Ultrafast search of all deposited bacterial and viral genomic data

Phelim Bradley1,2, Henk C. den Bakker3, Eduardo P. C. Rocha4,5, Gil McVean6 and Zamin Iqbal1,2*

Exponentially increasing amounts of unprocessed bacterial and viral genomic sequence data are stored in the global archives. The ability to query these data for sequence search terms would facilitate both basic research and applications such as real-time genomic epidemiology and surveillance. However, this is not possible with current methods. To solve this problem, we combine knowledge of microbial population genomics with computational methods devised for web search to produce a searchable dataset structure named Bitsliced Genomic Signature Index (BIGSI). We indexed the entire global corpus of 447,833 bacterial and viral whole-genome sequence datasets using four orders of magnitude less storage than previous methods. We applied our BIGSI search function to rapidly find resistance genes MCR-1, MCR-2, and MCR-3, determine the host-range of 2,827 plasmids, and quantify antibiotic resistance in archived datasets. Our index can grow incrementally as new (unprocessed or assembled) sequence datasets are deposited and can scale to millions of datasets.

Whole-genome sequencing (WGS) of bacteria and viruses offers unparalleled resolution for a range of research questions including contact tracing, mapping the spread of drug resistance, identifying zoonoses and investigating the biology of infectious diseases. Genome sequence data are deposited in public archives such as the European Nucleotide Archive (ENA) and the Sequence Read Archive (SRA), which mirror their underlying data. The volume of archived microbial sequence data has doubled in size every 2 years, and is likely to grow at an increasing pace. However, it is currently impossible to search these archives for DNA sequences of interest, such as mutations or genes. This functionality would transform infectious disease management by enabling rapid identification of already-sequenced organisms that are highly similar to an outbreak strain, or by enabling studies of antibiotic resistance mutations, genes or plasmids.

At first sight, BLAST and its successor algorithms seem to enable these searches, performing alignment of query sequences against large databases. There are two reasons why these tools do not suffice. First, they would not scale to databases the size of the ENA or beyond. Second, they require assembled genomes as input; if applied to raw sequence data they would only find matches completely contained within a single read. However, assembly is fundamentally lossy when the input data contain multiple strains, and the highly heterogeneous historical data in the ENA would result in very variable assemblies, particularly of plasmids.

The first scalable method for searching raw sequences, named Sequence Bloom Tree (SBT), comprised a k-mer (fixed-length DNA word) index. SBT enabled detection of specific transcripts in RNA-seq data and allowed users to search archived datasets for the first time. SBT, methods based on SBTs in RNA-seq data and allowed users to search archived datasets for the first time. SBT, methods based on SBTs in RNA-seq data and allowed users to search archived datasets for the first time. However, bacterial and viral genomes encapsulate billions of years of evolution; even within one bacterial species, there can be enormous diversity due to horizontal transfer of DNA (Supplementary Fig. 1). We show this diversity renders previous methods unable to scale. We solve the problem with an alternative type of k-mer index, using a fixed-length binary 'signature' of each dataset. We call our data structure a Bitsliced Genomic Signature Index (BIGSI).

Searching the DNA archive is one example of a ‘document retrieval’ problem, a subject that has been intensively studied and successfully implemented at massive scale by internet search engines. Our ‘search terms’ are k-mers from single-nucleotide polymorphisms (SNPs) and alleles, and our ‘web pages’ are raw read datasets or assemblies. Bitsliced signatures were once used for text search, but were largely abandoned after Zobel et al. showed in 1998 that an alternative method (inverted indexes) performed better for natural language. One notable exception in 2017 was the Microsoft Bing search engine, which revived its use. For our use case, where each new microbial dataset brings new variation, bitsliced signatures provide much better scaling than inverted indexes. Web and (microbial) DNA search have different dimensionality, as the language of microbial genomes is vastly more complex than English. Our ENA/SRA dataset was only 106 documents but contained 1018 unique terms (here, k-mers), and this would continue to increase with more data, whereas Google indexes 1012 documents containing (we estimate) 1018 terms, with a much more slowly growing lexicon.

Here, we use BIGSI to index the entire bacterial and viral WGS content in the ENA as of December 2016 (Fig. 1 for context) and apply it to solve basic microbial surveillance problems. We also make a demonstration search publicly available at http://bigsi.io.

Results

We developed a data structure suitable for storing microbial genomic data, called BIGSI. We use the generic term ‘dataset’ to refer to either an assembled genome or unassembled sequence read files from clonal or non-clonal samples. BIGSI combines a k-mer index with...
Fig. 1 | Sequence matching methods. a, Mapping of sequence reads to a reference genome from the same species, assuming relatively low divergence; requirement to map millions of reads in acceptable time and return an alignment and mapping score. Common tools: bwa11 and bowtie12. b, BLAST7 compares a query string with a database of reference genomes (in the figure we show RefSeq genomes in a dotted box) covering a massive phylogenetic range. BLAST takes k-mers from the query, and for each k-mer it creates a ‘neighborhood’ of k-mers within a fixed edit distance (edits are shown in red, (ii)), and searches for these in the reference genome database. Alignment is only done by extending from these hits. Blast can be applied to nucleotide and protein searches and can find close and remote homology matches. c, MASH stores a tiny fingerprint of each reference in the database (in this case RefSeq). Querying with an assembly, the fingerprint of the assembly is compared with that of RefSeq to find the closest reference. d, Sequence Bloom Tree13 was the first scalable method to search through raw unassembled readsets (unassembled readsets are shown as ‘piles’ of reads (short lines), all in the same color to signify same species), by indexing the k-mers in the data and then compressing the index. Designed for human data, SBT can be applied to generate the binary encoding; Fig. 2 and Methods), creating a probabilistic colored de Bruijn graph17,26. A comparison of functionality supported by BIGSI compared with other tools is shown in Supplementary Table 1. e, BIGSI can search the complete set of raw sequence data for bacteria and viruses. RefSeq is shown in a dotted box amongst unassembled readsets; different colors to signify the massive range of species and phyla. The different input data for SBT and BIGSI mean that these methods have different speed and compression trade-offs.

constraints on sequence queries, described below. A Bloom filter is a data structure25 that stores data (here, k-mers) in a bit-vector (array of zeroes and ones) and answers set-membership queries (“is this k-mer contained in the set?”) probabilistically. The false-negative rate is zero, and the false-positive rate is controlled by two parameters (size of bit-vector and the number of (hash) functions used to generate the binary encoding; Fig. 2 and Methods), creating a trade-off between false-positive rate and compression. We describe how we set the Bloom filter parameters below. BIGSI encodes data as a matrix in which each column is a Bloom filter of the k-mers in a dataset. Querying for presence of a k-mer involves applying the Bloom filter hash functions to this k-mer, which each return an integer, and taking the rows of the matrix (bit slices) indexed by these integers; datasets (columns) containing the k-mer have a 1 in all of those rows. Fast (O(1)) access to the rows is achieved using a hash mapping from row index to the corresponding bit-vector (Fig. 2 shows how data are processed and stored, and how queries are made; details are in Methods). Incorporating a new dataset simply requires the addition of a new column, without needing to rebuild the index. However, because appending a new column to a BIGSI requires modifying every key in the index, this is an expensive operation. Our solution is to batch new inserts, building a new index per batch, and then merging indexes. Our implementation supports both disk-based and in-memory stores, but all measurements in this report are from the disk-based store. Although we use BIGSI as a k-mer index, it can be viewed as a probabilistic colored de Bruijn graph17,26. A comparison of functionality supported by BIGSI compared with other tools is shown in Supplementary Table 1.

