SPAAM

Standards, Precautions & Advances in Ancient Metagenomics

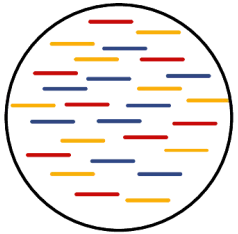
# Practical 4C: Genome assembly

Alexander Hübner and Nikolay Oskolkov

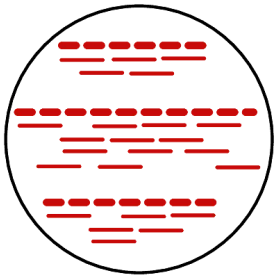
# *De novo* assembly of metagenomic data

pool of metagenomic sequencing data



reference-based alignment

metagenome assembly

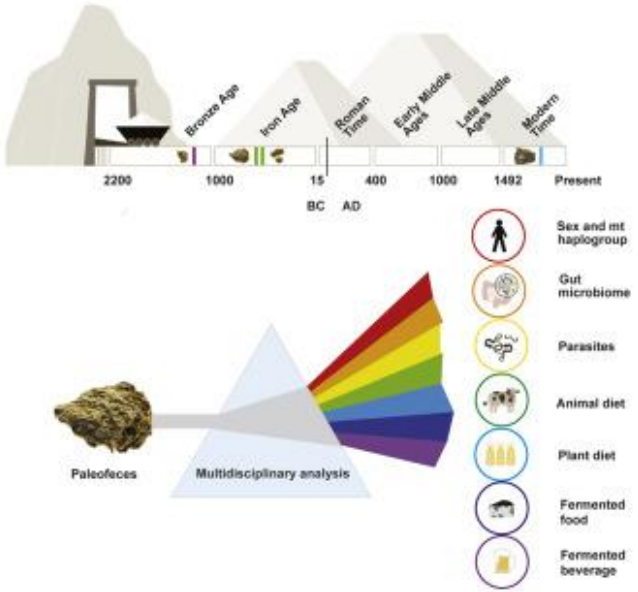


single-genome assembly

reference alignment based analyses **will miss taxons** that are not represented in the database → **possible solutions**:

- cultivation and assembly of isolates → not available for ancient DNA samples
- *de novo* assembly of the metagenomic data

# Sample overview





test sample: a **300-year old palaeofaeces sample**, 2612, excavated from the **Hallstatt salt mine** in Austria

In total, sequenced to a depth of 253 million paired-end DNA reads → **sub-sample** for this practical

Maixner *et al.* (Current Biology, 2021): Hallstatt miners consumed blue cheese and beer during the Iron Age and retained a non-Westernized gut microbiome until the Baroque period (DOI: 10.1016/j.cub.2021.09.031)

# Download the sequencing data

Download the sub-sampled short-read data of 2612 from a local server:

```
wget https://share.eva.mpg.de/index.php/s/CtLq2R9iqEcAFyg/download/2612_R1.fastq.gz
wget https://share.eva.mpg.de/index.php/s/mc5JrpDWdL4rC24/download/2612_R2.fastq.gz
```

or **access them from the local folder**

```
cd /vol/volume/4c-genome-assembly
ls 2612_R1.fastq.gz 2612_R2.fastq.gz
```

**Activate the conda environment:** `conda activate microbial-genomics`

**TASK**: How many sequences are in the FastQ files?
**HINT**: Run `bioawk -c fastx 'END{print NR}' <FastQ file>` to figure out.

# Interactive questions

Please visit the website to submit your answers to the questions:

partici.fi/39619826

# Time-consuming and memory-intensive steps

There are a couple of steps with either a long runtime or require more memory than can be provided on this cluster.

→ **skip the execution of these steps and provide the results**

**Locally available:** `/vol/volume/4c-genome-assembly`

Or via download:

