Quickly preprocessing and profiling microbial community sequencing data with a Nextflow workflow for metagenomics

We want to seamlessly process and summarize metagenomics data from Illumina or Nanopore technologies. We built a Nextflow workflow that handles common metagenomics tasks and produces useful outputs and intuitive visualizations.

Version 2, published Aug 1, 2023. Originally published May 25, 2023.

Arcadia Science DOI: 10.57844/arcadia-7etp-pj24

Purpose

Metagenomic sequencing of microbial communities can provide evolutionary and ecological insights into uncultivated microbial lineages and their interactions. However, processing metagenomic sequencing data involves several preprocessing steps that can be repetitive and cumbersome. We built a Nextflow workflow, Arcadia-Science/metagenomics, to automate common metagenomics tasks and produce output visualizations and files necessary for downstream decision-making. The products of this pipeline are interactive visualizations reported in an HTML with MultiQC, assemblies, mapped reads, and several output files used to assess taxonomic and functional composition of samples.

We built this pipeline using open-source software and tools, and we hope others will use and add to the tool to suit their own needs.

- This pub is part of the **platform effort**, "Software: Useful computing at Arcadia." Visit the platform narrative for more background and context.
- The **metagenomics Nextflow pipeline** is available in this GitHub repository.

- We've provided two **sample reports** of Illumina and Nanopore metagenomes that we processed from our "Paired long- and short-read metagenomics of cheese rind microbial communities at multiple time points" data set [1].
- We also include **examples of analysis you can do with specific outputs from the workflow**, using cheese metagenome data. The code for that analysis and the resulting figures is available in this GitHub repository, and the associated data is on Zenodo.

The strategy

The problem

Extracting valuable insights from metagenomic sequencing data first requires several preprocessing steps that are often repetitive and time-consuming. We want to quickly preprocess metagenomic sequences from either Illumina or Nanopore technologies by performing quality control (QC), assembly, and taxonomic profiling. This information will help us decide whether or not to move forward with particular microbial community samples for more involved downstream analyses and exploration. Although there are numerous existing solutions for processing metagenomics data, we sought a different approach that encourages the user to pause at critical decision points before moving forward with the analysis.

Our solution

We developed a computational resource that automates QC, assembly, mapping, taxonomic profiling, and functional prediction from raw metagenomic reads obtained through either Illumina or Nanopore technologies. This resource is a Nextflow [2] workflow, named Arcadia-Science/metagenomics. We built this Nextflow workflow with a custom template based on the nf-core template [3].

> The **Arcadia-Science/metagenomics pipeline** is available in this GitHub repository (DOI: 10.5281/zenodo.7972166).

The resource

An overview of the metagenomics workflow

The metagenomics pipeline ingests a sample sheet that includes the sample name and the local path, URL, or URI of either paired-end Illumina reads or Nanopore reads in FASTQ format (Figure 1).

