al. [20]) addresses this problem by probabilistically re-assigning reads from intermediate taxonomic nodes to the species level or above.

The output from *Kraken* is read count at each node of the taxonomic tree, similar to read placement for 16s rRNA sequencing reads. One can apply the methods that take into account the taxonomic tree structure in microbiome data analysis. Wang, Cai and Li [32] presented a method that is based on flow on the tree, which can be extended for the data from *Kraken*.

It should be emphasized that shotgun metagenomic sequencing data only provides information on the relative abundance of the species in the community. Such data are compositional and require special care in their analysis (see Li [19] for a review of methods for analysis of microbiome compositional data).

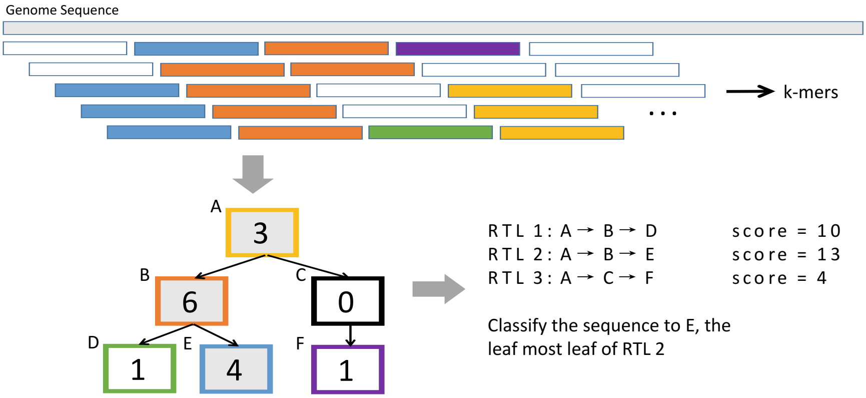

Genome Sequence
k-mers
...
A
3
B
6
C
0
D
1
E
4
F
1
RTL 1: A → B → D score = 10
RTL 2: A → B → E score = 13
RTL 3: A → C → F score = 4
Classify the sequence to E, the leaf most leaf of RTL 2

Fig. 1 Illustration of the *Kraken* algorithm for binning reads to taxon nodes on a taxonomic tree based on k-mer matching (modified from Figure 1 of Wood and Salzberg [33]) The number in each taxon node is the number of k-mers in the sequence that is associated with that taxon. The associated k-mers with each taxon node are marked with the corresponding color. The read sequence is assigned to the left-most leaf on the root-to-leaf (RTL) path with the greatest score, which is defined as the sum of the numbers in the nodes of the RTL path. The resulting tree can be used for taxonomic composition analysis and downstream statistical analysis

3 Metagenome Assembly and Applications

Besides quantifying the relative abundance of known bacterial species, new computational methods have also been developed for metagenome assemblies. The first step of metagenome assembly is to construct longer contiguous sequence based on the overlap of reads, a contig. These contigs are then clustered into bins based on their similarities. The algorithm outputs a large set of metagenome-assembled genomes (MAGs) (see Fig. 2 for an illustration), which are subject to downstream data analysis.

One important computational tool in genome assembly is to store the reads into the de Bruijn graph and to find Eulerian walks in the graph. Due to the large read counts for metagenomic data, metagenome assembly is time- and memory consuming. de Bruijn graph and Eulerian walks are powerful tools in computational genome sequence data analysis, but they are less known among statisticians. We briefly review the key concept in this section and point to the statistical questions.

3.1 de Bruijn Assembly of a Single Genome

de Bruijn graph, which is used widely in genome assembly, is a concept originated from graph theory. An n-dimensional de Bruijn graph of m symbols is basically a directed graph representing overlaps between sequences of symbols. It has m^n vertices, consisting of all possible length-n sequences of the given symbols. If one of the vertices can be expressed as another vertex by shifting all its symbols by one place to the left and adding a new symbol at the end of this vertex, then the latter has a directed edge to the former vertex. In genome assembly, it is explicit to create an assembly graph to illustrate the connecting relationships between reads or contigs. Oftentimes in an assembly graph, nodes represent DNA sequences (unitigs/contigs), while edges represent overlaps between those sequences. An assembly graph represents fundamental uncertainty in possible paths to go through the sequences.

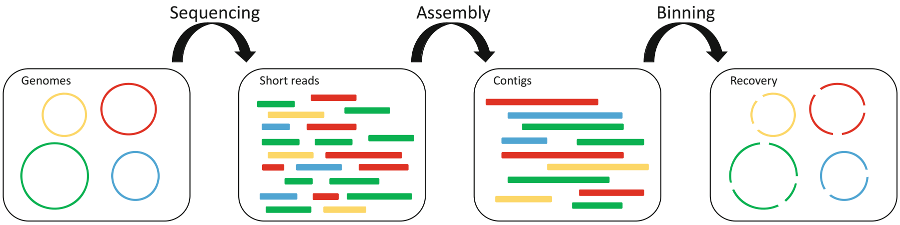


Fig. 2 Illustration of metagenome assembly to metagenome-assembled genomes (MAGs) that include a set of contigs. MAGHIT and MetaBAT2 are two most commonly used packages for assembly and for binning, respectively

de Bruijn graph can be used to construct an assembly graph based on the data of sequencing reads. The key point is to connect two substrings (represented by vertices) in a de Bruijn graph only if there is a read showing one substring can be transformed by shifting all its symbols by one place to the left and adding a new symbol at the end of this substring to another through that read. For instance, if there is a read whose sequence is GCCCA, as well as two substrings GCCC and CCCA, we can add an edge from the vertex representing GCCC to the vertex of CCCA. However, if there is not a read containing GCCCT as a part of it, even if there could be a vertex representing the substring CCCT, we should not add an edge from GCCC to CCCT in the de Bruijn graph. To make a de Bruijn graph consistent inside, we will need reads of length L, and they should overlap by L-1 bases. However, in most of the real cases, neither all reads overlap with each other perfectly, nor all reads have the same length. To resolve those problems, all k-length subsequences of the reads, i.e., the k-mers, are often used in genome assembly.