`https://share.eva.mpg.de/index.php/s/y9xyjFfFwK6Xz4P`

Trolley icons created by smalllikeart - Flaticon

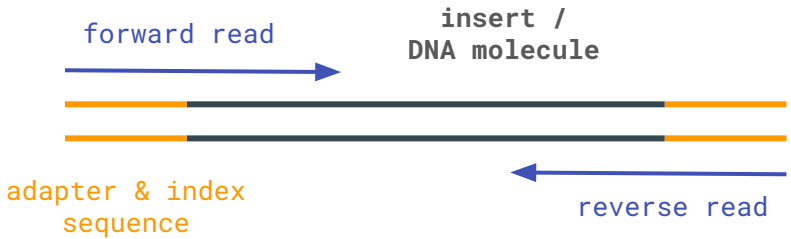
# Download the commands files

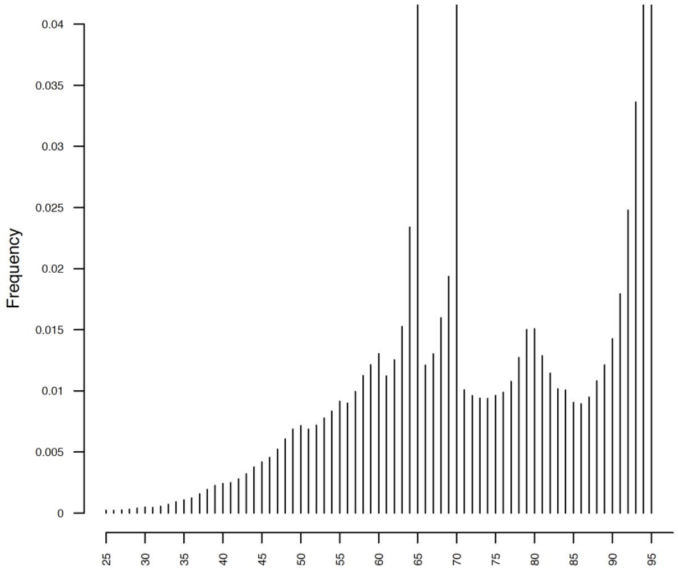
To have all commands readily available:

```
wget https://share.eva.mpg.de/index.php/s/ZTRkanepP8mymca/download/commands.txt
```

# Preparing sequencing data for assembly

*De novo* assembly algorithms use the insert size or **DNA molecule length distribution** as information for improving the assembly quality
→ **paired-end sequencing data** is highly recommended




Example for a **well-preserved horse bone sample**
Orlando *et al.* (Nature, 2013; doi: 10.1038/nature12323) Fig. S4

# Preparing sequencing data for assembly

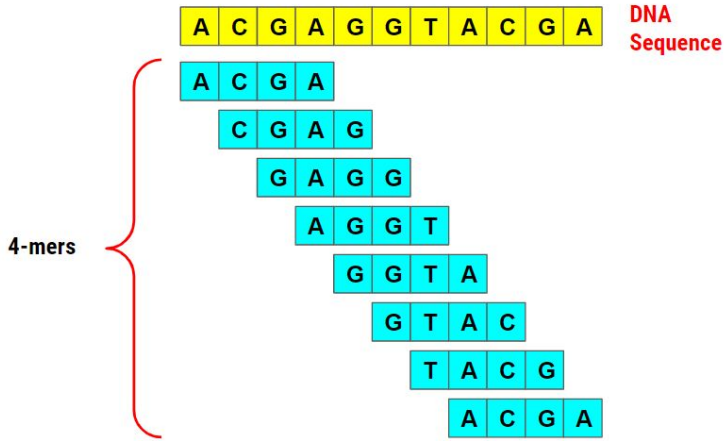
**TASK**: Infer the insert size distribution of the sequencing data

**HINT**: Use **fastp** to merge overlapping read pairs using the command and inspect the HTML report

```
fastp --in1 2612_R1.fastq.gz \
   --in2 2612_R2.fastq.gz \
   --stdout --merge -A -G -Q -L --json /dev/null --html overlaps.html \
   > /dev/null
```

# *De novo* assembly using MEGAHIT

**MEGAHIT**: de Bruijn-graph assembler using a distribution of different k-mer lengths inferred from the length of the sequencing data




https://medium.com/swlh/bioinformatics-1-k-mer-counting-8c1283a07e29

reasons for using MEGAHIT:

- **low-memory** footprint
- has little issues with the **presence of ancient DNA damage**
- works with **single-end data**

BUT: lower assembly quality than other assemblers for modern sequencing data (see CAMI II challenge; DOI: 10.1038/s41592-022-01431-4)

# *De novo* assembly using MEGAHIT

**TASK**: *De novo* assemble the short-read sequencing data using MEGAHIT. Which k-mer lengths did MEGAHIT select?

```
megahit -1 2612_R1.fastq.gz \
   -2 2612_R2.fastq.gz \
   -t 14 --min-contig-len 500 \
   --out-dir megahit
```