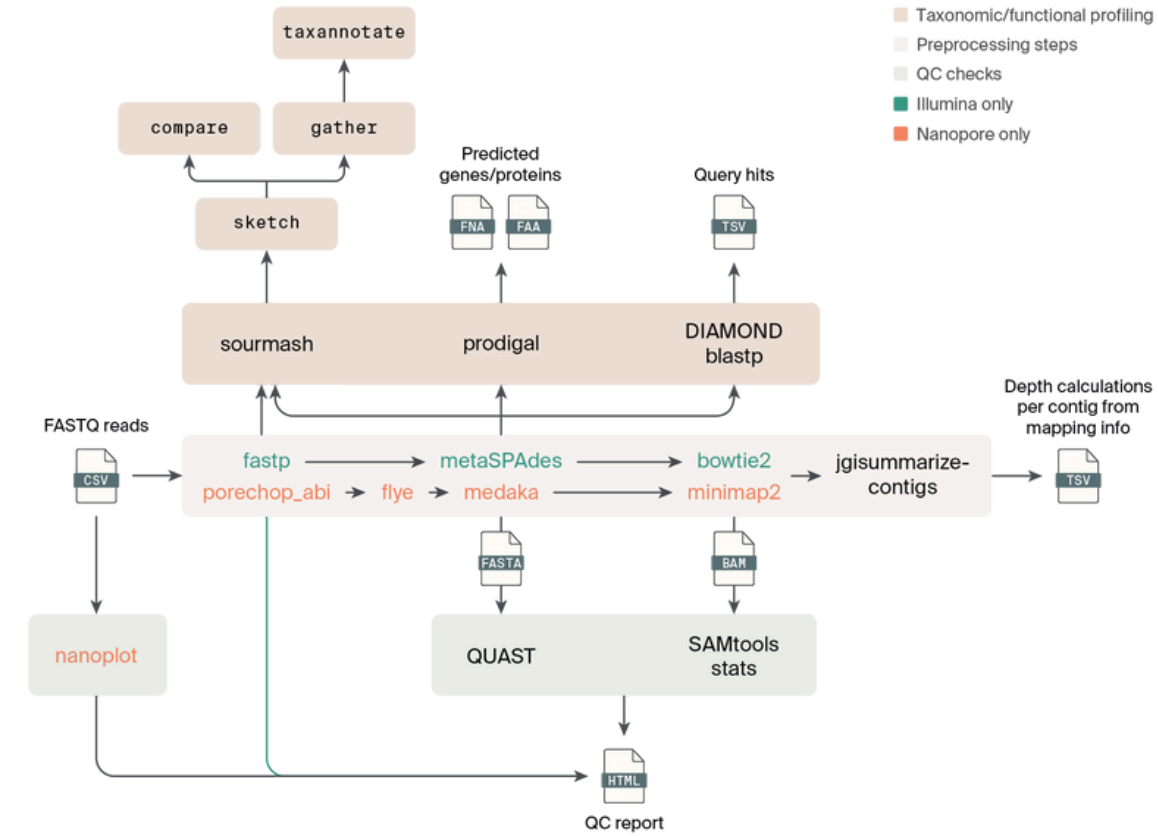


Figure 1. **An overview of the steps in the metagenomics workflow**.
Users provide FASTQ reads from either Illumina or Nanopore technologies in a CSV sample sheet as the input to the workflow. Note that tools that apply only to either Illumina or Nanopore data are highlighted in different colors. The main parts of the workflow encompass common preprocessing steps, QC checks, and taxonomic/functional profiling.

We designed the pipeline to separately process Illumina or Nanopore metagenomic samples, and therefore this pipeline cannot be used to scaffold Illumina assemblies with Nanopore reads or polish Nanopore assemblies with Illumina reads. We made this decision based on our most common internal use case, where we need to separately process Illumina or Nanopore metagenomic

experiments. Additionally, recent updates to Nanopore sequencing chemistries have improved the accuracy of reads and no longer necessarily require polishing with corresponding Illumina reads [4]. Therefore, the user has to select `--platform illumina` or `--platform nanopore` when launching the workflow.

Key functions

After inputting either Illumina or Nanopore reads, the pipeline performs basic read QC and adapter removal, assembly, mapping the reads back to the assembly, and then reports statistics and info about the workflow run in an HTML file produced by MultiQC [5]. Tools specific to either the Illumina or Nanopore workflows are listed below in their respective sections.