To construct a de Bruijn graph, we start from dividing each read into several k-mers with a pre-specified k. We traverse all of the k-mers of a given read and form the left $k - 1$-mer (a substring with length $k - 1$) and the right $k - 1$-mer of each k-mer. We include all the possible $k - 1$-mers as vertex in the prospective de Bruijn graph and draw a directed edge from each left $k - 1$-mer to its corresponding right $k - 1$-mer. If the left and right $k - 1$-mers are the same in a k-mer, we will draw an edge to itself. In the illustrative example in Fig. 3, we have three reads, CCCATGTAAG, CCATCTAAGC, and GCCCATCTA. We set $k = 5$ and find all of the 5-mers of the reads. In the first read CCCATGTAAG, all the 5-mers are CCCAT, CCATG, CATGT, ATGTA, TGTAA, and GTAAG. We then get the left and right 4-mers of each 5-mer and draw edges between them. There are edges from CCCA to CCAT, CCAT to CATG, CATG to ATGT, ATGT to TGTA, TGTA to GTAA, and GTAA to TAAG. We then construct a de Bruijn graph with all of the 4-mers of the 3 reads as vertices, which are shown in part (i) of Fig. 3, and draw an edge between two 4-mers if they together form a 5-mer of the reads. The constructed de Bruijn graph with the 3 reads above is shown in part (ii), where each vertex is a 4-mer, and the number in the vertex shows how many times that 4-mer appeared in all of the 3 reads. The letter on each edge indicated how a left 4-mer is transformed into its corresponding right 4-mer that is connected by that edge.

The next step in genome assembly is to find the origin genome sequence in the de Bruijn graph by looking for an Eulerian walk. If we manage to find an Eulerian walk in the de Bruijn graph, we then find the original genome sequence. After we build the de Bruijn graph as in Fig. 3, we next find a walk through it as a contig. In our example, one digit replacement, such as the C replaced by G in read 1, causes a branch of length 4 in the de Bruijn graph. In our example, we cannot find an Eulerian walk that visits each vertex exactly once, so we have to abandon a branch to get a walk through the graph. Here, we abandon the branch with lower frequency, which is defined as the sum of the numbers in the vertices on that branch, and choose the walk or branch with the highest frequency, shown in part (iii) of Fig. 3.

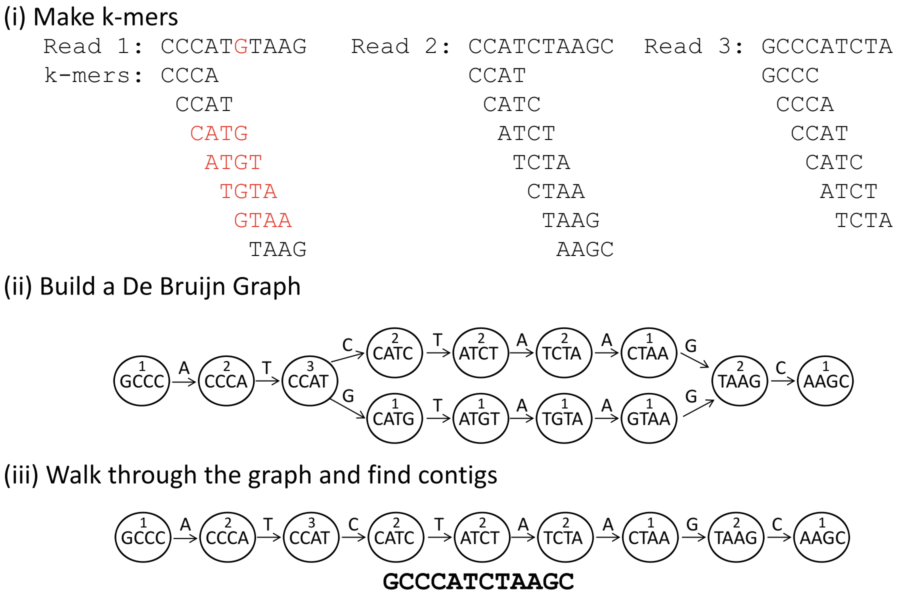


Fig. 3 An illustration of de Bruijn graph for genome assembly of three sequencing reads by using 4-mers, where nodes represent different *k*-mers and the numbers in the node indicate the number of the corresponding *k*-mer observed in the data. The Eulerian walk in the de Bruijn graph recovers the original genome sequence

3.2 *Modification for Metagenome and Metagenome-Assembled Genomes*

Various modifications of the methods for single-genome assembly have been made particularly for metagenome assembly to overcome the challenges of unknown abundance and diversity of the microbial community and related species in the metagenomes. Metagenome assembly graphs are frequently large, with millions of nodes, and require 10s to 100s of gigabytes of RAM for storage. Ayling et al. [2] present a review of various methods for metagenome assembly with short reads. Among various methods, *MEGAHIT* (Li et al. [18]) is most widely used method for contig construction. R package *bgtools* provides an interactive visualization tool for metagenomic bins, which is very useful for statisticians to explore the data (Seah and Gruber-Vodicka [29]).

MetaBAT2 (Kang et al. [15]) is most widely used computational package for binning the contigs. It performs pairwise comparisons of contigs by calculating probabilistic distances based on tetranucleotide frequency and then uses a *k*-medoid clustering algorithm to bin contigs to genomes.