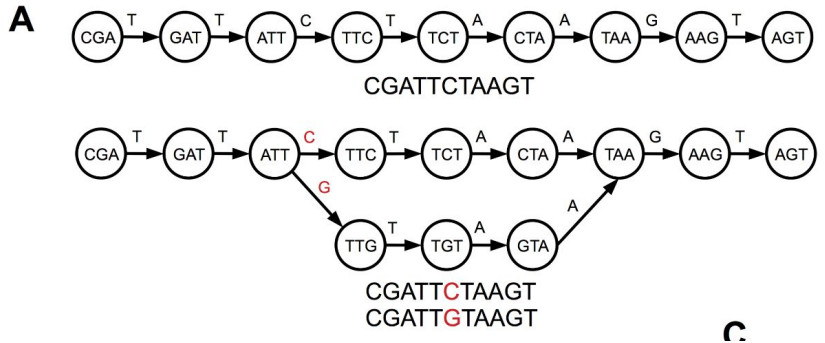
# Ancient DNA damage and *de novo* assembly

MEGAHIT can assemble ancient DNA sequencing data with a high amount of damage but **might introduce damage alleles** into the contig sequences

whole **pipeline** to correct these damage alleles in the contig sequences included in **nf-core/mag**:

- alignment of short-read data against the contigs
- genotyping with freeBayes

→ replace alleles with strong support for an alternative allele




De Bruijn graph with a bubble caused by an 2nd allele
Leggett *et al.* (PLoS One, 2013; doi: 10.1371/journal.pone.0060058) Fig. 1A

nf-core/mag v2.2.0

# Accessing the assembly quality

**TASK**: Summarise the number of contigs, the total contig length, and the maximum (N0), the median (N50), and the minimum contig length (N100) of the assembly produced by MEGAHIT.

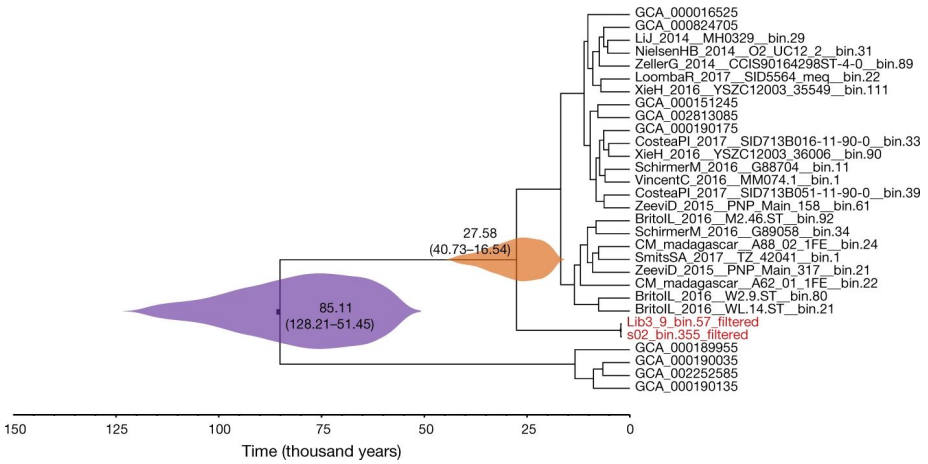
**HINT**: Download the script calN50 and run it on the FastA file

```
wget https://raw.githubusercontent.com/lh3/calN50/master/calN50.js
k8 ./calN50.js megahit/final.contigs.fa
```

# Investigating biological diversity in the sample

There are two major approaches to study the biological diversity in a sample after having *de novo* assembled it:


reconstructing a **non-redundant gene catalogue** for studying the functional diversity



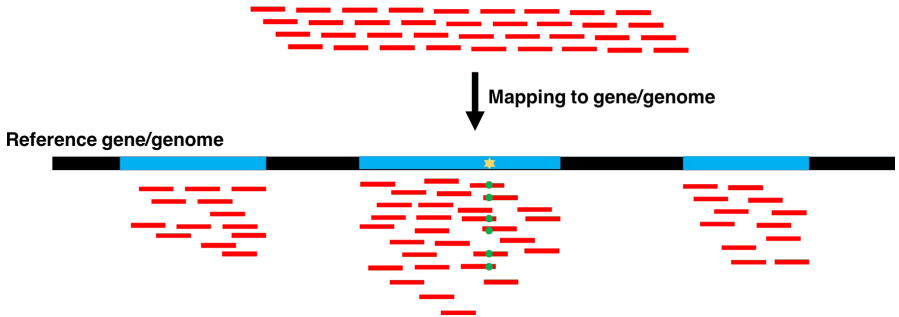

reconstructing **metagenome-assembled genomes** for studying the species diversity

Wibowo *et al.* (Nature, 2021; doi: 10.1038/s41586-021-03532-0)

# Alignment against the contigs

Many of the following steps require the alignment of the short-read data against the de novo assembled contigs, e.g.

