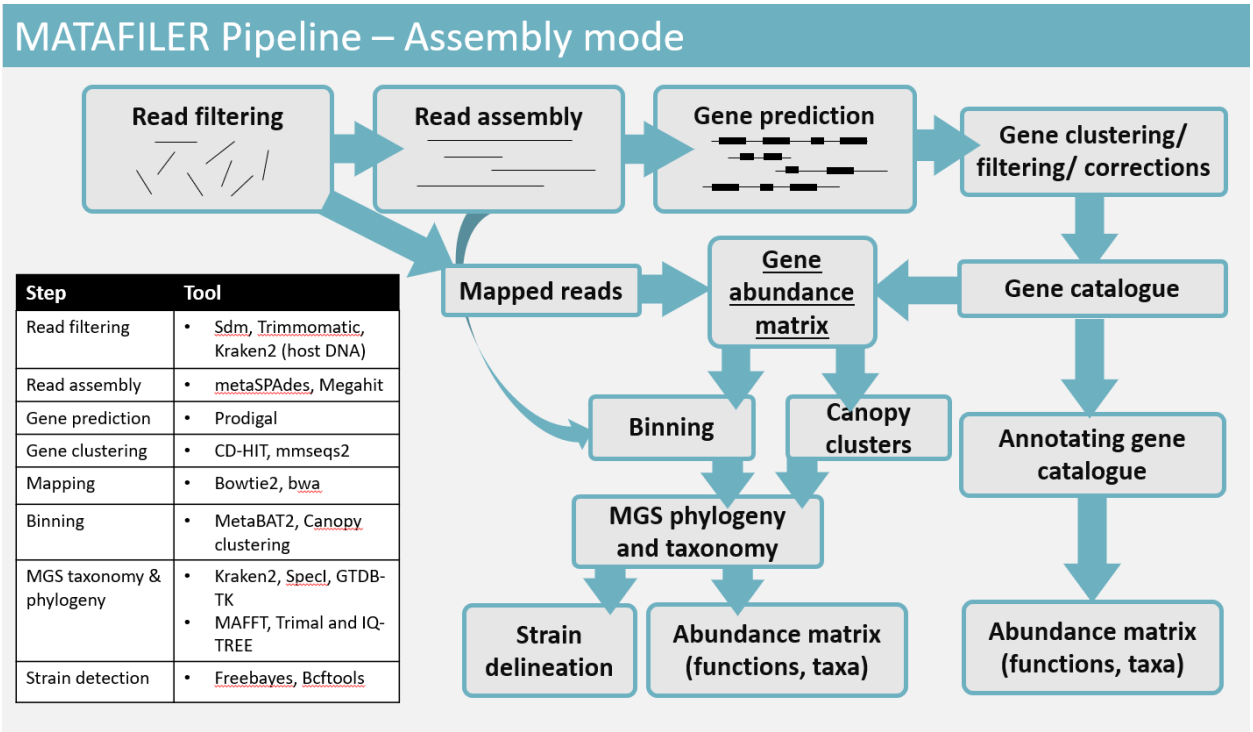
MATAFILER pipeline

Developed in Hildebrand lab, QIB, EI, Technical outline of program, input and output

Comments and questions, please address to falk.hildebrand@gmail.com



Step	Tool
Read filtering	• Sdm, Trimmomatic, Kraken2 (host DNA)
Read assembly	• metaSPAdes, Megahit
Gene prediction	• Prodigal
Gene clustering	• CD-HIT, mmseqs2
Mapping	• Bowtie2, bwa
Binning	• MetaBAT2, Canopy clustering
MGS taxonomy & phylogeny	• Kraken2, Specl, GTDB-TK • MAFFT, Trimal and IQ-TREE
Strain detection	• Freebayes, Bcftools

Figure 1 Overview of core bioinformatic workflow in MATAFILER pipeline to get to a gene catalog and MGS (metagenomic species).

Pipeline Introduction

MATAFILER (abbr. MF in this document) is under active development since 2014, with the aim to standardise and automate the raw sequence analysis of metagenomic experiments. Here assembly-free and assembly-dependent analysis has to be differentiated. Briefly, assembly-free analysis was developed for hard to assemble communities of a complexity, that simply cannot be assembled, such as soil metagenomes, examples of this are in [1]. For gut and other host associated microbiomes, species complexity usually allows for an assembly-dependent workflow, i.e. in these situations >80% of all reads can be represented through the metagenomic assembly (see Fig 1 for a typical assembly-dependent workflow).