Alternative to *MetaBAT2*, *CONCOCT* (Alneberg et al. [1]) is a binning method based on *k*-mer frequencies of the contigs. For metagenomics data, a co-assembly of reads from all samples is first performed to obtain the set of contigs, which can be further filtered by length, and only contigs greater than a minimum size are used. For samples $j = 1, 2, \ldots, M$, and contigs $i = 1, 2, \ldots, N$, a coverage number Y_{ij} is defined as the average number of reads that are mapped to contig i per base from sample j. For each contig, we get a vector of coverage $Y_i = (Y_{i,1}, \ldots, Y_{i,M})$ over M samples. In addition, a composition number is defined as the frequency for each k-mer and its reverse complement in that contig. For a fixed length k, the dimension of composition would be $V = f(k)$, which is the total number of possible k-mers, where reverse complements are considered as one possible k-mer. So for each contig i, we have its composition vector $Z_i = (Z_{i,1}, \ldots, Z_{i,V})$, where $Z_{i,v}$ is the count of k-mer v that appeared in contig i. After adding pseudo-counts to remove zero in the input, together with normalization and logarithm transformation, a profile for contig i of dimension $E = M + V + 1$ is formed, where 1 comes from the total coverage for a contig in all the samples. *CONCOCT* performs a dimension reduction using principal-component analysis (PCA) and then clusters the contigs into bins using a Gaussian mixture model with a variational Bayesian approximation.

Using *MEGAHIT* and *MetaBAT2*, Pasolli et al. [27] leveraged 9,428 metagenomes to reconstruct 154,723 microbial genomes (45% of high quality) spanning body sites, ages, countries, and lifestyles. They recapitulated 4,930 species-level genome bins (SGBs), 77% without genomes in public repositories (unknown SGBs [uSGBs]). As microbial genomes are available at an ever-increasing pace, as cultivation and sequencing become cheaper, obtaining metagenome-assembled genomes (MAGs) becomes more effective. These unknown SGBs are expected to explain additional variability of the phenotypes of interest. Zhu et al. [35] showed that these reads from unknown organisms significantly increase the prediction accuracy of the disease status.

3.3 Compacted de Bruijn Graph

Shotgun metagenomic data also provide information on strain-level variation or metagenome structural variation. For strain-level analysis of metagenomes, compacted de Bruijn graph provides an efficient way of describing the data, where long simple paths of a de Bruijn graph are compacted into single vertices in order to reduce computational burden of the vast amount of k-mers. Here, the simple path to be compacted is also known as a unitig, which is defined as a path with all but the first vertex having in-degree 1, and all but the last vertex having out-degree 1. Here, the in-degree of a vertex is the number of edges pointing to that vertex in the de Bruijn graph, and the out-degree of a vertex is the number of edges pointing from that vertex. The graph after compaction is called a compact de Bruijn graph (cDBG). In a cDBG, one vertex may represent more than one k-mer, in contrast with one vertex representing one k-mer in a de Bruijn graph.

To illustrate the ideas, Fig. 4a shows a de Bruijn graph. In the path GGCC→GCCC→CCCA, all vertices except for the first one, GGCC, have in-

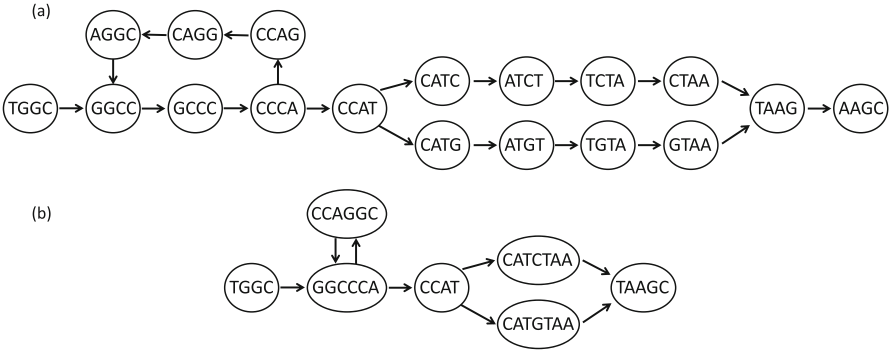


Fig. 4 Illustration of the compact de Bruijn graph (**b**) derived from the *k*-mer-based de Bruijn graph (**a**). For the de Bruijn graph, the nodes are *k*-mers, and for the compact de Bruijn graph, the nodes are contigs

degree equal to 1 (the in-degree of GGCC is 2), and all vertices except for the last one, CCCA, have out-degree 1. Therefore, the path GGCC→GCCC→CCCA is a "simple path" and can be compacted to GGCCCA. Similarly, in the paths CCAG→CAGG→AGGC, CATC→ATCT→TCTA→CTAA, and CATG→ATGT →TGTA→GTAA, all of their vertices have both in-degree and out-degree equal to 1, so they are simple paths and can be compacted to CCAGGC, CATCTAA, and CATGTAA, respectively. After we compacted all of the simple paths in the de Bruijn graph, we obtain the compact de Bruijn graph that is shown in Figure compact (b), where each vertex represents a unitig instead of one *k*-mer.

Chikhi et al. [6] developed an efficient algorithm *BCALM2* to construct the cDBG. There are three main steps to get a compact de Bruijn graph from a set of *k*-mers its correspondingly formed de Bruijn graph operated from metagenome reads. The first step is to distribute the *k*-mers into buckets based on their "minimizers" (defined in Chikhi et al. [6]), with some *k*-mers being thrown into two buckets. Next, each bucket is compacted separately. Finally, the *k*-mers that were thrown into two buckets are glued back together so that duplicates are removed.