- correction of the contig sequences
- binning of the contigs into MAGs (coverage along the contigs)
- quantification of the presence of ancient DNA damage




https://training.galaxyproject.org/training-material/topics/proteomics/images/variant_calling.png

# Alignment against the contigs

**TASK**: Align the short-read data against the contigs using **BowTie2**, sort the resulting alignment file using **samtools sort** and add the MD field using **samtools calmd**

```
mkdir alignment
bowtie2-build -f megahit/final.contigs.fa alignment/2612
bowtie2 -p 14 --very-sensitive -N 1 -x alignment/2612 \
    -1 2612_R1.fastq.gz -2 2612_R2.fastq.gz | \
samtools view -Sb - | \
samtools calmd -u /dev/stdin megahit/final.contigs.fa | \
samtools sort -o alignment/2612.sorted.calmd.bam -
samtools index alignment/2612.sorted.calmd.bam
```

# Alignment against the contigs

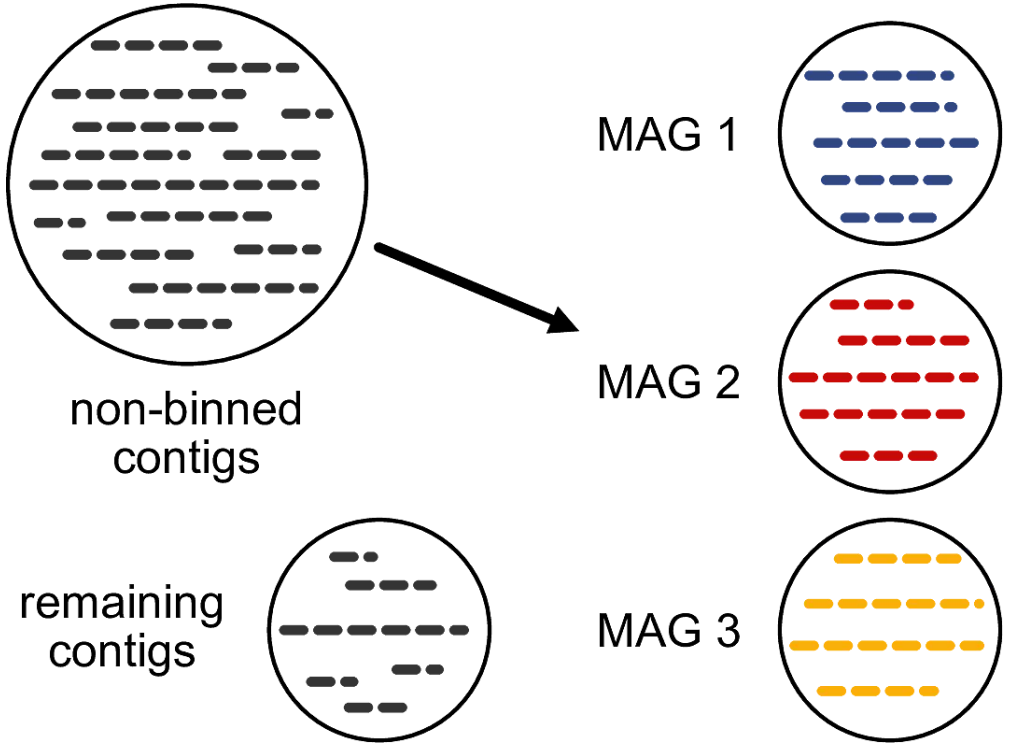
## Link these files from the local server

```
mkdir alignment
ln -s /vol/volume/4c-genome-assembly/2612.sorted.calmd.bam alignment/
ln -s /vol/volume/4c-genome-assembly/2612.sorted.calmd.bam.bai alignment/
ln -s /vol/volume/4c-genome-assembly/2612.fa alignment/
```

or download them:

```
mkdir alignment
wget -O alignment/2612.sorted.calmd.bam \
    https://share.eva.mpg.de/index.php/s/bDKgFLj9GpRFdPg/download/2612.sorted.calmd.bam
wget -O alignment/2612.sorted.calmd.bam.bai \
    https://share.eva.mpg.de/index.php/s/HWqg6fJj6ZEEBAL/download/2612.sorted.calmd.bam.bai
wget -O alignment/2612.fa \
    https://share.eva.mpg.de/index.php/s/z6ZAai42RPribX5/download/final.contigs.fa
```

# Construction of metagenome-assembled genomes




non-reference binning of all contigs by

- **nucleotide frequency** (%A, %C, %G, %T)
- **mean coverage**

into metagenome-assembled genomes (MAGs)