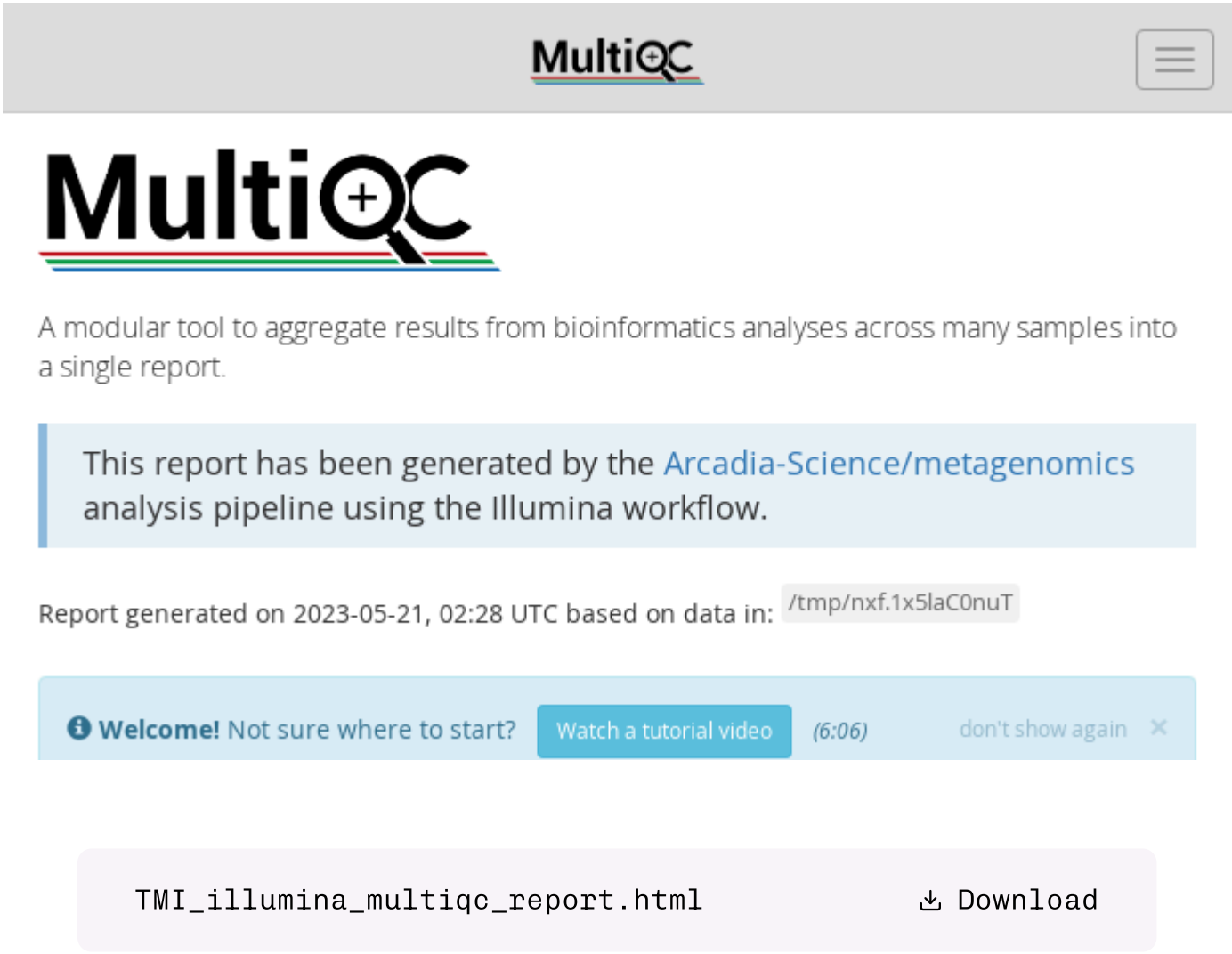
Both workflows report QC stats of assemblies with QUAST [6], summarize mapping rates and alignments statistics with `samtools -stat` [7], predict open reading frames and proteins from the assemblies with Prodigal [8], generate coverage statistics with the program jgisummarizecontigs from MetaBat2 [9], and compare these proteins to the Uniprot Uniref90 database using DIAMOND [10].

The pipeline also launches a series of sourmash commands to produce signatures, compare the signatures against each other, compare the signatures to reference databases, and produce taxonomy summaries based on hits to reference databases [11]. For each sample, we apply these commands to both the reads and assemblies to produce files amenable to comparing samples against each other and exploring taxonomic compositions. We chose to implement sourmash in particular for generating taxonomic summaries due to the ability to rapidly search any sequence input against reference databases [12]. Additionally, we recently created an R package, sourmashconsumr [13], for working with the output files from sourmash and generating intuitive figures (see Figure 2 and Figure 3 below for examples).

Illumina-specific processing

When a user launches the Illumina workflow using `--platform illumina`, it filters each set of paired-end reads with fastp [14] and individually assembles them using SPAdes with the metagenomic option [15]. It then maps the corresponding reads back to each assembly using Bowtie2 [16].