Using the compacted de Bruijn graph, Brown et al. [5] developed an efficient graph algorithm for investigating graph neighborhoods of a very large metagenome assembly de Bruijn graph. They developed and implemented a scalable graph query framework for extracting unbinned sequence from metagenome assembly graphs with millions of nodes by exploiting the structural sparsity of compact de Bruijn assembly graphs. These unbinned sequences can be further analyzed to discover new strains and new hidden sequence diversity. One application is to identify the genome neighborhood for a known bacterial genome. The reads from this neighborhood can be assembled and compared with the known genome to identify the strain variability of the known bacterium.

4 Estimation of Growth Rates for Metagenome-Assembled Genomes (MAGs)

The previous section reviews methods for metagenome assembly. In order to make the metagenomic data comparable across different samples, metagenome assembly has to be performed jointly over the combined reads of all the samples. After we obtain the contigs and bins, we usually align the metagenomic reads to each of the contigs to obtain the read coverage for each of the contigs and each of the samples. With appropriate normalization and correcting for possible GC bias, one can quantify the bacterial abundance based on these read coverage data.

Besides the relative abundance information, the uneven read coverage data can be used for estimating the bacterial growth dynamics or replication rates (Korem et al. [16]; Brown et al. [4]; Gao and Li [12]). Such bacterial replication rates provide important insights into the contribution of individual microbiome members to community functions. In a microbiome community, dividing cells are expected to contain, on average, more than one copy of their genome. Since the growing bacterial cells are unsynchronized and contain genomes that are replicated to different extents, we expect to observe a gradual reduction in the average genome copy number from the origin to the terminus of replication (Korem et al. [16]; Brown et al. [4]). This decrease in genome copy number can be detected by measuring changes in DNA sequencing coverage across complete genomes. Figure 5 illustrates this key idea. For the actively dividing bacteria, due to the bidirectional DNA replication from the replication starting sites, the read coverage is expected to decrease along the genome and the rate of decrease can be used to quantify the bacterial replication rate. Korem et al. [16] define the peak-to-trough ratio to quantify the bacterial replication rate for those bacteria with complete genome sequences available.

For MAGs, since we do not know the order of the contigs along the true genome, to estimate the replication rates, one has to first estimate the order of these contigs. Motivated by a simple linear growth model of DNAs, Gao and Li [12] proposed to apply PCA with contigs as observations to estimate the order, which has been shown to be very effective. Consider the following permuted monotone matrix model:

$$Y = \Theta\pi + Z, \tag{1}$$

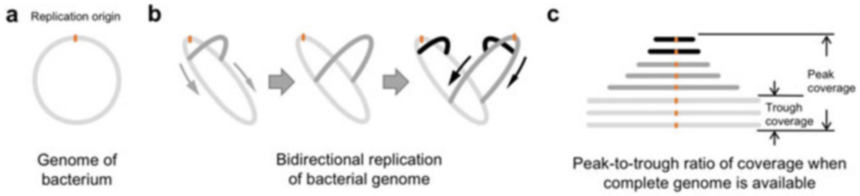

a
Replication origin
Genome of bacterium
b
Bidirectional replication of bacterial genome
c
Peak coverage
Trough coverage
Peak-to-trough ratio of coverage when complete genome is available

Fig. 5 Illustration of bacterial replication rate estimation. Bacterial circular genome (**a**), bidirectional replication (**b**), and peak-to-trough ratio of uneven read coverage (**c**)

where the observed data $Y \in \mathcal{R}^{n\times p}$ is the matrix of the preprocessed contig coverage for a given bacterial species. Specifically, the entry Y_{ij} represents the log-transformed averaged read counts of the j-th contig of the bacterial species for the i-th sample after the pre-processing steps, including genome assemblies, GC adjustment of read counts, and outlier filtering. In practice, the data set is usually high-dimensional in the sense that the number of contigs p far exceeds the sample size n. The signal matrix $\Theta \in \mathcal{R}^{n\times p}$ represents the true log-transformed coverage matrix of n samples and p contigs, where each row is monotone due to the bidirectional DNA replication mechanism. Under the permuted linear growth model, we assume that model (1) holds over the restricted set

$$\mathcal{D}_0 = \left\{ (\Theta, \pi) \in \mathcal{D} \times \mathcal{S}_p : \begin{array}{l} \theta_{ij} = a_i \eta_j + b_i, \text{ where } a_i, b_i \in \mathcal{R} \text{ for } 1 \le i \le n, \\ \eta_j \le \eta_{j+1} \text{ for } 1 \le j \le p-1 \text{ and } \sum_{j=1}^{p} \eta_j = 0. \end{array} \right\}.$$

In other words, each row of Θ has a linear growth pattern with possibly different intercepts and slopes. Under this model, the true coverage matrix is rank-1. We consider the row-normalized observation matrix $X = Y(I_p - \frac{1}{p}ee^\top)$ and its first right singular vector, i.e.,