# Initial binning using metaWRAP

MetaWRAP allows convenient binning of the contigs using three binners at the same time:

- metaBAT2 (DOI: 10.7717/peerj.7359)
- MaxBin2 (DOI: 10.1002/cpz1.128)
- CONCOCT (DOI: 10.1038/nmeth.3103)

There are many more binners available (see CAMI II challenge; DOI: 10.1038/s41592-022-01431-4)

# Initial binning using metaWRAP

MetaWRAP is a full pipeline that can also run the assembly automatically → requires some **minor tweaking to skip some steps** such as the alignment

```
mkdir -p metawrap/INITIAL_BINNING/2612/work_files
ln -s $PWD/alignment/2612.sorted.calmd.bam \
    metawrap/INITIAL_BINNING/2612/work_files/2612.bam
mkdir -p metawrap/faux_reads
echo "@" > metawrap/faux_reads/2612_1.fastq
echo "@" > metawrap/faux_reads/2612_2.fastq
```

# Initial binning using metaWRAP

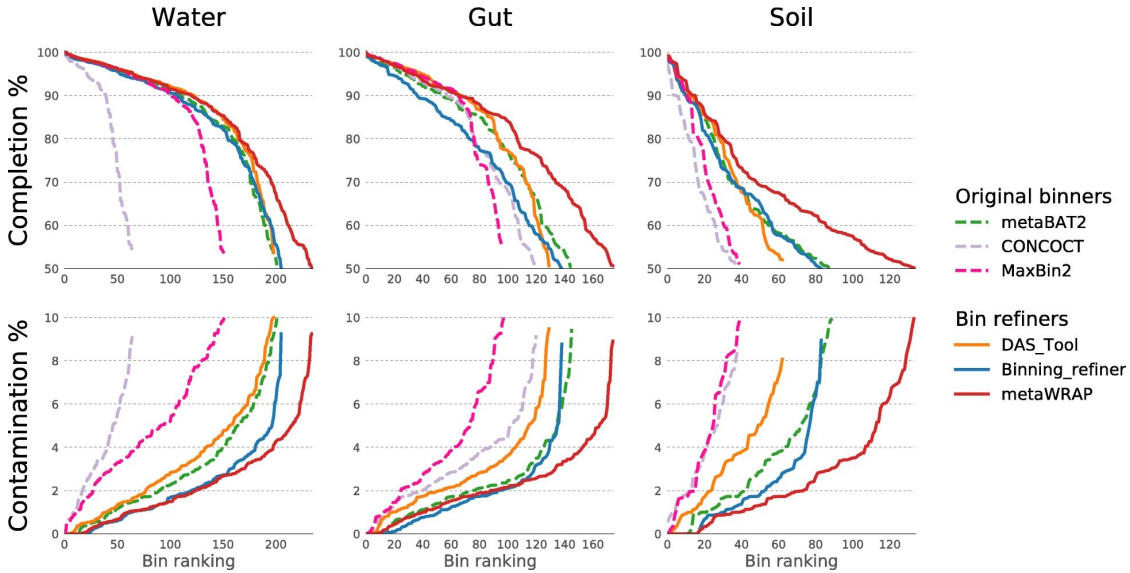
**TASK**: Bin the contigs with the two binners metaBAT2 and MaxBin2 using the **binning module of metaWRAP**. Check the results of each binner and compare the **number of bins** and the **bin sizes** to each other!

```
conda activate metawrap-env
metawrap binning -o metawrap/INITIAL_BINNING/2612 \
    -t 14 \
    -a alignment/2612.fa \
    --metabat2 --maxbin2 --universal \
    metawrap/faux_reads/2612_1.fastq metawrap/faux_reads/2612_2.fastq
conda deactivate
```

**HINT**: You can use the previously introduced script `k8 ./calN50.js` to analyse the bins!

# Bin refinement using metaWRAP

MetaWRAP has a module that uses its own algorithm to **refine the bins** obtained from the three different binning tools




use a **single-copy marker gene database** from the program checkM to **split bins** with contigs from more than one lineages