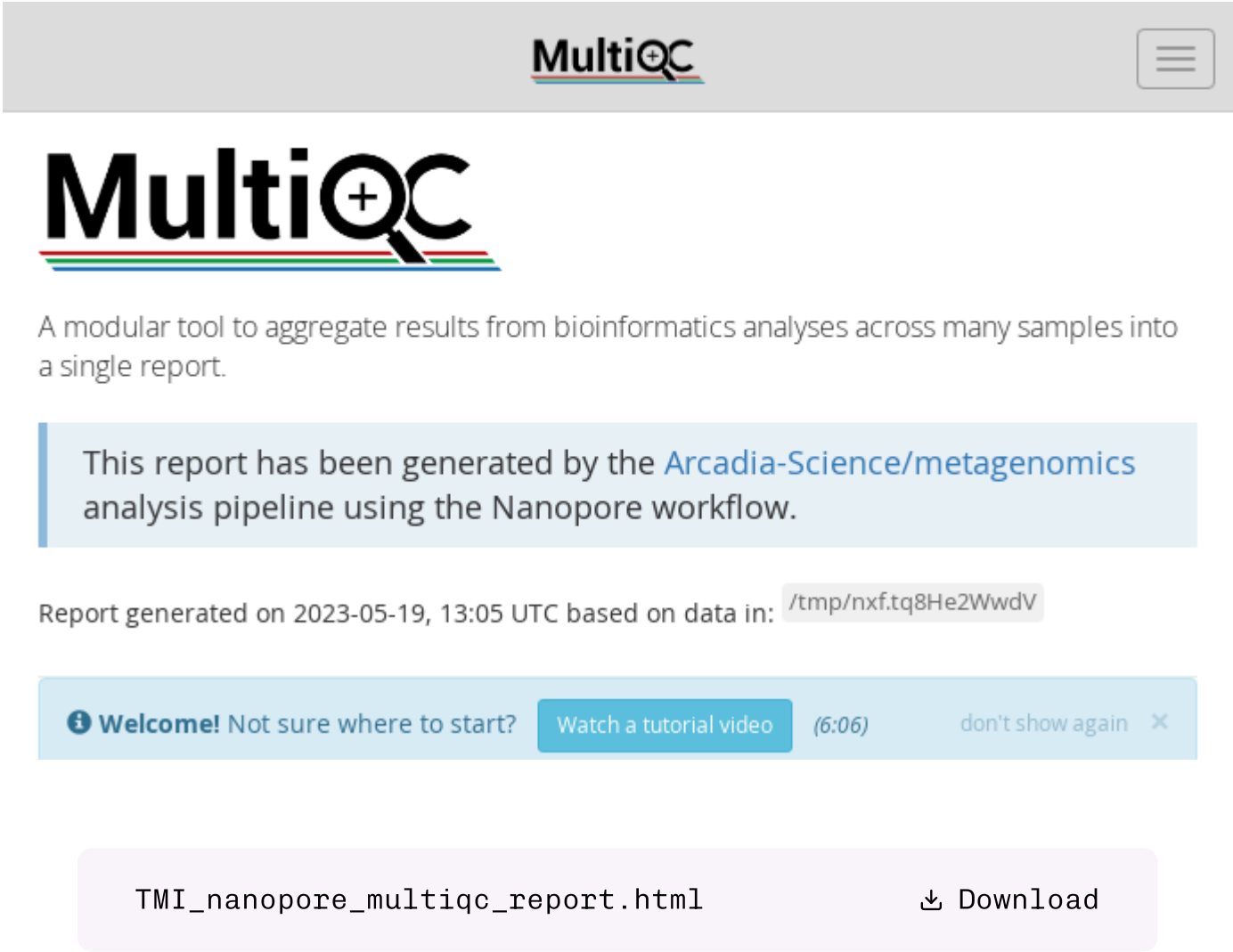
See an example MultiQC report that this workflow generated from short-read Illumina data:



Nanopore-specific processing

When the user launches the Nanopore workflow using `--platform nanopore`, it summarizes each set of Nanopore reads in FASTQ format using Nanoplot [17], removes adapters using Porechop_ABI [18], and individually assembles them using Flye [19] using the `--nano-hq` option. The workflow then polishes assemblies using Medaka with default parameters [20] and maps reads back to each assembly using minimap2 [21].

Here's a sample MultiQC report that the workflow output from long-read Nanopore data:



Deployment

We deploy the pipeline with continuous integration testing using subsampled metagenomic reads from Illumina and Nanopore sequencing efforts of cheese rind microbiomes from our "Paired long- and short-read metagenomics of cheese rind microbial communities at multiple time points" dataset [1]. This ensures that the workflow executes properly as we add new features over time. The pipeline can be run with conda, Docker, or singularity, although we highly recommend using Docker when possible.

We are currently deploying all of our Nextflow workflows, including metagenomics, through Nextflow Tower using our AWS Batch setup [22]. The pipeline is still fully executable locally via the command line and works on diverse compute infrastructure setups.