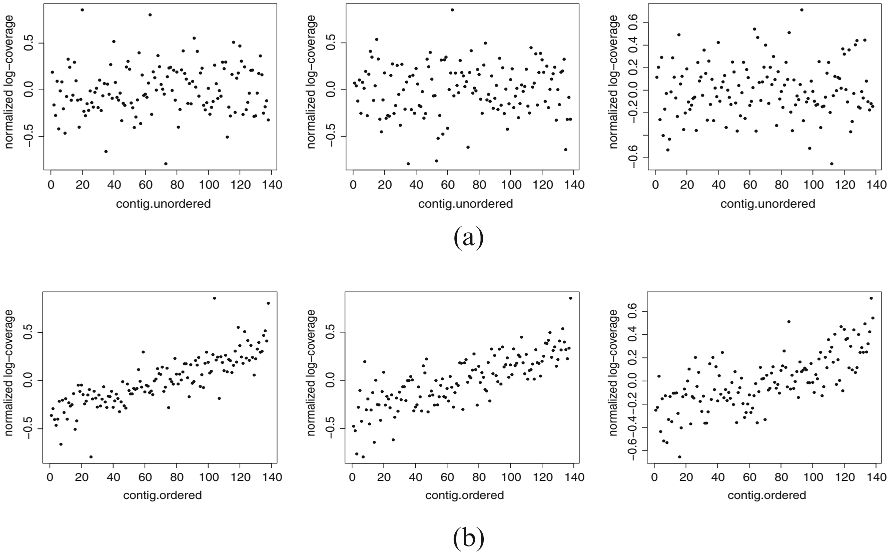
$$\widehat{v} = (\widehat{v}_1, \ldots, \widehat{v}_p)^\top = \underset{v \in \mathcal{R}^p : \|v\|_2 = 1}{\arg\max} \; v^\top X^\top X v.$$

Ma, Cai and Li [21] showed that the order statistics $\{\widehat{v}_{(1)}, \ldots, \widehat{v}_{(p)}\}$ can be used to optimally recover the permutation π, or the original column orders, by tracing back the permutation map between the elements of $\widehat{v}$ and their order statistics.

As an example, Fig. 6a shows the read coverage for one MAG over its contigs for three gut microbiome samples with Crohn's disease from the study of Lewis et al. [17]. We cannot see any patterns of the data. However, after sorting the contigs based on the PCA, we observe a clear monotone pattern of the read coverage (see Fig. 6b). Based on this sorted contig coverage, Gao and Li [12] developed *DEMIC* to estimate the bacterial replication rates for the MAGs. Ma, Cai and Li [21] further showed that the PCA-based estimate of the ordered contigs achieves the minimax rate under certain conditions.

5 Methods for Identifying Biosynthetic Gene Clusters

The next phase of human microbiome research is moving from taxonomic and gene content profiling to functional microbiome by identifying, characterizing, and quantifying microbiome-derived small molecules that are responsible for a specific phenotype. Thousands of functionally interesting small molecules coded by various genes of microbiota have been discovered, including many antibiotics, toxins, pigments, immunosuppressants (Donia and Fischbach [9]). These small molecules



(a)

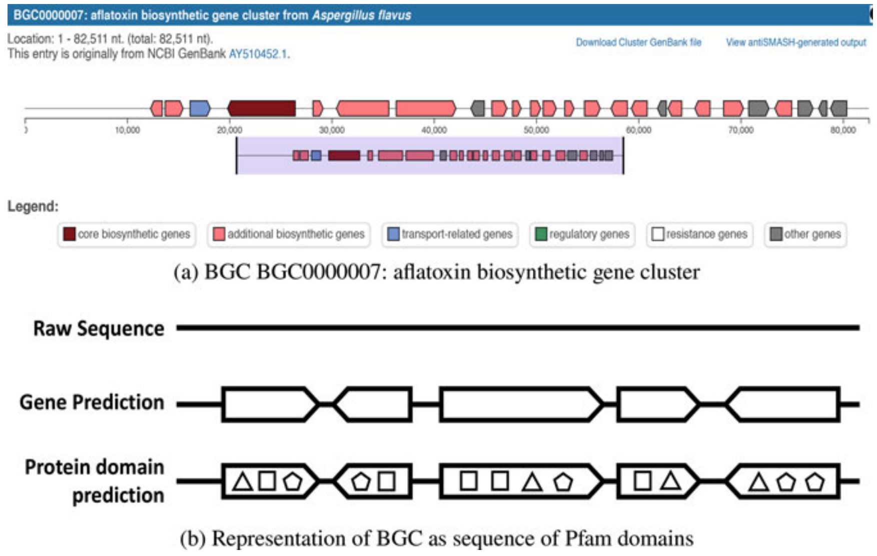
(b)

Fig. 6 Illustration of read coverage for an assembled genome for three gut samples of patients with Crohn's disease, where each dot represents a contig for the assembled genome. Y-axis: log of normalized read coverages. PCA is used to order the contigs based on the coverage over the samples . (**a**) Log read coverage for 3 children with Crohn's disease before contig ordering. (**b**) Log read coverage for 3 children with Crohn's disease after contig ordering

represent a major source of important nature products. Due to the wide range of bioactivities and pharmacological properties, identification of these natural products from microorganisms is an important problem in microbiome research.

The small molecules produced by bacteria are coded by biosynthetic gene clusters (BGCs) discovered along the bacterial genomes. These genes encode enzyme complexes or proteins participating in a common pathway that are continuously clustered in a chromosome region (see Fig. 7a). The BGCs are often collinearly arranged according to their biochemical reaction order (Cimermancic et al. [7]). The chemical and biological mechanisms of known BGCs such as non-ribosomal peptide synthetase (NRPS) and polyketide synthase (PKS) indicate that these multi-domain enzyme complexes are coordinated between the BGC genes. The end products of BGC pathways are bioactive small chemicals or nature products that are diverse in both structures and functions.

The Minimum Information about a Biosynthetic Gene Cluster (MIBiG) database (https://mibig.secondarymetabolites.org) includes an updated list of verified BGCs identified in various microorganisms and provides an important resource for BGC research (Medema et al. [24]). As an example, Fig. 7a shows the structure of BGC BGC0000007: aflatoxin biosynthetic gene cluster from Aspergillus flavus, which includes genes and their functions. New GBCs and their biosynthetic classes have been discovered and deposited into the database based on various experimental methods.