Input to pipeline

Inputs are the raw metagenomic reads and a mapping file describing the samples, every other step is done by the pipeline, and flags describing the analysis to be done on the metagenomic reads.

A typical call would look like:

```
MAP=/hpc-home/hildebra/dev/Perl/MATAF3/maps/T2D.map #mapping file, define in variable "$MAP"
perl /hpc-home/hildebra/dev/Perl/MATAF3/MATAFILER.pl -map $MAP \
-inputFQregex1 '.*_1\.f[^\.]*q\.gz$' -inputFQregex2 '.*_2\.f[^\.]*q\.gz$' -inputFQregexSingle
'single.*\.fastq\.gz$' -mergeReads 0 \ #regular expression to recognize read pairs based on file names
-assembleMG 2 -spadesCores 12 -spadesKmers "25,43,67,87,111,131" -spadesMemory 100 -
binSpeciesMG 1 \ #option for the  assembler, megahit in this case
-mapReadsOntoAssembly 1 -profileFunct 0 -reParseFunct 0 -reProfileFunct 0 -diamondDBs ABRc -
kmerPerGene 1 \ #backmapping of reads to assembly
-wcKeyJobs f59b0dc4-a00f-48c6-a03a-067087106369 \ #EI specific tag to register job
-filterHostKrak2DB /hpc-home/hildebra/DB/kraken2/hsap/ \ #filtering for human reads (can be any org)
-saveReadsNotMap2Assembly 0 -remap2assembly 0 -maxConcurrentJobs 1100 \ #flow management
-getAssemblConsSNP 0 -redoAssmblConsSNP 0 \ #SNP calls?
-profileRibosome 0 -reProfileRibosome 0 \ #miTAags? -> deactivated here
-profileMOTU2 0 -profileMetaphlan2 0 \ #motus2, metaphlan? -> deactivated here
-inputReadLength 150 \
-mappingMem 5 \
-submit 1  \ #submit or try "dry" run first?
-from 940 -to 950 #maybe test on 10 samples (940-949 from mapping file) first before going big..
```

More flags can be defined for additional analysis (about 200 flags are in total available), please ask Falk about this, if needed.

Mapping file

The mapping file is an integral part of the analysis and needs to be created by hand. It is very similar in structure and purpose to the LotuS (lotu2.earlham.ac.uk) mapping files, but adapted to metagenomics data. A typical mapping file would look like:

```
#SmplID  Path      AssmblGrps      IDX      offNm    SmplPrefix          externalID
#OutPath/ei/workarea/users/hildebra/projects/
#RunID   IBD
#Gut microbiota composition and functional changes in inflammatory bowel disease and irritable bowel syndrome

#DirPath  /ei/projects/115b210e-aa45-4469-9eb1-d0d85d879fc0/data/raw/metag/EGAD00001004194/allFQs/
BD00001                                    1000IBD00001
BD00004                                    1000IBD00004
#DirPath  /ei/projects/115b210e-aa45-4469-9eb1-d0d85d879fc0/data/raw/metag/EGAD00001001991/
NHLI1     EGAF00001150355/
NHLI2     EGAF00001150356/
```

Mapping files are tab-delimited text files (important: LINUX EOL!). The first line (the header) determines what kind of information will be available in a mapping file, only the first entry (#SmplID) needs to be in this position, the columns can otherwise come in any orientation. In the above example, MATAFILER specific tags are in **bold**. This example contains 4 total samples (that can contain numerous paired- or

single read fastq(.gz)) from two different studies, the studies are saved in different folders. Available columnnames in the map are:

#SmplID Path SmplPrefix MapGrps AssmblGrps SeqTech ReadLength EstCoverage SupportReads ExcludeAssembly