→ outperforms the individual binners

Uritskiy *et al.* (Microbiome, 2018; doi: 10.1186/s40168-018-0541-1) Fig. 4

# Minimum information for MAG (MIMAG)

MIMAG provides a **standardised checklist** for reporting MAGs and their quality (**completeness** and **contamination**):

| Criterion | Description |
|---|---|
| **Finished (SAG/MAG)** | |
| Assembly quality[a] | Single contiguous sequence without gaps or ambiguities with a consensus error rate equivalent to Q50 or better |
| **High-quality draft (SAG/MAG)** | |
| Assembly quality[a] | Multiple fragments where gaps span repetitive regions. Presence of the 23S, 16S, and 5S rRNA genes and at least 18 tRNAs. |
| Completion[b] | >90% |
| Contamination[c] | <5% |
| **Medium-quality draft (SAG/MAG)** | |
| Assembly quality[a] | Many fragments with little to no review of assembly other than reporting of standard assembly statistics. |
| Completion[b] | ≥50% |
| Contamination[c] | <10% |
| **Low-quality draft (SAG/MAG)** | |
| Assembly quality[a] | Many fragments with little to no review of assembly other than reporting of standard assembly statistics. |
| Completion[b] | <50% |
| Contamination[c] | <10% |
| This is a compressed set of genome reporting standards for SAGs and MAGs. For a complete list of mandatory and optional standards, see Supplementary Table 1. | |

[a]Assembly statistics include but are not limited to: N50, L50, largest contig, number of contigs, assembly size, percentage of reads that map back to the assembly, and number of predicted genes per genome.

[b]Completion: ratio of observed single-copy marker genes to total single-copy marker genes in chosen marker gene set.

[c]Contamination: ratio of observed single-copy marker genes in ≥2 copies to total single-copy marker genes in chosen marker gene set.

Bowers et al. (Nat. Biotech., 2017; doi: 10.1038/nbt.3893)

full version of Table 1: https://www.nature.com/articles/nbt.3893/tables/1

de-facto for estimating the MAG quality: **checkM** (Parks *et al.* (2015; doi: 10.1101/gr.186072.114)

# Preparing checkM for metaWRAP

MetaWRAP requires a working installation of checkM including its database.

```
mkdir checkM
wget -O checkM/checkm_data_2015_01_16.tar.gz \
    https://data.ace.uq.edu.au/public/CheckM_databases/checkm_data_2015_01_16.tar.gz
tar xvf checkM/checkm_data_2015_01_16.tar.gz -C checkM

echo checkM | checkm data setRoot checkM
```

# Bin refinement using metaWRAP

**TASK**: Refine the bins obtained from CONCOCT, metaBAT2, and MaxBin2 using the **refinement module of metaWRAP**. How many bins were kept after the refinement step? How well do the score regarding the MIMAG criteria?

```
mkdir -p metawrap/BIN_REFINEMENT/2612
metawrap bin_refinement -o metawrap/BIN_REFINEMENT/2612 \
    -t 14 \
    -c 50 \
    -x 10 \
    -A metawrap/INITIAL_BINNING/2612/maxbin2_bins \
    -B metawrap/INITIAL_BINNING/2612/metabat2_bins \
    -C metawrap/INITIAL_BINNING/2612/concoct_bins
```

**HINT**: Check the table `metawrap_50_10_bins.stats`

# Visualising tables on the terminal - visidata

visidata provides a **table calculation program** that can be used to visualise, sort, search, or manipulate tabular data **on the terminal** → Excel for the command line

Install visidata: `pip install visidata`

Open a table on the command line: `vd -f tsv metawrap_50_10_bins.stats`

A lot of functionality → check the documentation: https://www.visidata.org/docs/

VISIDATA

# Taxonomic classification - on contig level

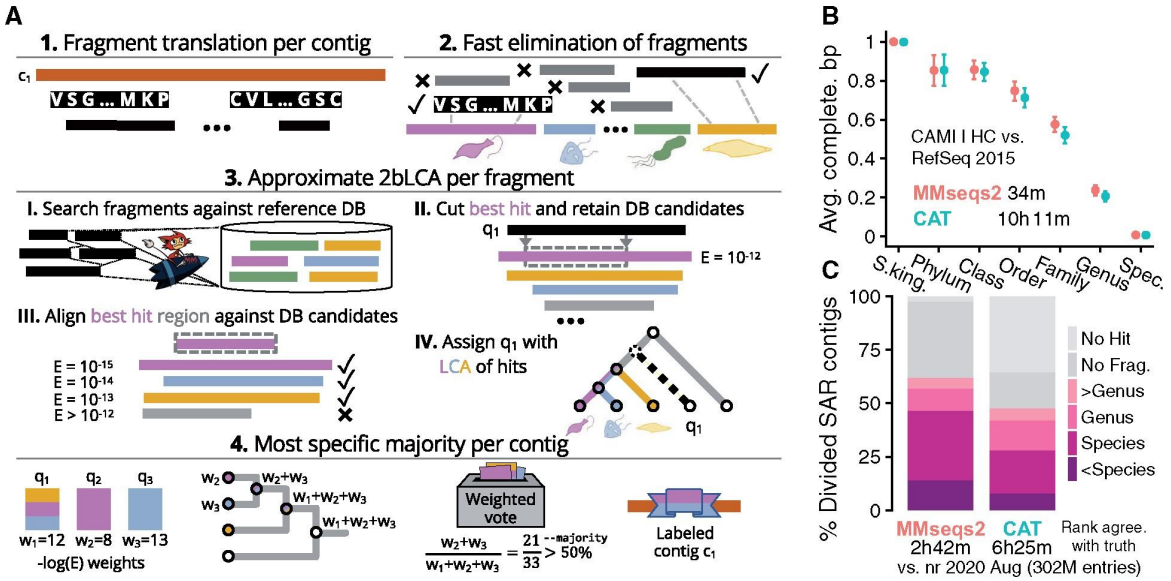
The likely taxonomic origin of contigs can be determined by aligning them against a reference database.