BGC0000007: aflatoxin biosynthetic gene cluster from *Aspergillus flavus*

Location: 1 - 82,511 nt. (total: 82,511 nt).
This entry is originally from NCBI GenBank AY510452.1.

(a) BGC BGC0000007: aflatoxin biosynthetic gene cluster

(b) Representation of BGC as sequence of Pfam domains

Fig. 7 (**a**) Illustration of BGC BGC0000007: aflatoxin biosynthetic gene cluster from Aspergillus flavus. https://mibig.secondarymetabolites.org/repository/BGC0000007/index.html#r1c1. (**b**) A BGC presented as a sequence of protein family (Pfam) domains (modified based on (Hannigan et al. [13]))

The BGCs listed under MIBiG are used in various computational methods for identifying new BGCs and predicting their classes, among which *ClusterFinder* and *DeepBGC* are the two state-of-the-art methods. Both *ClusterFinder* and *DeepBGC* are developed for identifying the BGCs in the bacteria with known complete genome sequences. *ClusterFinder* and *DeepBGC* use the Pfam domain sequential order information in BGC and non-BGC sequences in making the predictions. Specifically, raw genomic sequences are used for gene/ORF prediction using tools like *Prodigal* (Hyatt et al. [14]), and the Pfam domains are assigned to each ORF using *hmmscan* (Eddy [10]). Each BGC is then represented as a sequence of Pfam domains (see Fig. 7b for an illustration).

5.1 A Hidden Markov Model-Based Approach

Cimermancic et al. [7] developed a HMM probabilistic model (*ClusterFinder*), which provided a general solution for BGC identification for both well-studied and novel BGC classes. Using known gene annotations and predicted open reading frames (ORFs), *ClusterFinder* models the data at the protein family domain levels (Pfam) (Fig. 7b) and implements a standard two-stage HMM for estimating the posterior probability of being a BGC for each Pfam domain along the genome, where the emission probabilities are simply the probabilities of observing a particular Pfam domain in BGCs and in non-BGC background. These probabilities are pre-estimated using the training data. HMM then estimates the posterior probability of being in BGC for each of the Pfam domain. The posterior probabilities are further processed to identify the BGCs.

Using *ClusterFinder*, they performed a systematic screening of BGCs in over 1000 bacterial genomes throughout the prokaryotic tree of life and revealed a striking finding of the predominance of Saccharides, a BGC class that has been overlooked in previous research. Compared to the traditional lab-based methods for BGC identification, their work shed light on the possibility of discovering unknown BGCs using computational methods, even for the less studied BGC classes.

5.2 A Deep Learning Approach

Following a similar setting as *ClusterFinder*, *DeepBGC* is the first attempt to employ nature language processing (NLP) and deep learning (DL) strategy for improved BGC identification (Hannigan et al. [13]), where the Pfam sequences of known BGCs and non-BGC are treated as labeled text data, with the Pfam names serving as words of the texts. As commonly used in DL and NLP, Word2Vec is used to learn word (Pfam domain names) embeddings with shallow two-layer neural network and outputs a set of numerical vectors. Word2Vec groups the vectors of similar Pfams together in vector space, where it detects similarities mathematically. Word2Vec creates vectors that are numerical representations of word features such as the context of individual Pfam. *ClusterFinder* then applies the bidirectional long short-term memory (BiLSTM) deep learning model to build predictive model for BGC vs. non-BGC. They showed *DeepBGC* outperformed *ClusterFinder* in both AUC and precision recall in detecting the BGCs on the same validation set. Unlike *ClusterFinder*, *DeepBGC* uses the Pfam domain sequential order information in BGC and non-BGC sequences in making the predictions. Specifically, each Pfam name is numerically coded using Pfam2vec trained using the Pfam names. The BiLSTM outputs classification score for each domain, and the domain scores are summarized across genes, which are selected accordingly as the BGCs. They showed improved performance of *DeepBGC* over the *ClusterFinder*.

5.3 BGC Identification Based on Metagenomic Data

Since both *ClusterFinder* and *DeepBGC* have limited their predictions of the BGCs in the bacteria with known complete genome sequences, with new metagenomic data being generated in very large scale, a logical next step is to identify possible new BGCs based on shotgun metagenomic data. Research in this direction is very limited.

One straightforward approach is to first perform metagenome assembly using the methods introduced in Sect. 3.2 and then apply methods such as *ClusterFinder* or *DeepBGS* to the genome assemblies. This approach was recently explored by Cuadrat et al. [8] to recover BGCs using metagenomic data sampled from Lake Stechlin. One limitation with this assembly-based method is that some BGCs might be scattered through multiple contigs, which make the direct application of *DeepBGC* or *ClusterFinder* infeasible, especially in the post-processing steps when the Pfam-specific predictions are combined into BGCs. Since the contigs in shotgun metagenomics are often short, the existing tools may fail to predict a large fraction of long BGCs.

Meleshko et al. [25] developed *biosyntheticSPAdes*, a tool for predicting BGCs in assembly graphs. This algorithm does not assume that each BGC is encoded within a single contig in the genome assembly, a condition that is violated for most sequenced microbial genomes where BGCs are often scattered through several contigs, making it difficult to reconstruct them. *biosyntheticSPAdes* involves identifying the Pfam domain edges in the assembly graph using HMMER (Eddy [11]), extracting BGC subgraphs, and restoring collapsed domain in the assembly graphs. This is another interesting application of the de Bruijn graph.