To search for a sequence, we query the index for all the k-mers within it. Exact matching requires all k-mers be present (threshold $T = 100\%$) and can be implemented as a fast AND operation on the bit-vector rows. Inexact matching, primarily used for long alleles, requires the presence of some proportion ($T < 100\%$) of k-mers be present and is slower, as the bits are unpacked and summed.
The relationship between the proportion of k-mers present ('k-mer identity') and the more traditional sequence identity used by BLAST is non-linear but monotonic (Methods and Supplementary Figs. 2 and 3). In creating BIGSI, we use a relatively high per-k-mer error rate of ~0.3 to enable a higher Bloom filter compression rate, but require a longer minimum query length (61) (Methods). For example, to search for and genotyping a SNP, we create a sequence probe for each allele, with k-1 bases on either side, and demand multiple k-mers from the query be present, reducing the false-positive rate for SNP allele detection exponentially. The theoretical false discovery rate for a SNP allele from a probe of length (2k-1) with our Bloom filter parameters is $10^{-22}$ per column, which is well below the expected error rate from the underlying sequence data ($10^{-22}$).

We measured query speed by first building a BIGSI of 3,480 datasets of Mycobacterium tuberculosis, obtained from ref. 27, and genotyping 68,269 SNPs. Searching all of these datasets for these SNPs took less than 90 minutes on a single CPU core, which translates to a genotyping rate of more than 46,000 genotypes per second. We validated SNP genotyping accuracy using a subset of 100 of the M. tuberculosis datasets for which we had high-quality SNP calls using samtools30 (Methods). The concordance between methods was 99.997%, with a total of only 286/682,690 discrepancies. We measured accuracy of longer allele detection by searching (with $T=70\%$ match) for a catalog of Escherichia coli Multi Locus Sequence Type (MLST) alleles and choosing the best-scored allele for each gene. We then compared calls on a set of 954 datasets with the MLST allele calls from a high-quality caller: SRST2 (ref. 29). Where both methods made a call (6,483/6,678 alleles), there was 99.9% agreement; otherwise, SRST2 failed ($n=167$) or BIGSI failed to find an allele version above $T=70\%$ ($n=28$).

**Benchmarking BIGSI.** We benchmarked the empirical scaling properties of BIGSI against the Sequence Bloom Tree (SBT)13 and the Split Sequence Bloom Tree11 (SSBT) using a dataset of 10,000 random microbial sequence datasets from the ENA. We compared build and query times and peak storage requirements on 21 subsets ranging from 100 to 10,000 samples. For each subset, we built a BIGSI, an SBT and an SSBT using two different parameter settings: one optimized for speed (recommended by the authors in Lemma 1, and consequent text, of Solomon et al.13 and the SBT/SSBT user manual) (SBT-fast/SSBT-fast) and another optimized for compression (SBT-small/SSBT-small) (Methods). For database sizes of more than 2,000 samples, peak storage requirements of SBT-fast and SSBT-fast exceeded our available disk space (1 Tb; by comparison, the total storage required for the input data was 403 Gb; Methods).

We queried the resulting indexes for 2,157 antimicrobial resistance genes with a mean length of 937 bp and total query length of 2,021,655 bp. SBT-fast, SSBT-fast, SSBT-small and BIGSI returned near-identical hits from the exact match search, with only one difference across all queries (measured on $N=1,000$ database size). With the inexact search, concordance was >99%, with the differences likely due to differences between methods in the construction of the underlying Bloom filters. Inexact ($T=40\%$) query times versus peak storage requirements for the various methods are shown in Fig. 3; BIGSI maintains good query performance in small space for all input data sizes, whereas both SBT and SSBT trade storage requirements for performance (because they need to build an uncompressed tree of Bloom filters before compressing it; we quantify performance below). SSBT-fast and SBT-fast have query times comparable to or better than that of BIGSI but require
orders of magnitude more storage to build the uncompressed tree. SBT-small and SSBT-small have almost equivalent storage as BIGSI, but slower query performance, due to saturation of internal Bloom filters within the SBT, a result of the total number of unique k-mers being significantly larger than the number of unique k-mers in any individual dataset. With 2,000 samples in the index, the SBT-fast query was only slightly slower than BIGSI’s (292s versus 274s) but required 132x (878GB/6.6 GB) more storage during construction and 9x (57.5 GB/6.6 GB) more storage after compression (details in Methods). Similar results were found for exact queries (T = 100%, Supplementary Fig. 4); exact-match searches with BIGSI were up to 2–3x faster than inexact queries.

To ensure our SBT comparison was fair, we also benchmarked BIGSI and SBT on the same human RNA-seq dataset used in refs. 13,15,30, using the prebuilt SBT index provided by the authors. We measured the query time of 1,000 RNA transcripts randomly selected from the 214,294 known transcripts (reported in ref. 13) and compressed index size. BIGSI has faster query time than SBT in smaller space for these datasets (360 s and 144 Gb for BIGSI, and 2,221 s and 200 Gb for SBT (T = 70%); details in Methods and full results, including for exact search, in Supplementary Table 2).

Finally, we simulated the scaling of storage requirements required to build SBT-fast and BIGSI for datasets of up to 1 million genomes of two types: genomes with high proportions of k-mer sharing (e.g., human) and genomes with lower proportions of k-mer sharing (e.g., bacteria) (Supplementary Fig. 5). BIGSI scales linearly with the number of datasets, performing identically in both cases. For the low-k-mer-sharing simulation, an unsaturated (i.e., fast) SBT/SSBT would require four orders of magnitude more storage than BIGSI to construct (tens of Pb rather than 3 Tb).

An index of all bacterial and viral sequence in the ENA. We downloaded the entire set of 469,654 bacterial, viral and parasitic WGS datasets in the ENA as of December 2016. After a step to exclude eukaryotic genomes on the basis of size (Methods), we were left with 447,833 datasets, from which we created a BIGSI index. This index, named the ‘all-microbial index’, required 1.5 TB of storage, took <1% of the original data size (170 TB), and contained more than 60 billion unique k-mers. Data download took 6 weeks, and we built Bloom filters on the fly. Combining the complete set of Bloom filters took approximately 2 d (Methods). We estimate that the intermediate storage required to build an SBT-fast or SSBT-fast of the same data with recommended Bloom filter size equal to the number of unique k-mers in the collection would be >6.7 PB.