available aligners:

- BLAST/DIAMOND
- Kraken2
- Centrifuge
- **MMSeqs2**

available databases:

- NCBI NT/RefSeq
- **GTDB**




Mirdita *et al.* (Bioinformatics, 2021; doi: bioinformatics/btab184) Fig. 1

# Taxonomic classification - MMSeqs2

Installation of the precompiled GTDB database for MMSeqs2:

```
mkdir -p refdbs/mmseqs2/gtdb
mmseqs databases GTDB \
    refdbs/mmseqs2/gtdb /tmp --threads 14
```

MMSeqs2 requires a large amount of disk space for storing the database (~ 78 GB) and requires a lot of memory to run (~ 500 GB)

→ alternative for less powerful computers: **KrakenUniq**
(https://github.com/fbreitwieser/krakenuniq)

# Taxonomic classification - MMSeqs2

**TASK**: Run the **MMSeqs2 classify workflow** using the **GTDB** to assign the contigs!
What is the proportion of contigs that can be assign to the rank species, genus etc.?
What are the dominant taxa?

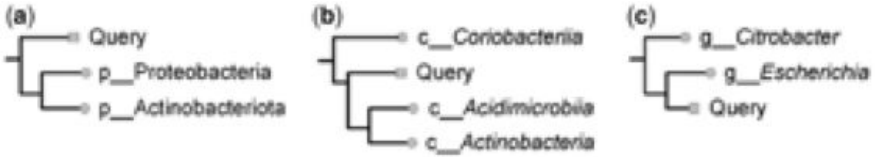
```
mkdir mmseqs2
mmseqs createdb alignment/2612.fa mmseqs2/2612.contigs
mmseqs taxonomy  mmseqs2/2612.contigs \
    refdbs/mmseqs2/gtdb/mmseqs2_gtdb \
    mmseqs2/2612.mmseqs2_gtdb \
    /tmp \
    -a --tax-lineage 1 \
    --lca-ranks kingdom,phylum,class,order,family,genus,species \
    --orf-filter 1 \
    --remove-tmp-files 1 \
    --threads 14
mmseqs createtsv mmseqs2/2612.contigs \
    mmseqs2/2612.mmseqs2_gtdb \
    mmseqs2/2612.mmseqs2_gtdb.tsv
```

**HINT**: Check the TSV file `2612.mmseqs2_gtdb.tsv`

# Taxonomic classification - on MAG level

The exact taxonomic classification of MAGs is more complicated than just aligning all contigs against a reference database. **GTDBTK** provides a more sophisticated approach:

1. **Lineage identification** based on single-copy marker genes using Hidden Markov models (HMMs)
2. **Multi-sequence alignment** for these marker genes
3. Placement of MAG genome into a **fixed reference tree** at class-level




Chaumeil *et al.* (Bioinformatics, 2019; doi: bioinformatics/btz848) Fig. 1

# Taxonomic classification - GTDBTK

Installation of the precompiled GTDB database for GTDBTK:

```
mkdir -p refdbs/gtdbtk
wget -O refdbs/gtdbtk/gtdbtk_v2_data.tar.gz \
    https://data.gtdb.ecogenomic.org/releases/latest/auxillary_files/gtdbtk_v2_data.tar.gz
tar xvf refdbs/gtdbtk/gtdbtk_v2_data.tar.gz -C refdbs/gtdbtk
```