For most steps in the workflow, we can take advantage of AWS EC2 spot instances to save cost. However, we found that for long-running jobs such as metagenomic

assembly and Nanopore polishing, we needed to modify the workflow to run these processes via on-demand instances so they wouldn't be interrupted. We configured this through setting up queue directives in Tower so that all processes except assembly and polishing will run on AWS EC2 spot instances.

Example taxonomic insights from outputs of the workflow

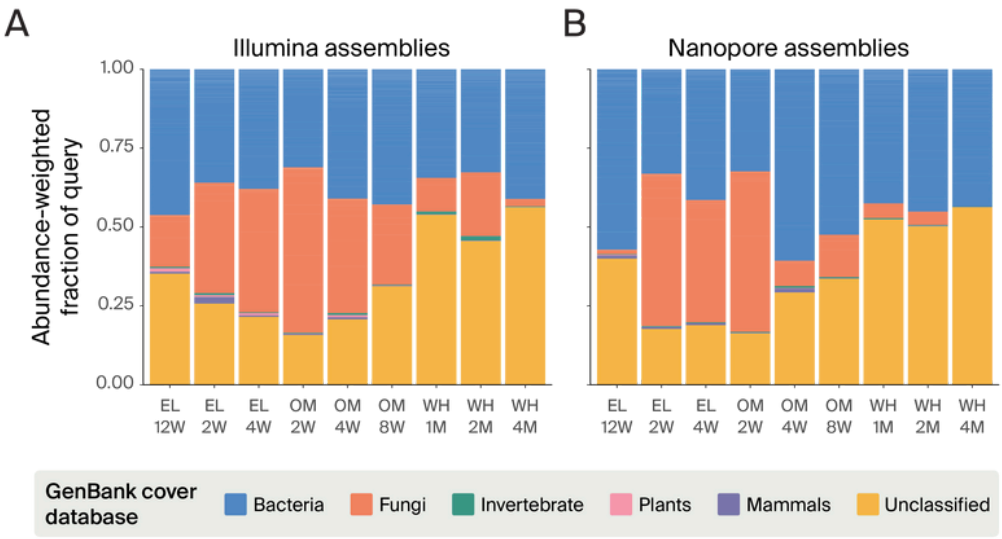


Figure 4. **Proportion of classified sequences in Illumina and Nanopore assemblies**.
We compared our assembled contigs to several "cover databases" (simplified databases that contain each k-mer only once) and used the R package sourmashconsumr to depict the proportion of sequences that sourmash could classify in each database or that remain unclassified for (A) Illumina or (B) Nanopore assemblies. X-axis labels are abbreviated cheese sample names and aging durations (W = week; M = month).

In addition to insights from the MultiQC HTML report, we can use files generated from different sourmash subcommands to quickly inspect metagenomic reads and assemblies. The sourmashconsumr R package provides parsing, visualization, and analysis functions for working with the output files of sourmash. Below, we give two examples of how the outputs of `sourmash gather` and `sourmash taxonomy` summarize the proportion of sequences in either unassembled reads or assembled contigs that are assigned taxonomy based on comparison to a database (Figure 2) and the breakdown of those taxonomic classifications (Figure 3) using data from a prior cheese metagenomics study [1].

The **code** we used for this taxonomic analysis and the resulting figures is available in this GitHub repository (DOI: 10.5281/zenodo.7972177), and the **associated data** is on Zenodo (DOI: 10.5281/zenodo.7968234).

