Supplementary Method S2: Metagenomic assembly

Two complementary assembly strategies were employed to maximise recovery of metagenome-assembled genomes (MAGs) across the full range of community abundance: per-sample hybrid assembly using both Illumina and Nanopore reads, and iterative Illumina-only co-assembly designed to recover contigs from lower-abundance community members.

Per-sample hybrid assembly

Each sample was assembled independently using **SPAdes 3.15.5** in metagenomic hybrid mode (`--meta`; Bankevich et al., 2012; Antipov et al., 2016), combining paired-end Illumina short reads with the corresponding deduplicated Nanopore long reads. Long reads were provided to SPAdes as supplementary scaffolding information via the `--nanopore` flag. Assemblies were run as SLURM array jobs on the HPC cluster, processing four samples in parallel at 24 CPUs and 250 GB RAM per sample.

Two array jobs were run to cover both sequencing cohorts:

- **Main cohort** (41 samples; PromethION P2Solo, kit14 Nanopore reads): `ARRAY_hybridspades_by_sample.sh`
- **Pilot cohort** (15 samples; MinION MK1C, kit12 Nanopore reads): `ARRAY_hybridspades_by_sample_pilot15.sh`

Illumina co-assembly — Round 1

All paired-end Illumina reads were concatenated across samples into pooled R1 and R2 files and co-assembled using **MEGAHIT** (Li et al., 2015) on the HPC fat memory node (96 CPUs, 1800 GB RAM; script: `megahit.slurm.sh`). A custom k-mer list was specified (`--k-list 33,43,53,63,73,83,93,103,113,123,133`) beginning at k = 33; assemblies using smaller minimum k-mer values failed to complete due to memory errors. The co-assembly output directory was `ohph_megahit_coassembly_output_k33/`.

Contig filtering

Following co-assembly, Illumina reads were mapped back to the contigs to generate per-sample depth of coverage (see Supplementary Method S3). Contigs were retained for downstream binning if they met all of the following criteria, applied using a depth file generated by `jgi_summarize_bam_contig_depths` and filtered with `seqtk subseq` (script: `filter_coassembly.sh`):

- Length 1,000 bp
- Mean per-sample depth 1× in at least 3 samples

This prevalence-based filter was intended to exclude spurious contigs assembled from sequencing error or contamination while retaining genuine low-abundance community members present across multiple samples.

Illumina co-assembly — Round 2 (rare-contig recovery)

To improve recovery of lower-abundance taxa that may be underrepresented in the Round 1 co-assembly, a second co-assembly was performed using reads that did not map to the Round 1 MAG catalogue. Unmapped reads were extracted after mapping all Illumina reads to the Round 1 MAG set (see Supplementary Method S3), then co-assembled using MEGAHIT with parameters tuned for sensitivity to low-coverage sequences (script: `megahit-unmapped.sh`; 64 CPUs, 1800 GB RAM):

```
--k-min 27 --k-max 127 --k-step 10
--min-count 2
--no-mercy
--kmin-1pass
--min-contig-len 1000
```

The lower minimum k-mer (`k-min 27`) and reduced minimum k-mer count (`--min-count 2`) were chosen to maximise assembly of reads from sparse community members. The `--no-mercy` and `--kmin-1pass` flags were set to improve performance under low-coverage conditions. The output of this second co-assembly

(`ohph_megahit_coassembly_unmapped_k27/`) was carried forward directly into binning without an additional contig filtering step.

Contigs from the per-sample hybrid assemblies, the filtered Round 1 co-assembly, and the Round 2 unmapped-read co-assembly were all pooled for dereplication following binning (see Supplementary Methods S3 and S4).

References

- Bankevich, A. et al. (2012). SPAdes: a new genome assembly algorithm and its applications to single-cell sequencing. *Journal of Computational Biology*, 19(5), 455–477. https://doi.org/10.1089/cmb.2012.0021
- Antipov, D. et al. (2016). hybridSPAdes: an algorithm for hybrid assembly of short and long reads. *Bioinformatics*, 32(7), 1009–1015. https://doi.org/10.1093/bioinformatics/btv688
- Li, D. et al. (2015). MEGAHIT: an ultra-fast single-node solution for large and complex metagenomics assembly via succinct de Bruijn graph. *Bioinformatics*, 31(10), 1674–1676. https://doi.org/10.1093/bioinformatics/btv033