requires about ~ 70 GB of hard drive storage space

# Taxonomic classification - GTDBTK

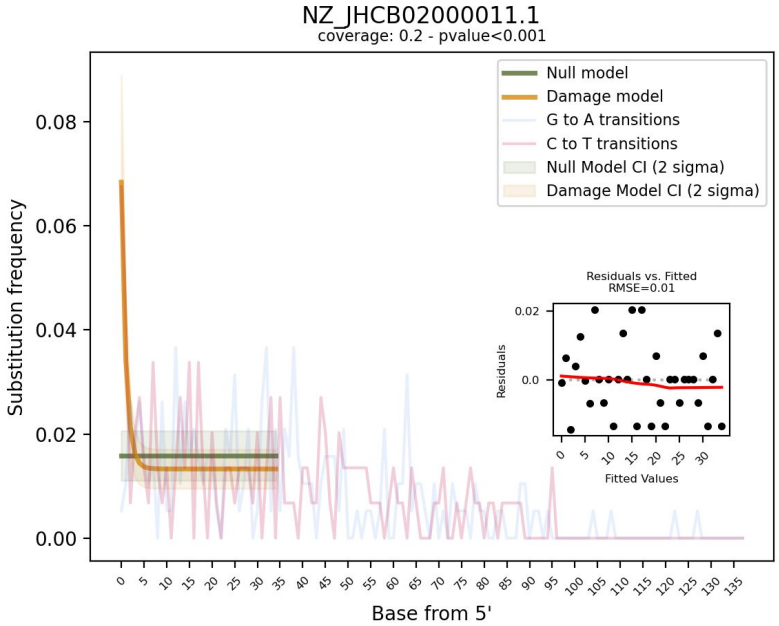
**TASK**: Classify the MAGs refined with metaWRAP to the **GTDB taxonomy** using GTDBTK. Do these classifications match the assignments obtained from MMSeqs2? Would you expect these taxa given the **archaeological context of the sample**?

```
mkdir gtdbtk
GTDBTK_DATA_PATH="$PWD/refdbs/gtdbtk/gtdbtk_r207_v2" \
gtdbtk classify_wf --cpu 14 --extension fa \
    --genome_dir metawrap/BIN_REFINEMENT/2612/metawrap_50_10_bins \
    --out_dir gtdbtk
```

**HINT**: Check the TSV files `2612.gtdbtk_archaea.tsv` and `2612.gtdbtk_bacteria.tsv`

# Evaluating the amount of aDNA damage

PyDamage evaluates the **amount of aDNA damage** and **tests the hypothesis** whether a model assuming the presence of aDNA damage better explains the data than the null model




PyDamage

# Evaluating the amount of aDNA damage

**TASK**: Evaluate the pyDamage results with the respect of the **amount of C-to-T substitutions** observed on the contigs, the **number of contigs** considered as being "ancient", and how much power there was for these decisions ("**prediction accuracy**")! Are their MAGs that are strongly "ancient" or "modern"?

**HINT**: Run pyDamage on the sorted BAM file and check the CSV file

```
pydamage analyze -w 30 -p 14 alignment/2612.sorted.calmd.bam
```

Download table:

```
wget https://share.eva.mpg.de/index.php/s/awaE9Ss4WsRm6wm/download/pydamage_results.csv
```

# Annotating genomes

So far, we only have the nucleotide sequences and their likely origin.
**Functional elements** that require annotation:

- coding sequences
- transfer or ribosomal RNAs
- CRISPR sequences

can be conveniently achieved using **Prokka**

# Annotating genomes using Prokka

**TASK**: Annotate the genome of the MAG `bin.3.fa` using Prokka. What type of files does Prokka return? How many genes/tRNAs/rRNAs were detected?

```
prokka --outdir prokka \
    --prefix 2612_003 \
    --compliant --metagenome --cpus 14 \
    metawrap_50_10_bins/bin.3.fa
```

**HINT**: Check the file `prokka/2612_003.txt`

# Summary

- *de novo* assembly of ancient short-read sequencing data
- non-reference binning of the contigs based on nucleotide frequency and mean coverage
- bin refinement and quality evaluation using single-copy marker genes
- taxonomic classification against the GTDB
- analysis of aDNA damage on contig level
- genome annotation

# Cautionary note - sequencing depth

This sample was non-randomly **subsampled** from the original sequencing data of sample 2612:

- the tutorial sample: < 5 million reads
- the original sample: 196.25 million reads

→ genome assembly need **a lot of sequencing data** than reference-based profiling **BUT** also dependent on **the complexity of sample**

**GIVE IT A TRY!**

# Useful resources

- https://nf-co.re/mag: Nextflow nf-core pipeline for the *de novo* assembly and non-reference binning of short-read sequencing data that is suitable for ancient DNA
- anvi'o: tool suite for the analysis and visualisation of 'omics data that allows for the manual curation of MAGs