We used our all-microbial index for several analyses. First, we estimated the species and abundances present in each dataset within the all-microbial index using the Bayesian abundance estimator Bracken31, which parses output from the read-classifier Kraken 32 (Methods). Using this method, we found that more than 90% of the input datasets were from only 20 genera, and 65% of the input data sets were from the five most common bacterial genera (Salmonella, Streptococcus, Staphylococcus, Escherichia and Mycobacterium; counts for the most prevalent bacterial genera in the all-microbial index are shown in Supplementary Fig. 6).

Ultrafast gene search using the all-microbial index. We searched for exact matches to the mobilized colistin resistance genes MCR-1, MCR-2 and MCR-3 (refs. 31–33) in the all-microbial index. Searching for all 3 genes in ten times more genomes than has been previously reported took just 1.73 s in total. We did not detect MCR-2. MCR-1 was present in 169 datasets from 3 species (Escherichia coli, Salmonella enterica, Enterobacter aerogenes) and MCR-3 was present in 34 datasets from 3 species (E. coli, S. enterica, Klebsiella pneumoniae) (Supplementary Data 1). Our all microbial index therefore enables almost instant detection of drug resistance genes, or indeed any other gene studied, across the global corpus.

Host-range of plasmids in the all-microbial index. We downloaded a set of 2,827 plasmids from the ENA (Methods and Supplementary Data 2) and ran an inexact (T = 40%) search for them in the all-microbial index. We restricted analysis to hits with T > 90%. The total length of query sequences was 227 Mbp, and the query took 2,120 CPU-hours (11 d) on a single server using eight cores and 1.5 GB RAM per process. The search returned 665,619 hits with 121,758 unique accessions across 258 genera. Because contamination could confound observations of a plasmid in a genus, we excluded all datasets containing evidence of more than one genus at abundance above 0.1%. Only 41% (184,652) of datasets and 62% of search hits passed this filter.

In the filtered output, we identified plasmids shared by closely related genera, for example, Escherichia, Shigella and Salmonella; or Enterococcus, Streptococcus and Staphylococcus. We found 37 plasmids present in at least five datasets from at least two genera (Fig. 4 and Supplementary Data 2 and 3) and five plasmids that were present in multiple orders and families. The plasmid pETHIS-1 (GenBank identifier: AF012911) was found in five phyla, ten taxonomic classes and 17 genera. This plasmid is used as an expression vector, and its identification in so many species serves as a positive control, confirming that BIGSI can detect common plasmids. Of more biological interest, the Tn916 conjugative transposon encoding tetracycline resistance that was first found in Enterococcus faecium and is known to have broad host range33 (GenBank identifier: U09422) was found in Streptococcus (n = 3,951), Staphylococcus (n = 1,212), Enterococcus (n = 43), Clostridioides (n = 29), Listeria (n = 19) and Erysipelothrix (n = 11).

Sampling biases in the ENA preclude inference about plasmid distribution, but they do allow us to ask whether plasmids bearing antibiotic resistance genes are more widely distributed than plasmids that do not have antibiotic resistance genes. We defined ‘phylogenetic spread’ of a plasmid as the median of the pairwise distances along a large-subunit rRNA tree (incorporating branch lengths) between all pairs of genera in which the plasmid is detected. We tested whether plasmids harboring at least three antibiotic resistance genes are more widely distributed across this phylogeny than plasmids with no antibiotic resistance genes by comparing the 95% quantile of these two distributions and found that they are not significantly different (P = 0.0024, 1 million replicates) (Supplementary Fig. 7). Because the sampling of the ENA data is biased toward the Enterobacteriaceae, we would restrict this conclusion to that family.

The distribution of different versions of the machinery for conjugation of DNA between bacteria has previously been analyzed in 1,124 genomes38 using sensitive, but slow, protein profile searches for relaxase (MOB) and type 4 secretion system (T4SS) genes. We undertook a similar analysis of the whole ENA, by using an exact match search of the all-microbial index for nucleotide alleles corresponding to previously identified MOB and T4SS “types” (alleles defined in amino acid space). We applied the same contamination filter mentioned above (restricting to 184,652 datasets) and required exact matches to at least one MOB and one T4SS to be present to be considered a putative conjugative system. 36,030 datasets met these criteria, giving an estimate of 19.5% of bacteria in the ENA containing a conjugative system, which is consistent with the previous estimate of 18% in Guglielmini et al.39. This proportion varied by phylum (minimum 0.5% in Spirochaetes and maximum 31.7% in Firmicutes, full data in Supplementary Table 3, and MOB type distribution in Supplementary Fig. 8). These data could be mined further to explore the potential spread of antibiotic resistance genes. For example, by analyzing datasets shown to contain MOB₉ we observed genetic flux between Staphylococcus and Streptococcus, but not between either of them and Salmonella. We also found MOB₉ in Salmonella and Streptococcus but not Staphylococcus, indicating a different
probability of cross-genus (and cross-phylum) transfer by conjugation (Supplementary Data 4).

Antibiotic resistance genes in the ENA. To understand whether antibiotic resistance gene prevalence in the ENA has changed over time, we downloaded the set of 2,157 sequences associated with antibiotic resistance from the Comprehensive Antibiotic Resistance Database (CARD) (v1.1.7) and searched for this set of genes in the all-microbial index with thresholds of 100% and 70% (full results for both searches are in Supplementary Data 5). An exact search for a single gene took an average of 1.1 s and returned an average of 438 hits, whereas the corresponding figures for an inexact search (T = 70%) were 34.4 s and 5,320 hits. We show the prevalence over time in 3 genera and in total in Fig. 6. Focusing only on *Staphylococcus*, we found that the proportion of datasets containing *mecA* gene (methicillin resistance) fell from 70% in 2013 to 40% in 2016 and that prevalence of *tet* and *aac* genes also decreased (Fig. 6b). For *Klebsiella*, however, the prevalence of all resistance genes that we screened for in the all-microbial dataset increased over time (Fig. 6c).