6 Future Directions

Shotgun metagenomics have an increasingly important part to play in diverse biomedical applications. We have reviewed some statistical and computational methods for analyzing the shotgun metagenomic data in microbiome studies, focusing more on the computational tools. We feel that it is important to understand how the raw sequencing reads data are processed to summarize the metagenomic data into biologically relevant features in order to understand the uncertainty and possible bias of such estimates. By using statistical inference ideas, we can improve some existing methods. For example, *DEMIC* (Gao and Li [12]) improves *iRep* (Brown et al. [4]) in estimating the bacterial replication rates by using the data across all samples in order to determine the contig order along the genome. Ma, Cai and Li [21] developed a permuted monotone matrix model and provided a theoretical justification of using the first right singular vector in ordering the contigs. They further showed that such a procedure is minimax rate optimal.

Although the methods we reviewed were largely developed by computational biologists or computer scientists, we think that statisticians should be more involved in these initial data processing steps as measurement determines downstream data analysis. When processing the raw sequencing data, we should be aware of the experimental bias, measurement errors, and possible batch effects. As an example, McLaren, Willis and Callahan [23] observed that the measured relative abundances within an experiment are biased by unknown but constant multiplicative factors. When bias acts consistently in this manner, it can be accounted for through the use of bias-insensitive analyses such as ratio-based methods or corrected by a calibration procedure.

We can also make larger impact to metagenomic data analysis by further improving some of the methods based on either the intermediate or the final outputs from these efficient computational methods. For example, after we have the read placements on the taxonomic tree using *Kraken*, we may develop better statistical methods for quantifying the species abundance or identifying the bacterial taxa that are associated with outcomes. After we summarize the metagenomic data as k-mer counts using algorithm such as *JELLYFISH* (Marcais and Kingsfors [22]), we can develop methods to analyze such very large and potentially sparse count tables. Such alignment-free methods have recently been explored by Zhu et al. [35], who showed improvements in predicting diseases using the unaligned reads. Menegaux and Vert [26] proposed to bin together k-mers that appear together in the sequencing reads by learning a vector embedded for the vertices of a compacted de Bruijn graph, allowing us to embed any DNA sequence in a low-dimensional vector space where a machine learning system can be trained.

One challenge in analyzing metagenomic data is the volume of the data that requires large storage and computing power. Although great efforts have been devoted to improve the computation efficiency, for a typical metagenomic study of hundreds of subjects, it takes days to process the data using either *Kraken* or genome assembly. It is also very time-consuming to obtain the intermediate data such as counts of all 31-mers in a metagenomic sample used in *Kraken* algorithm or to construct the de Bruijn graph for shotgun data. Another challenge faced by statisticians is how to effectively access and utilize the data in the public domains, for example, all the BGCs and related information in the BGC repository (https://mibig.secondarymetabolites.org/repository) and the complete genome sequences of all the bacterial genomes.

Acknowledgments This work is supported by NIH grants GM123056 and GM129781.

References

1. Alneberg, J., Bjarnason, B., de Bruijn, I. et al.: Binning metagenomic contigs by coverage and composition. Nature Methods **11**, 1144–1146 (2014)
2. Ayling, M., Clark, M.D., Leggett, R.M.: New approaches for metagenome assembly with short reads. Brief. Bioinform. **21**(2), 584–594 (2020)
3. Breitwieser, F.P., Lu, J., Salzberg, S.L.: A review of methods and databases for metagenomic classification and assembly. Brief. Bioinform. **20**(4), 1125–1136 (2019)
4. Brown, C.T., Olm, M.R., Thomas, B.C., Banfield, J.F.: Measurement of bacterial replication rates in microbial communities. Nature Biotechnology **34**(12), 1256–1263 (2016)
5. Brown, C.T., Moritz, D., O'Brien, M.P., Reidl, F., Reiter, T., Sullivan, B.D.: Exploring neighborhoods in large metagenome assembly graphs using spacegraphcats reveals hidden sequence diversity. Genome Biology **21**, 164 (2020)
6. Chikhi, R., Limasset, A., Medvedev, P.: Compacting de Bruijn graphs from sequencing data quickly and in low memory. Bioinformatics **32**(12), i201–i208 (2016)