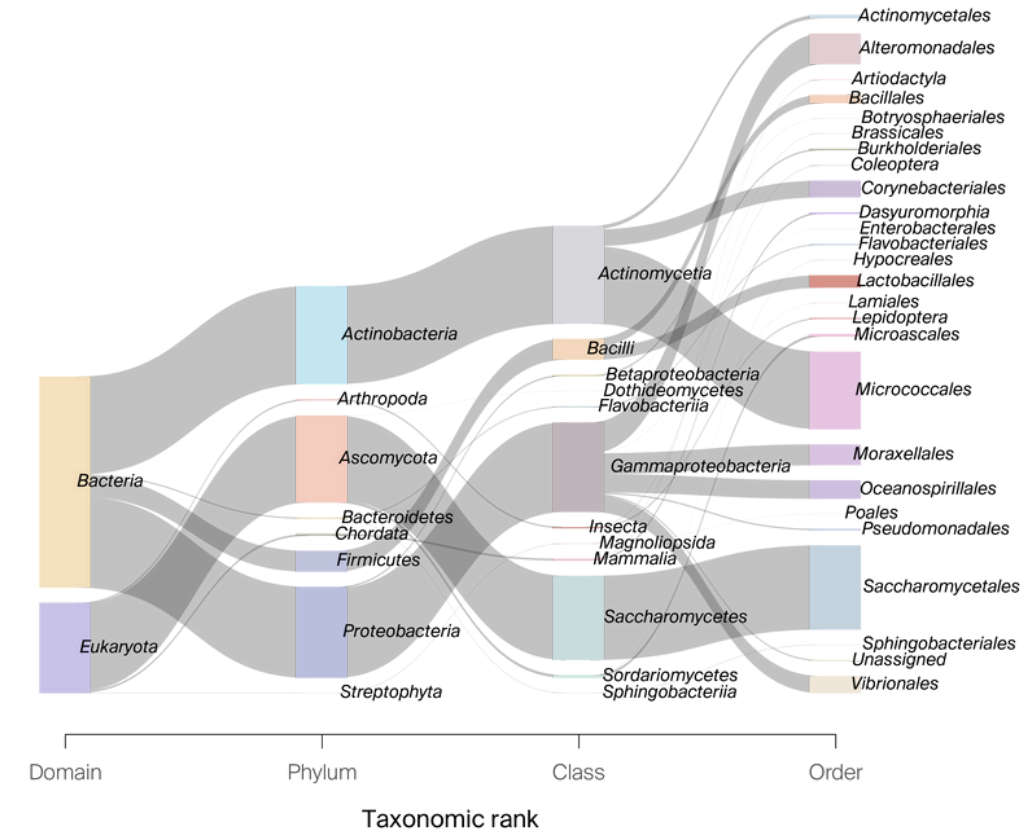


Figure 5. **Breakdown of classified sequences in all Nanopore assemblies**.

We ran `sourmash gather` on all input sample metagenomic reads and assembled contigs against "cover databases" of archaea, bacteria, viruses, fungi, invertebrates, plants, protozoa, and vertebrates available in GenBank using a k-mer size of 31 (Figure 2). Cover databases "cover," or contain, each k-mer in the full database only once [23]. To build a cover database, sourmash sequentially examines each sketch and retains only the hashes that have not been previously observed. This reduces the total database size, which in turn reduces search times and search RAM. In practice, cover databases decreased RAM by an order of magnitude (~124 GB RAM to ~12 GB RAM) and halved runtimes. While less computationally intensive, strain-level assignments are likely inaccurate with cover

databases, so it might be necessary to summarize one level up in taxonomy (i.e. to species).

We compared our assembled contigs to several “cover databases” (simplified databases that contain each k-mer only once) and used the R package sourmashconsumr to depict the proportion of contigs that sourmash could classify in each database or that remain unclassified for (A) Illumina or (B) Nanopore assemblies.

Additional methods

We used ChatGPT to suggest wording ideas and then edited the AI-generated text.

Next steps

The first version of the metagenomics workflow performs common preprocessing tasks that are necessary for downstream steps and analyses of Illumina and Nanopore metagenomic samples. In the future, we would like to:

- Add support for reciprocal mapping of all reads to all assemblies for time-series metagenomics experiments.
- Add support for preprocessing PacBio HiFi metagenomic reads.
- Detect mobile elements such as plasmids and diverse phages beyond those contained in GenBank databases.
- Automate sourmashconsumr reports for comparing samples and taxonomy summaries.
- Build subsequent workflows for binning metagenomic contigs, multi-omics layering, etc.

For some of these efforts, we have created GitHub issues in the metagenomics workflow GitHub repository and welcome outside suggestions and contributions through pull requests!

Contributors (A–Z)

- **Adair L. Borges**: Critical Feedback
- **Feridun Mert Celebi**: Critical Feedback, Validation
- **Rachel J. Dutton**: Supervision
- **Megan L. Hochstrasser**: Editing, Visualization
- **Elizabeth A. McDaniel**: Conceptualization, Formal Analysis, Software, Visualization, Writing
- **Manon Morin**: Critical Feedback
- **Taylor Reiter**: Critical Feedback, Validation
- **Emily C.P. Weiss**: Critical Feedback