In *M. tuberculosis*, antibiotic resistance mainly arises through amino acid mutations in specific genes. We genotyped the all-microbial index, which contains 30,226 *M. tuberculosis* genome datasets, for the 206 resistance mutations enumerated by Walker et al. This exercise took only 103 minutes on a single core, which is approximately 10,000× faster than typing each dataset individually with the fast resistance prediction software Mykrobe predictor.
Fig. 6 | Antibiotic resistance gene prevalence in ENA over time. a, Counts of samples in the all-microbial index containing a range of antibiotic resistance genes; each gene was treated independently, so a single dataset containing both CTX-M and OXA, for example, is counted twice. b, Year-by-year frequency (defined by date of public availability) in Staphylococcus (dominated by S. aureus) of meca, and all tet and aac genes, which encode resistance to methicillin, tetracycline and aminoglycosides, respectively. c, Year-by-year frequency in Klebsiella of various antibiotic resistance genes; increase in prevalence since 2014 may be due to increased extended spectrum β-lactamase surveillance and sampling of KPC-resistant Klebsiella globally. d, Year-by-year breakdown of M. tuberculosis datasets, classified by genotypes as resistant (R), pan-susceptible (S), multiple-drug resistant (MDR) or extensively drug resistant (XDR), as follows: all datasets were genotyped for variants from the resistance catalog from ref. 27, then classified as resistant or susceptible to 12 antibiotics based on their genotype. Datasets were classed as MDR (multi-drug resistant) if resistant to isoniazid and rifampicin; as XDR (extensively drug-resistant) if MDR and also resistant to a fluoroquinolone and any of capreomycin, kanamycin and amikacin; as resistant if resistant to any antibiotic but not MDR or XDR; and susceptible otherwise. Error bars around the estimated frequency (mean) in b–d show the 95% confidence interval calculated using the Wilson binomial confidence test.
Our results indicate an increase in prevalence of MDR *M. tuberculosis* in the ENA since 2011 (Fig. 6d). It should be noted that one cannot infer anything about global prevalence from this, as the contents of the ENA are subject to sampling biases (that is, are dependent on specific research studies rather than containing an unbiased sampling of isolates from tuberculosis patients). Indeed, the latest WHO estimates for 2016\(^2\) put MDR prevalence at 6.6% as compared with the 18.9% of datasets deposited in the ENA/SRA in 2016 that we detected (Fig. 6d).

**Discussion**

Hundreds of thousands of bacterial and viral samples are sequenced and shared every year, and this number is growing exponentially. A scalable online sequence search facility that could access all deposited sequence data would unlock the archives for fundamental research, clinical microbiology and public health, in much the same way that web search allows people to narrow down to webpages of interest. In particular, there is a pressing need for a global infrastructure for surveillance and management of infectious diseases\(^6\). Until now, this has not been possible. We have developed a data structure (BIGSI) to meet this need and have indexed the entire bacterial and viral WGS content of the global DNA archive. BIGSI allows tracking of several key entities: genes, SNPs, plasmids, MLST types, or clusters defined by SNPs. Importantly, new datasets can be readily and quickly added to our index, allowing it to grow as new datasets are sequenced. Our indexing method works for both raw data and assembled genomes, so it is poised for a future in which finished reference genomes are routine.

BIGSI was designed with SNP or indel genotyping and allele search queries in mind. It allows a user to identify datasets worthy of in-depth study, as well as enabling global monitoring of specific alleles. Nevertheless, BIGSI has limitations, which it shares with previous \(k\)-mer tools such as the SBT (main limitations discussed here; see also Methods). It is a \(k\)-mer index and is therefore as lossy as all de Bruijn methods. Reconstruction of stored genomes is impossible and repeat regions cannot be resolved. Also BIGSI does not store coverage information, which rules out queries in which copy number is important. One example of such a query is detecting azithromycin resistance in *Neisseria gonorrhoeae*, in which the level of resistance is mediated by the number of RNA genes containing a particular SNP\(^6\). BIGSI only supports nucleotide \(k\)-mers, although we would note that an extension to amino acid search would be straightforward because Bloom filters are agnostic to what they are storing. Finally, BIGSI is optimized for diverse datasets in which the combined unique \(k\)-mer count is much higher than that of any individual sample, e.g., the entire microbial ENA/SRA. Where samples are less diverse (e.g., human), we expect SBT or Mantis to be a better choice, as they take advantage of compression from sample similarity.

In the future, we envisage sequencing data volumes continuing to grow, and the user base that wants to access and exploit those data will increase to include clinical and public health practitioners. There will be a pressing need for quick and accurate sequence searches across the global corpus of microbial sequences. MASH combined with both nucleotide and protein BIGSI could be a powerful combined approach to achieve this goal. We are now investigating implementing our BIGSI as a live service at the EMBL-EBI, to be updated as data are added to the ENA. These and complementary approaches will put shared DNA resources at everyone’s fingertips.

**Online content**

Any methods, additional references, Nature Research reporting summaries, source data, statements of data availability and associated accession codes are available at [https://doi.org/10.1038/s41587-018-0010-1](https://doi.org/10.1038/s41587-018-0010-1).