#SmplID will be the sample name that the study is saved as in all resulting abundance matrices, assemblies, gene predictions, gene catalogs etc, it is important to chose this carefully. As a general rule, only alphanumeric characters (a-zA-Z0-9) should be used for sample IDs, special characters cannot be used ("-_+=*.,~!" etc should not be used). Since SmplID will be saved multiple times and for readability, it is recommended to save easy to remember names that are short and descriptive.
Path SmplPrefix are the primary form to communicate file locations, files associated to each SmplID. Path refers to a folder (#DIRPATH/Path/), SmplPrefix to the prefix of a file (#DIRPATH/SmplPrefix*). Several files can be associated to each sample, whether these are read1, read2 or single reads depends on the file ending, defined by the flag set in "-inputFQregex1" etc. Expected input format is either fastq, fastq.gz or fasta.
MapGrps AssmblGrps Samples can be grouped together for either Assembly (AssmblGrps) or mapping (MapGrps). Assembly groups are convenient for e.g. longitudinal data, grouping an individual represented by several samples. Mapping groups are rarely used and mostly designed in case the same biological sample is represented by several technical samples, but it is more practical to just copy all reads belonging to the same biological sample to the same folder.
SeqTech ReadLength sequencing technology used, can be proto, GAII_solexa, GAII, miSeq, hiSeq, ONT or PB. If left empty, hiSeq is assumed. Depending on seq tech, different sdm filtering options will be used, e.g. GAII (early illumina) has very different error profile from modern miSeq/hiSeq sequencers. ONT/PB (3rd gen sequencers) will also trigger a different assembler to be used
EstCoverage ExcludeAssembly EstCoverage refers to an old algorithm, that uses kmers to estimate the coverage of genomes within the metagenomic reads. Excl Assembly, set to 1 or 0 to exclude some samples from being assembled.
SupportReads In case mate pairs/3rd gen seq data is available and should be used to support the assembly, this can be entered here. Quite specialized option, talk to Falk before using this.

Output of pipeline

Depending on the analysis, very different outputs can be provided, but in general these are **feature abundance matrices**, that is tab delimited files with samples as column names and features (genes, functions, taxa) as row names.

Assembly dependent analysis

Single (or groups of in case of e.g. longitudinal data) samples are assembled using either MegaHIT or MetaSPAdes, genes are called on assemblies, reads are mapped to these assemblies to calculate the abundance of single genes. All genes are pooled from the assembled samples wtihin a study, and subsequently clustered into a set on non-redundant genes, the **Gene catalog**. Before a gene catalog is created, the single metagenomic samples need to be assembled, genes called on these assemblies and

reads backmapped to assemblies (to estimate gene/contig abundance). This is can be done with the MATAFILER.pl scripts (see Fig. 1 for an overview).

Sample assemblies

Samples assemblies (and all associated analysis) are stored in the directory #OutPath defined in the mapping file. Within this dir also other results are stored, such as "pseudoGC" (pseudo gene catalog, not entirely correct any longer) and global LOG dirs. pseudoGC is important for most non-assembly dependent analysis results (*miTAGs*, mOTU2, metaPhlan2, *functional assignments*) are being stored in easily accessible tsv format here. However, most space occupying are assemblies and postprocessing, stored in a folder with the same name as the sample ID, so "#OutPath/#SMPLID/". Let's look at an example sample "DLF0001".

```
(MFF) [hildebra@NBI-HPC DLF001]$ ls
assemblies  input_fil.txt  input_raw.txt  LOGandSUB  mapping
```

This is a minimal example and based on MATAFILER options a lot more folders/subfolders than this example can be created. assemblies contains the assemblies, filter by size (200bp default) as well as assembly stats:

```
                          0 May 29 10:54 ass.done.sto
                       3.5K May 29 10:54 AssemblyStats.500.txt
                       3.5K May 29 10:54 AssemblyStats.ini.txt
                       3.5K May 29 10:54 AssemblyStats.txt
                         94 Jun  5 11:17 assembly.txt
                        198 May 29 10:54 checkpoints.txt
                        590 Jun  5 12:34 ContigStats
                          0 May 29 10:54 done
                        129 Jun  5 12:31 genePred
                       110K May 29 10:54 megaAss.log
                       1.2K May 29 10:54 options.json
                        46M May 29 10:54 scaffolds.fasta.filt
                       509K May 29 10:54 scaffolds.fasta.filt2.gz
                       754K Jun  4 13:13 scaffolds.fasta.filt.fai
                        15M May 29 10:54 scaffolds.fasta.gz
                       273K May 29 10:54 scaffolds.fasta.lnk.gz
assemblies/metag/:      77 May 29 10:54 smpls_used.txt
```