7. Cimermancic, P., Medema, M.H., Claesen, J., Kurita, K., Brown, L.C.W., Mavrommatis, K., Pati, A., Godfrey, P.A., Koehrsen, M., Clardy, J., Birren, B.W., Takano, E., Sali, A., Linington R.G., Fischbach, M.A.: Insights into secondary metabolism from a global analysis of prokaryotic biosynthetic gene clusters. Cell **158**(2), 412–421 (2014)
8. Cuadrat, R.R.C., Ionescu, D., Dávila, A.M.R., Grossart, H.P.: Recovering genomics clusters of secondary metabolites from lakes using genome-resolved metagenomics. Front. Microbiol. **9**, 251 (2018)
9. Donia, M.S., Fischbach, M.A.: Small molecules from the human microbiota. Science **349**(6246), 125476 (2015)
10. Eddy, S.R.: Profile hidden Markov models. Bioinformatics **14**(9), 755–63 (1998)
11. Eddy, S.R.: Accelerated profile HMM searches. PLoS Comput. Biol. **7**, e1002195 (2011)
12. Gao, Y., Li, H.: Quantifying and comparing bacterial growth dynamics in multiple metagenomic samples. Nature Methods **15**, 1041–1044 (2018)
13. Hannigan, G.D., Prihoda, D., Palicka, A., Soukup, J., Klempir, O., Rampula, L., Durcak, J., Wurst, M., Kotowski, J., Chang, D., Wang, R., Piizzi, G., Temesi, G., Hazuda, D.J., Woelk, C.H., Bitton, D.A.: A deep learning genome-mining strategy for biosynthetic gene cluster prediction. Nucleic Acids Res. **47**(18), e110 (2019)
14. Hyatt, D., Chen, G., LoCascio, P.F. et al.: Prodigal: prokaryotic gene recognition and translation initiation site identification. BMC Bioinformatics **11**, 119 (2010)
15. Kang, D.D., Li, F., Kirton, E., Thomas, A., Egan, R., An, H., Wang, Z.: MetaBAT 2: an adaptive binning algorithm for robust and efficient genome reconstruction from metagenome assemblies. PeerJ **7**, e7359 (2019)
16. Korem, T., Zeevi, D., Suez, J., Weinberger, A., Avnit-Sagi, T., Pompan-Lotan, M., Matot, E., Jona, G., Harmelin, A., Cohen, N., Sirota-Madi, A., Thaiss, C.A., Pevsner-Fischer, M., Sorek, R., Xavier, R., Elinav, E., Segal, E.: Growth dynamics of gut microbiota in health and disease inferred from single metagenomic samples. Science **349**(6252), 1101–1106 (2015)
17. Lewis, J.D., Chen, E.Z., Baldassano, R.N., Otley, A.R., Griffiths, A.M., Lee, D., Bittinger, K., Bailey, A., Friedman, E.S., Hoffmann, C., Albenberg, L., Sinha, R., Compher, C., Gilroy, E., Nessel, L., Grant, A., Chehoud, C., Li, H., Wu, G.D., Bushman F.D.: Inflammation, antibiotics, and diet as environmental stressors of the gut microbiome in pediatric Crohn's disease. Cell Host Microbe **18**(4), 489–500 (2015)
18. Li, D., Liu, C.M., Luo, R., Sadakane, K., Lam, T.W.: MEGAHIT: an ultra-fast single-node solution for large and complex metagenomics assembly via succinct de Bruijn graph. Bioinformatics **31**(10), 1674–1676 (2015)
19. Li, H.: Microbiome, metagenomics and high dimensional compositional data analysis. Annu. Rev. Stat. Appl. **2**, 73–94 (2015)
20. Lu, J., Breitwieser, F.P., Thielen, P., Salzberg, S.L.: Bracken: estimating species abundance in metagenomics data. PeerJ Comput. Sci. **3**, e104 (2017)
21. Ma, R., Cai, T.T., Li, H.: Optimal permutation recovery in permuted monotone matrix model. J. Am. Stat. Assoc. Accepted (2020)
22. Marcais, G., Kingsford, C.: A fast, lock-free approach for efficient parallel counting of occurrences of k-mers. Bioinformatics **27**(6), 764–770 (2011)
23. McLaren, M.R., Willis, A.D., Callahan, B.J.: Consistent and correctable bias in metagenomic sequencing experiments. eLife, article 46923 (2019)
24. Medema, M.H., Blin, K., Cimermancic, P., de Jager, V., Zakrzewski, P., Fischbach, M.A., Weber, T., Takano, E., Breitling, R.: antiSMASH: rapid identification, annotation and analysis of secondary metabolite biosynthesis gene clusters in bacterial and fungal genome sequences. Nucleic Acids Res. **39**(2), W339–W346 (2011)
25. Meleshko, D., Mohimani, H., Tracanna, V., et al.: BiosyntheticSPAdes: reconstructing biosynthetic gene clusters from assembly graphs. Genome Research **29**(8), 1352–1362 (2019)
26. Menegaux, R., Vert, J.P.: Embedding the de Bruijn graph, and applications to metagenomics. bioRxiv 2020.03.06.980979
27. Pasolli, E., Asnicar, F., Manara, S., et al.: Extensive unexplored human microbiome diversity revealed by over 150,000 genomes from metagenomes spanning age, geography, and lifestyle. Cell **176**(3), 649–662, e20 (2019)

28. Quince, C, Walker, A.W., Simpson, J.T., Lomanm N.J., Segata, N.: Shotgun metagenomics, from sampling to analysis. Nature Biotechnology **35**(9), 833–844 (2017)
29. Seah, B.K.B., Gruber-Vodicka. H.R.: gbtools: Interactive visualization of metagenome bins in R. Front. Microbiol. **6**, 1451 (2015)
30. Sunagawa, S., Mende, D.R., Zeller, G., Izquierdo-Carrasco, F., Berger, S.A., Kultima, J.R., Coelho, L.P., Arumugam, M., Tap, J., Nielsen, H.B., Rasmussen, S., Brunak, S., Pedersen, O., Guarner, F., de Vos, W.M., Wang, J., Li, J., Doré, J., Ehrlich, S.D., Stamatakis, A., Bork, P.: Metagenomic species profiling using universal phylogenetic marker genes. Nature Methods **10**, 1196–1199 (2013)
31. Truong, D.T., Franzosa, E.A., Tickle, T.L., Scholz, M., Weingart, G., Pasolli, E., Tett, A., Huttenhower, C., Segata, N.: MetaPhlAn2 for enhanced metagenomic taxonomic profiling. Nature Methods **12**, 902–903 (2015)
32. Wang, S., Cai, T.T., Li, H.: Hypothesis testing for phylogenetic composition: A minimum-cost flow perspective. Biometrika. Accepted (2020)
33. Wood, D.E., Salzberg, S.L.: Kraken: ultrafast metagenomic sequence classification using exact alignments. Genome Biology **15**, R46 (2014)
34. Ye, S.H., Siddle, K.J., Park, D.J., Sabeti, P.C.: Benchmarking metagenomics tools for taxonomic classification. Cell **178**(4), 779–794 (2019)
35. Zhu, Z., Ren, J., Michail, S., Sun, F.: MicroPro: using metagenomic unmapped reads to provide insights into human microbiota and disease associations. Genome Biology **20**(1), 154 (2019)