**References**

1. Bradley, P. et al. Rapid antibiotic-resistance predictions from genome sequence data for *Staphylococcus aureus* and *Mycobacterium tuberculosis*. *Nat. Commun.* 6, 10063 (2015).
2. Brown, A. C. et al. Rapid whole-genome sequencing of *Mycobacterium tuberculosis* isolates directly from clinical samples. *J. Clin. Microbiol.* 53, 2230–2237 (2015).
3. Quick, J. et al. Real-time, portable genome sequencing for Ebola surveillance. *Nature* 530, 228–232 (2016).
4. Schmidt, K. et al. Succinct colored de Bruijn graphs. *Nat. Biotechnol.* 34, 2556–2565 (2017).
5. Altschul, S. F., Gish, W., Miller, W., Myers, E. W. & Lipman, D. J. Basic local alignment search tool. *J. Mol. Biol.* 215, 403–410 (1990).
6. Shea, J. et al. Comprehensive whole-genome sequencing and reporting of drug resistance profiles on clinical cases of *Mycobacterium tuberculosis* in New York state. *J. Clin. Microbiol.* 55, 1871–1882 (2017).
7. Inouye, M. et al. SRST2: Rapid genomic surveillance for public health and hospital microbiology labs. *Genome Med.* 6, 90 (2014).
8. Kent, W. J. BLAT—the BLAST-like alignment tool. *Genome Res.* 12, 656–666 (2002).
9. Zhang, Z. et al. A greedy algorithm for aligning DNA sequences. *J. Comput. Biol.* 7, 203–214 (2000).
10. Walker, T. M. et al. Whole-genome sequencing for prediction of *Staphylococcus aureus* drug resistance. *Cell Syst.* 7, 201–207 e204 (2018).
11. Solomon, B. & Kingsford, C. Fast search of thousands of short-read sequencing experiments. *Nat. Biotechnol.* 34, 300–302 (2016).
12. Almodaresi, F., Pandey, P. & Patro, R. Inverted files versus signature text indexing. *Proc. 11th Algorithms in Bioinformatics* (Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2017).
13. Li, H. et al. The sequence alignment/map format and SAMtools. *Bioinformatics* 25, 2078–2079 (2009).
32. Wood, D. E. & Salzberg, S. L. Kraken: ultrafast metagenomic sequence classification using exact alignments. Genome Biol. 15, R46 (2014).
33. Hu, Y., Liu, F., Lin, I. Y., Gao, G. F. & Zhai, B. Dissemination of the mcr-1 colistin resistance gene. Lancet Infect. Dis. 16, 146–147 (2016).
34. Lu, X. et al. MCR-1.6, a new MCR variant carried by an incp plasmid in a colistin-resistant Salmonella enterica serovar typhimurium isolate from a healthy individual. Antimicrob. Agents Chemother. 61, e02632–16 (2017).
35. Matamoros, S. et al. Global phylogenetic analysis of Escherichia coli and plasmids carrying the mcr-1 gene indicates bacterial diversity but plasmid restriction. Sci. Rep. 7, 15364 (2017).
36. Xavier, B. B. et al. Identification of a novel plasmid-mediated colistin-resistance gene, mcr-2, in Escherichia coli, Belgium, June 2016. Euro. Surveill. https://doi.org/10.2807/1560-7917.ES.2016.21.27.30280 (2016).
37. Xin, W. et al. Novel plasmid-mediated colistin resistance gene mcr-3 in Escherichia coli. mBio https://doi.org/10.1128/mBio.00543-17 (2017).
38. Ciric, L. J. A., Elvira de Vries, L., Agero, Y., Mullany, P., Roberts, A. P. In: Madame Curie Bioscience Database [Internet] (Landes Bioscience, Austin, TX, 2013).
39. Guglielmini, J., Quintais, L., García-Fernández, M. P., de la Cruz, F. & Rocha, E. P. The repertoire of ICE in prokaryotes underscores the unity, diversity, and ubiquity of conjugation. PLoS Genet. 7, e1002222 (2011).
40. Jia, B. et al. CARD 2017: expansion and model-centric curation of the comprehensive antibiotic resistance database. Nucleic Acids Res. 45, D566–D573 (2017).
41. Eldholm, V. & Balloux, F. Antimicrobial resistance in Mycobacterium tuberculosis: the odd one out. Trends Microbiol. 24, 637–648 (2016).
42. World Health Organization. Global Tuberculosis Report 2017. https://www.who.int/tb/publications/global_report/gtbr2017_main_text.pdf (2017).
43. Gardy, J. L. & Loman, N. J. Towards a genomics-informed, real-time, global pathogen surveillance system. Nat. Rev. Genet. 19, 9–20 (2018).
44. Schatz, M. C. & Phillippy, A. M. The rise of a digital immune system. Gigascience 1, 4 (2012).
45. Eyre, D. W. et al. WGS to predict antibiotic MICs for Neisseria gonorrhoeae. J. Antimicrob. Chemother. 72, 1937–1947 (2017).
46. Li, H. & Durbin, R. Fast and accurate short read alignment with Burrows-Wheeler transform. Bioinformatics 25, 1754–1760 (2009).
47. Langmead, B., Trapnell, C., Pop, M. & Salzberg, S. L. Ultrafast and memory-efficient alignment of short DNA sequences to the human genome. Genome Biol. 10, R25 (2009).

Acknowledgements
We would like to thank, for critical reading and helpful suggestions: G. Blackwell, R. B. Franson, M. Hunt, J. Kelleher, J. Thornton, R. Patro, K. Malone, J. Marioni, A. Page, S. Gog, T. Bellman, F. Gauger. For enormous assistance with data download from EBI: R. Enouf, G. Cochrane. For hosting our BIGSI demonstration: CLIMB. We acknowledge funding from Wellcome Trust Core Award Grant Number 203141/Z/16/Z. Z.I. was funded by a Wellcome Trust/Royal Society Sir Henry Dale Fellowship, grant number 102541/A/13/Z. G.M. was funded by Wellcome Trust grant 100956/Z/13/Z. E.P.C.R. was funded by the ANR MAGISBAC grant number ANR-14-CE10-0007-02. P.B. was funded by Wellcome Trust Studentship H5RZCO00.

Author contributions
Z.I., G.M. designed and oversaw the study; P.B. invented the method, developed software and performed analyses; H.C.d.B. performed analyses for the plasmid study; E.P.C.R. co-designed and analyzed the conjugative system and plasmid analysis; Z.I. wrote the paper; all authors gave detailed feedback on the paper.

Competing interests
G.M. is a cofounder of, holder of shares in, and is a consultant to Genomics PLC.

Additional information
Supplementary information is available for this paper at https://doi.org/10.1038/s41587-018-0010-1.
Reprints and permissions information is available at www.nature.com/reprints.
Correspondence and requests for materials should be addressed to Z.I.

Publisher’s note: Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.
© The Author(s), under exclusive licence to Springer Nature America, Inc. 2019
Methods

**BIGSI construction and querying.** BIGSI indexes a set of \( N \) (number of datasets) Bloom filters by position in the Bloom filter. Each Bloom filter must be constructed with the same parameters \((m, \eta)\), where \( m \) is the Bloom filter’s length in bits and \( \eta \) is the number of hash functions applied to each \( k \)-mer. The same hash functions must also be used to construct each Bloom filter. To construct a BIGSI, the \( N \) Bloom filters are column-wise concatenated into a matrix. The row index and row bit-vectors are then inserted into a hash table or key-value store as key-value pairs so that row lookups can be done in \( O(1) \) time. This set of key-value pairs can be stored on disk, in memory or distributed across several machines and is indexed via a hash index (a b-tree would be an alternative option). To insert a new Bloom filter, we append it as a column to the existing bitmap. To query the BIGSI for a \( k \)-mer, we hash the \( k \)-mer \( q \) times, look up the resulting keys in the key-value store, and take the bit-wise AND of the resulting bit-vectors (Fig. 1).

**Parameter choices.** The choice of BIGSI parameters \((m, \eta)\) depends on the maximum number of \( k \)-mers expected in any dataset \((K_{\text{max}})\), the number of datasets \((N)\) expected, the smallest number of unique \( k \)-mers in a query sequence \((L_{\text{min}})\) to be supported, the \( k \)-mer size \( (k) \) and the maximum number of acceptable false discoveries per query \((q_{\text{max}})\). The expected number of false discoveries \((V)\) for any query can be calculated as \( q = \mathbb{E}[V] = Np^L \) where \( p \) is the false positive rate of the Bloom filter and \( L \) is the number of unique \( k \)-mers in the query, assuming independence of the \( k \)-mers and Bloom filters. Parameters \( m \) and \( \eta \) determine false positive rate for a Bloom filter with \( K_{\text{max}} \)-mers if there are fewer elements, then the false positive rate will be lower. We assume below that all Bloom filters have the maximum number of unique \( k \)-mers inserted to give an upper bound on error rate. However, the independence assumption mentioned above is, strictly speaking, false for real data archives (such as the ENA/SRA), which have biased distribution of datasets across the phylogeny. If there is enrichment of datasets between datasets. Even for this small benchmark dataset, constructing unsaturated datasets (\( \geq 1 \) billion) takes more than 755,176;2 \( \times \) require \( ~11 \) TB of storage to build SBT-fast, 350 \( \times \) more space than is required by BIGSI.