References

1. Borges AL, Dutton RJ, McDaniel EA, Reiter T, Weiss EC. (2023). Paired long- and short-read metagenomics of cheese rind microbial communities at multiple time points. https://doi.org/10.57844/arcadia-0zvp-xz86
2. Di Tommaso P, Chatzou M, Floden EW, Barja PP, Palumbo E, Notredame C. (2017). Nextflow enables reproducible computational workflows. https://doi.org/10.1038/nbt.3820
3. Ewels PA, Peltzer A, Fillinger S, Patel H, Alneberg J, Wilm A, Garcia MU, Di Tommaso P, Nahnsen S. (2020). The nf-core framework for community-curated bioinformatics pipelines. https://doi.org/10.1038/s41587-020-0439-x
4. Sereika M, Kirkegaard RH, Karst SM, Michaelsen TY, Sørensen EA, Wollenberg RD, Albertsen M. (2022). Oxford Nanopore R10.4 long-read sequencing enables the generation of near-finished bacterial genomes from pure cultures and metagenomes without short-read or reference polishing. https://doi.org/10.1038/s41592-022-01539-7
5. Ewels P, Magnusson M, Lundin S, Käller M. (2016). MultiQC: summarize analysis results for multiple tools and samples in a single report. https://doi.org/10.1093/bioinformatics/btw354
6. Gurevich A, Saveliev V, Vyahhi N, Tesler G. (2013). QUAST: quality assessment tool for genome assemblies. https://doi.org/10.1093/bioinformatics/btt086
7. Danecek P, Bonfield JK, Liddle J, Marshall J, Ohan V, Pollard MO, Whitwham A, Keane T, McCarthy SA, Davies RM, Li H. (2021). Twelve years of SAMtools and BCFtools. https://doi.org/10.1093/gigascience/giab008

8. Hyatt D, Chen G-L, LoCascio PF, Land ML, Larimer FW, Hauser LJ. (2010). Prodigal: prokaryotic gene recognition and translation initiation site identification. https://doi.org/10.1186/1471-2105-11-119
9. Kang DD, Li F, Kirton E, Thomas A, Egan R, An H, Wang Z. (2019). MetaBAT 2: an adaptive binning algorithm for robust and efficient genome reconstruction from metagenome assemblies. https://doi.org/10.7717/peerj.7359
10. Buchfink B, Xie C, Huson DH. (2014). Fast and sensitive protein alignment using DIAMOND. https://doi.org/10.1038/nmeth.3176
11. Titus Brown C, Irber L. (2016). sourmash: a library for MinHash sketching of DNA. https://doi.org/10.21105/joss.00027
12. Portik DM, Brown CT, Pierce-Ward NT. (2022). Evaluation of taxonomic classification and profiling methods for long-read shotgun metagenomic sequencing datasets. https://doi.org/10.1186/s12859-022-05103-0
13. Chou S, Reiter T. (2023). A new R package, sourmashconsumr, for analyzing and visualizing the outputs of sourmash. https://doi.org/10.57844/arcadia-1896-ke33
14. Chen S, Zhou Y, Chen Y, Gu J. (2018). fastp: an ultra-fast all-in-one FASTQ preprocessor. https://doi.org/10.1093/bioinformatics/bty560
15. Prjibelski A, Antipov D, Meleshko D, Lapidus A, Korobeynikov A. (2020). Using SPAdes De Novo Assembler. https://doi.org/10.1002/cpbi.102
16. Langmead B, Salzberg SL. (2012). Fast gapped-read alignment with Bowtie 2. https://doi.org/10.1038/nmeth.1923
17. De Coster W, D'Hert S, Schultz DT, Cruts M, Van Broeckhoven C. (2018). NanoPack: visualizing and processing long-read sequencing data. https://doi.org/10.1093/bioinformatics/bty149
18. Bonenfant Q, Noé L, Touzet H. (2022). Porechop_ABI: discovering unknown adapters in ONT sequencing reads for downstream trimming. https://doi.org/10.1101/2022.07.07.499093
19. Kolmogorov M, Yuan J, Lin Y, Pevzner PA. (2019). Assembly of long, error-prone reads using repeat graphs. https://doi.org/10.1038/s41587-019-0072-8
20. https://github.com/nanoporetech/medaka
21. Li H. (2018). Minimap2: pairwise alignment for nucleotide sequences. https://doi.org/10.1093/bioinformatics/bty191
22. Celebi FM, McDaniel EA, Reiter T. (2023). Creating reproducible workflows for complex computational pipelines. https://doi.org/10.57844/arcadia-cc5j-a519

23. https://github.com/sourmash-bio/sourmash/issues/1852