For example, given \( N = 10^6; K_{\text{max}} = 10^4; L_{\text{min}} = 31 \) \( k \)-mers; \( k = 31; q_{\text{max}} = 10^{-6} \) the resulting expected number of false positives per \( k \)-mer lookup per Bloom filter \((p)\), would be:

\[
p = \left( \frac{q_{\text{max}}}{N} \right)^L = \frac{1}{10^6} = 0.2511...
\]

Solving the above equations gives:

\[
m = 28, 755, 176; \eta = 2
\]

Finally, we note that any path through the de Bruijn graph that was not in the original genome will be classified as present by BIGSI (as all the \( k \)-mers are present), creating a false positive that is not considered in the above modeling. This can only happen if a query includes \( k \)-mers repeated in a genome.

**BIGSI parameters for the all-microbial index.** We assume initially that a bacterial dataset contains at most 10 million \( k \)-mers since bacterial genomes are generally under 6 Mb in length, leaving 4 million \( k \)-mers available for some sequencing errors that escape de-noising and for plasmid variation. Unless otherwise specified we use BIGSI parameters \( m = 25,000,000; \eta = 3 \) for all analyses. For these parameters, the upper bound on number of false discoveries, \( q_{\text{max}} \), for an SNP allele from a probe (flanks plus allele) of length \((L_{\text{min}} = 2k − 1 = 61)\) is \(10^{-55}(\text{if } K_{\text{max}} = 10^6, N = 10^6)\).

**k-mer identity and scoring.** If, for example, the \( k \)-mer size is 31, each SNP difference between the query and the nearest sequence in the index causes a window of 31 absent \( k \)-mers. Therefore, \( k \)-mer identity drops more rapidly than sequence identity; e.g., for \( k = 31 \), matches with sequence identity above 80% can be detected (Supplementary Fig. 2). Therefore, BIGSI most naturally enables exact or close match searches, or situations where combinatorial searching is feasible, such as querying all sequences 2 SNPs different from a given allele or all single amino acid changes in a gene. Datasets containing matches for a more divergent allele can be sought by searching for sub-sequences (seeds).

Although BIGSI does not carry out an alignment, an approximation to a megabLAST alignment score can be inferred from the presence/absence pattern of \( k \)-mers in the query. To approximate a megabLAST score, we take the presence/absence vector for a query of \( L \)-mers. From the estimate, we approximate the number of mismatches of the query from the hit by counting the number of zeroes in contiguous runs of length greater than 1 and dividing by the \( k \)-mer size. From these estimated mismatches and matches we calculate a score for an untagged alignment, with \( p \)-values calculated using the same scheme as BLAST. By default the costs are −2 for a mismatch and +1 for a matched position. This approximation deteriorates as \( k \)-mer identity drops (see Supplementary Fig. 2). Nevertheless, we show the strong correlation \((r = 0.998)\) between MegaBLAST score and BIGSI score for 100 \( E. \ coli \) AMR genes using a BIGSI of RefSeq-bacteria (release 81) in Supplementary Fig. 3.

Benchmarking query time and storage requirements of BIGSI, SBT, and SSBT. We randomly chose 10,000 microbial cleaned de Bruijn graphs from the all-microbial-index archives, and we then further randomly sub-sampled these into collections of 10, 100, 200, 300, 400, 500, 600, 700, 800, 900, 1,000, 2,000, 3,000, 4,000, 5,000, 6,000, 7,000, 8,000, 9,000, 10,000 datasets. A BIGSI of each set of datasets was built with parameters \((m = 2.5 \times 10^7; \eta = 3)\). A SBT and SSBT were built for each set of datasets with \( \eta = 1 \) and Bloom filter size \((m)\) equal to the count of the total number of \( k \)-mers in the collections of graphs, as recommended in the text following Lemma 1 of Solomon et al., called “SBT-fast” and “SSBT-fast” respectively. Redis hyperloglog was used to count the unique \( k \)-mers in the set of cleaned graphs for each incremented dataset. A SBT and SSBT were then further randomly sub-sampled these all-microbial-index accessions, and we then further randomly sub-sampled these into collections of 10, 100, 200, 300, 400, 500, 600, 700, 800, 900, 1,000, 2,000, 3,000, 4,000, 5,000, 6,000, 7,000, 8,000, 9,000, 10,000 datasets. A BIGSI of each set of datasets was built with parameters \((m = 2.5 \times 10^7; \eta = 3)\) (the same Bloom filter parameters as BIGSI), called “SBT-small” and “SSBT-small.” Construction and query time analyses were run on an Amazon Web Service i3.8xlarge instance with 32vCPUs, 224 GiB of memory and 4 × 1.9 TB non-volatile SSD-backed instance storage. SBT-fast construction exceeded 1 TB for 3,000 datasets, the practical limit we set, and as a result SBT-fast was not built for increments above 2,000 datasets.

The search time comparison was run with ‘bt query’ and ‘bigsi search –seqfile $f ’, using \( k \)-mer thresholds 40% and 100%. A full table of results can be found in Supplementary Data 6.

As mentioned above, SBT-fast and SSBT-fast construction failed for databases >2,000 due to excessive disk requirements, but we were able to calculate a lower bound for the peak disk-use for database sizes >2,000; query times were extrapolated linearly (these points are shown as triangles on Fig. 3; details of lower bound below).

There were approximately \(4.4 \times 10^7\) unique \( k \)-mers in the union of all 10,000 datasets, almost 1,000 times more than in a typical individual dataset. It would require >11 TB of storage to build SBT-fast, 350x more space than is required by BIGSI. By contrast, querying SBT-small took 65x longer and SSBT-small 448x longer than a BIGSI of the same database size \((N = 10,000)\). SSBT-small was nearly 7x slower than SBT-small, perhaps due to the lower proportion of shared \( k \)-mers between datasets. Even for this small benchmark dataset, constructing unsaturated (i.e., fast) SBTs and SSBTs for larger numbers of datasets is prohibitive in terms of storage requirements; we address scaling further below. Query times for SBT/SSBT slow were unacceptable for this benchmark, and so we excluded them as candidates for storing the ENA/SRA, which is 50x bigger than this benchmark dataset.

Simulation of storage requirements for a BIGSI for \( N \) datasets is given by:

\[
\text{BIGSIbytes}[\text{bytes}] = \frac{mN}{8}
\]
Although it is possible to append to an SBT incrementally, as new microbial datasets will keep adding new k-mers, this will lead to saturation of the root-level Bloom filter in the SBT and a collapse in query performance. This can be avoided by reconstructing the SBT, ensuring the Bloom filters are large enough to support the full set of k-mers. This was borne out by our benchmarking. In simulating scaling to 1 million genomes, we therefore focused on SBT-fast rather than SBT-small. As the best case for a binary tree with N leaves is 2N – 1 nodes, we estimate:

$$SBT_{\text{target}} \text{[bytes]} = \frac{(2N - 1)N_k}{8}$$

where N_k is the total number of k-mers in the combined set of datasets and also equal to the size of the Bloom filter required. This is a lower bound for the peak storage use. See Supplementary Data 6 for the close correspondence between this theoretical estimate and the empirical measurement in our benchmark datasets.

As a result, we can calculate lower bounds for the peak disk usage for SBT/SSBT; when the explosion of disk usage made construction of indexes impractical for the benchmark datasets with size > 2,000, we used this lower bound to plot (suitably labelled) extrapolated data points.

**Benchmarking against Mantis.** We attempted to benchmark against Mantis but despite assistance from the authors we were unable to resolve a number of issues: large numbers of false positive hits on some data and a bug causing segmentation faults when trying to build more than 3,000 datasets. We reluctantly excluded Mantis from our benchmarking due to limitations of time. As currently implemented, the Mantis data structure does not support incremental insertion, as it needs up front the full set of k-mers and, for each k-mer, the list of datasets containing it ("color class," stored as a bit-vector). There is currently an intermediate stage where the uncompressed color-class matrix is held in RAM, scaling quadratically in number of datasets.

**Indexing of ENA snapshot.** De Bruijn graphs (k = 31) were constructed and cleaned from the downloaded RNA-seq fastq files listed in ref. 13. A BIGSI was built with Bloom filter parameters K_max = 4,000,000,000; m = 1 by chunking into 1,000 batches, building in parallel and then combining into a final index. The SBT and the 214,294 transcripts were provided by the authors (personal communication). 1,000 transcripts were randomly selected from the full set and queried with 'bt query' and 'bigsi search -seqlen 8', respectively. Since the SBT was pre-built by the authors, and therefore compressed, we used a RocksDB back end store for BIGSI, which used the cbg Bloom filters were built using the mccortex library, allowing us to compare compressed index size.

Using an array of BIGSIs to support variable dataset size. A limitation of BIGSI is that K_max, the maximum number of k-mers per dataset, must be set in advance. One way to extend BIGSI to datasets with varying k-mer cardinality (e.g., metagenomic data) is to build multiple BIGSIs with different K_max, e.g., K_max = 10^9, 10^10, 10^11, . . . , and insert each sequence into the appropriate level by k-mer counting before insertion. Queries can then be sent to all the indexes in parallel and the results combined.

**Genotyping accuracy measurement on M. tuberculosis.** Conservative SNP calls were made using Cortex (independent workflow, k = 31) on 3,480 Mycobacterium tuberculosis datasets from Walker et al.27. Singletons were discarded, and a de-duplicated list of 68,695 SNPs was constructed. We generated "probe sets" consisting of a reference and alternate alleles of these variants from the NC_000962.3 reference. An index of the 3,480 datasets was built and 100 random datasets with those generated with the samtools pipeline from Walker et al.27, excluding filtered positions. As described in the main text, the concordance labelled extrapolated data points.

Using a 1 Gb cache size and 8 CPUs. A full table of the search results can be found in Supplementary Data 5.

**Conjugative system search.** T4SS sequences (VirD4, TRA_U and VirD4_ TCA) as defined in Guglielmini et al.28 and Supplementary Data 8 in the all-microbial index with T = 100%. Full search results are available in Supplementary Data 9. Results were filtered for bacteria and contamination following the same method as described in "Plasmid search." Accessions with at least one MOb and T4SS were said to contain a putative conjugative system. BIGSI does not return copy number or location on chromosome or plasmid, so it was not possible to determine whether the genes were colocated on a chromosome or on a plasmid.

**MCR genes.** We searched for MCR-1, MCR-2 and MCR-3 in the all-microbial index using k-mer percent threshold T = 100%. See Supplementary Data 1 for sequences and results.

**Searching for ABR genes in the ENA.** We downloaded all 2,157 sequences associated with antimicrobial resistance from the CARD database (v1.1.7)46. We searched for these in the all-microbial index with thresholds of 100% and 70%, using a 1 Gb cache size and 8 CPUs. A full table of the search results can be found in Supplementary Data 5.

**Searching for M. tuberculosis variants in the ENA.** We searched the all-microbial index for the variants from the catalog described in ref. 27 by generating "probe sets"
consisting of a reference and alternate alleles of these variants from the NC_000962.3 reference and searching for these alleles. If only a reference allele is present, the genotype is returned as 0/0; if only an alternate allele, 1/1; if both, 0/1; and if neither, –/–. From the resulting genotypes, we classified each of the datasets as resistant or susceptible to 12 antibiotics following the model described in ref. 27. The date when these data were first available to the public was extracted from their ENA metadata. Datasets were classed as MDR (multidrug resistant) if resistant to isoniazid and rifampicin; as XDR (extensively drug resistant) if MDR and also resistant to a fluoroquinolone and any of capreomycin, kanamycin and amikacin; as resistant if resistant to any antibiotic but not MDR or XDR; and susceptible otherwise.

Trade-offs and limitations in BIGSI. BIGSI can be considered a colored de Bruijn graph, but we would note that it is optimized for search (genotyping), not traversal of the graph. Although one could search for remote homologs using exact matching of short seeds followed by graph traversal, this would require additional software development to handle false positive edges. BIGSI is able to achieve speed, compression and accuracy by using queries that are longer than the indexing k-mer size (here 31)—the relatively high false-positive rate at a single k-mer drops exponentially with each extra unique k-mer in the query. This does, however, mean that short or low-complexity queries should only be used in circumstances where a higher error rate is acceptable (e.g., we recommend ≥61 bp queries and ≥21 unique k-mers for the all-microbial index).

Reporting Summary. Further information on experimental design is available in the Nature Research Reporting Summary linked to this article.

Code availability
An open source implementation of BIGSI can be found at https://github.com/phelimb/BIGSI. BIGSI v0.3.0 supports disk-based indexing via Berkeley-DB or rocksDB, as well as distributed in-memory (via redis (https://redis.io)) key-value stores, and can be extended to any key-value store. The benchmarking uses the rocksDB key-value store and v0.2.0 of BIGSI, and the all-microbial index uses Berkeley DB and BIGSI version v0.1.7.

Data availability
All of the underlying genomic data for this study are publicly available at the ENA, and Supplementary Data can be found in the directory http://ftp.ebi.ac.uk/pub/software/bigsI/nat_biotech_2018. Supplementary Data 1–9 can be found at http://ftp.ebi.ac.uk/pub/software/bigsI/nat_biotech_2018/supp or at https://figshare.com/s/b3653816c9d9550e361da. Contents are as follows: Supplementary Data 1, MCR search results; Supplementary Data 2, plasmid search results; Supplementary Data 3, counts of five specific plasmids across genera; Supplementary Data 4, counts of MOB types across genera; Supplementary Data 5, CARD antibiotic resistance gene search results (T = 70%); Supplementary Data 6, benchmarking results; Supplementary Data 7, Bracken taxonomic results; Supplementary Data 8, MOB type definition fasta; Supplementary Data 9, MOB and T4SS search results (T = 100%). The all-microbial index itself is available at http://ftp.ebi.ac.uk/pub/software/bigsI/nat_biotech_2018/all-microbial-index/. In order to facilitate reproducibility for others without having to download and process 170 Tb of raw data, we made the 26 Tb of cleaned de Bruijn (binary) graph files for the entire all-microbial index snapshot of the ENA available at http://ftp.ebi.ac.uk/pub/software/bigsI/nat_biotech_2018/ctx. An archive of computational code can be found at http://ftp.ebi.ac.uk/pub/software/bigsI/nat_biotech_2018/bigsI.tar.gz. An archive of the data underlying the figures can be found at http://ftp.ebi.ac.uk/pub/software/bigsI/nat_biotech_2018/figure-data.zip and http://ftp.ebi.ac.uk/pub/software/bigsI/nat_biotech_2018/figure-data/ or as Supplementary Data 10 http://ftp.ebi.ac.uk/pub/software/bigsI/nat_biotech_2018/figure-data.zip. We have also made a public instance of our index of the ENA available at http://bigsI.io, where the user can paste sequence and search. This instance uses BIGSI v0.1.7 (using berkeleyDB) and is hosted by CLIMB (http://www.climb.ac.uk/) on a 3 Tb RAM server.

References
48. Broder, A. & Mitzenmacher, M. Network applications of Bloom filters: a survey. Internet Math. 1, 485–509 (2004).
Reporting Summary

Nature Research wishes to improve the reproducibility of the work that we publish. This form provides structure for consistency and transparency in reporting. For further information on Nature Research policies, see Authors & Referees and the Editorial Policy Checklist.

Statistical parameters

When statistical analyses are reported, confirm that the following items are present in the relevant location (e.g. figure legend, table legend, main text, or Methods section).

- The exact sample size (n) for each experimental group/condition, given as a discrete number and unit of measurement
- An indication of whether measurements were taken from distinct samples or whether the same sample was measured repeatedly
- The statistical test(s) used AND whether they are one- or two-sided
- Only common tests should be described solely by name; describe more complex techniques in the Methods section.
- A description of all covariates tested
- A description of any assumptions or corrections, such as tests of normality and adjustment for multiple comparisons
- A full description of the statistics including central tendency (e.g. means) or other basic estimates (e.g. regression coefficient) AND variation (e.g. standard deviation) or associated estimates of uncertainty (e.g. confidence intervals)
- For null hypothesis testing, the test statistic (e.g. F, t, r) with confidence intervals, effect sizes, degrees of freedom and P value noted. Give P values as exact values whenever suitable.
- For Bayesian analysis, information on the choice of priors and Markov chain Monte Carlo settings
- For hierarchical and complex designs, identification of the appropriate level for tests and full reporting of outcomes
- Estimates of effect sizes (e.g. Cohen’s d, Pearson’s r), indicating how they were calculated
- Clearly defined error bars
  - State explicitly what error bars represent (e.g. SD, SE, CI)

Our web collection on statistics for biologists may be useful.

Software and code

Policy information about availability of computer code

Data collection

All data collected was from public repositories by direct download. All processing was done with the software tool BIGSI (the subject of this paper); accessions specified in the paper.

Data analysis

The paper is about a new software tool, BIGSI, which we make publicly available, open source, under an MIT license, here: https://github.com/phelimb/BIGSI. Precise versions for analyses are specified in the article. v0.1.7 for the all-microbial-index and v0.2.0 for benchmarking. We also used kraken version 0.10.5, bracken version 1.0, mccortex version 0.2, SSBT version 0.1 and SBT commit f7986e45119f9d381d365517628b396b1aefa

For manuscripts utilizing custom algorithms or software that are central to the research but not yet described in published literature, software must be made available to editors/reviewers upon request. We strongly encourage code deposition in a community repository (e.g. GitHub). See the Nature Research guidelines for submitting code & software for further information.

Data

Policy information about availability of data

All manuscripts must include a data availability statement. This statement should provide the following information, where applicable:
- Accession codes, unique identifiers, or web links for publicly available datasets
- A list of figures that have associated raw data
- A description of any restrictions on data availability

All the data is public, all the code is public, all the accessions are given in the supplementary material of the paper. Intermediate files ("graph files") which
Field-specific reporting

Please select the best fit for your research. If you are not sure, read the appropriate sections before making your selection.

☐ Life sciences  ☐ Behavioural & social sciences  ☐ Ecological, evolutionary & environmental sciences

For a reference copy of the document with all sections, see nature.com/authors/policies/ReportingSummary-flat.pdf

Life sciences study design

All studies must disclose on these points even when the disclosure is negative.

| Sample size | This is not a study where sample size affects the statistical validity of a conclusion. We have built a DNA search engine for microbes. Our sample size was all sequenced DNA in the world, N~500,000. |
| Data exclusions | We decided in advance that we intended to restrict analysis to bacteria and viruses. The public repository made download most easy if you downloaded these AND eukaryotic microbes. We therefore were forced to explicitly exclude parasites after download |
| Replication | All the data/code is public. Intermediate files (“graph files”) which dramatically accelerate reproducibility allowing the user to avoid downloading 170 Terabytes of raw data, are available on an EBI ftp site: ftp.ebi.ac.uk/pub/software/bigsi/nat_biotech_2018/ The study does not make claims that need replication in the way that, for example, a GWAS study might. The paper describes a search tool, and shows that it works by simulation and on empirical data including using validation against multiple other methods (e.g. SNP calling, MLST typing). It does not make statistical inferences about a population or sample which would need independent replication in another datasets. |
| Randomization | We had no experimental groups |
| Blinding | There was no need for blinding (no groups) |

Reporting for specific materials, systems and methods

| Materials & experimental systems | Methods |
|---------------------------------|---------|
| n/a  | n/a |
| ☒ Unique biological materials | ☒ ChIP-seq |
| ☒ Antibodies | ☒ Flow cytometry |
| ☒ Eukaryotic cell lines | ☒ MRI-based neuroimaging |
| ☒ Palaeontology | |