Within this dir are contigstats (related to abundance) and genePred (gene predictions, can be eukaryotic or prokaryotic, depending on MF options):

```
                                  1.7M Jun  5 11:18 Coverage.count_pergene
                                  568K Jun  5 11:18 Coverage.median.percontig
                                  1.7M Jun  5 11:18 Coverage.median.pergene
                                  499K Jun  5 11:18 Coverage.percontig
                                  1.7M Jun  5 11:18 Coverage.pergene
                                     0 Jun  5 11:18 Coverage.stone
                                   174 Jun  5 11:18 Coverage.window
                                  8.0K Jun  5 12:32 ess100genes
                                  2.4K Jun  5 12:31 FMG
                                   163 Jun  5 11:18 GeneStats.txt
                                  6.0M Jun  5 12:33 scaff.4kmer.gz
                                  152K Jun  5 11:18 scaff.GC3.gz
                                  150K Jun  5 11:18 scaff.GC.gz
                                   15M Jun  5 12:34 scaff.pergene.4kmer.gz
                                   17M Jun  5 12:35 scaff.pergene.4kmer.pm5.gz
                                  377K Jun  5 11:18 scaff.pergene.GC3.gz
assemblies/metag/ContigStats/:    371K Jun  5 11:18 scaff.pergene.GC.gz
```

Ess100genes and FMG [2] are both sets of essential marker genes, FMG is thoroughly used through various MF routines, detailed later. “Coverage.*” are various forms of counting gene/contig coverage and the rest of the files contains nucleotide statistics for the genes/contigs that are used in binning.

```
                                   17M May 29 10:58 genes.gff
                                  437K May 29 10:58 genes.per.ctg
                                   42M May 29 10:58 genes.shrtHD.fna
assemblies/metag/genePred/:        15M May 29 10:58 proteins.shrtHD.faa
```

In the gene predictions folders, mostly statistics, and NT genes (genes.shrtHD.fna) as well as AA genes (proteins.shrtHD.faa) can be found, the gff file is essential to later backtrack genes to their contig of origin and used in gene catalog creations, strain delineation etc. The “shrtHD” indicates that the header was rewritten to MF format, this format is essential for the pipeline to retrack genes across samples, e.g. consider:

```
DIYHFELPSSKGVL*
>DLF001__C1263_L=114017=_35
MFFRIISILLMQKQIPLCLLAIVLSVSLFGQNLKADKIILSDSDLTNLYQIDSGVYRSEQ
PSKEGFKALEKYGIGEVLNLRNRHSDDDEAKGTSIKLHRVKTKAHSISEKQLIQALRIIK
NRKAPIVFHCHHGSDRTGAVCAFYRIIFQNVSKEDAIHEMTEGGYGFHRIYKNIIRRIKE
ANVEQIRKEVMEGGEL*
>DLF001__C1263_L=114017=_36
MNYGFVKVAAAVPHVKVADCKFNVEKIESLIAVAEGKGVQIIIFPEMSITGYTCGDLFGQ
QLLLEEAEMGLMQILNNTRQLDIISIVGMPVVVNSTVINAAAVIQKGKVLGVTAKTYLPN
YKEFYEQRWFTSALQLTTNSVRLCGQIVPIGSNLLFETSDTTFGIEICEDLWSTIPPSSS
```

The name of genes can be used to know the origin sample (DLF001), the contig and it’s length (C1263_L=114017) as well as the gene number on this contig (36). Beyond backtracking this system is used in various scripts to ascertain orientations and contig linkage of genes. The “mapping” directory contains

```
1.5M Jun  5 12:30 DLF001-smd.bam.bai
105M Jun  5 12:30 DLF001-smd.bam.coverage.gz
600M Jun  5 12:30 DLF001-smd.cram
   0 Jun  5 12:30 DLF001-smd.cram.sto
  15 Jun  5 12:30 done.sto
```

Sample named mappings of the reads (cram). Beyond this there is the option to store unaligned reads, which would be stored in this directory, in subdir “unaligned” if user requested.

If these files are present in a dir, then the sample is ready to be incorporated into the gene catalog.

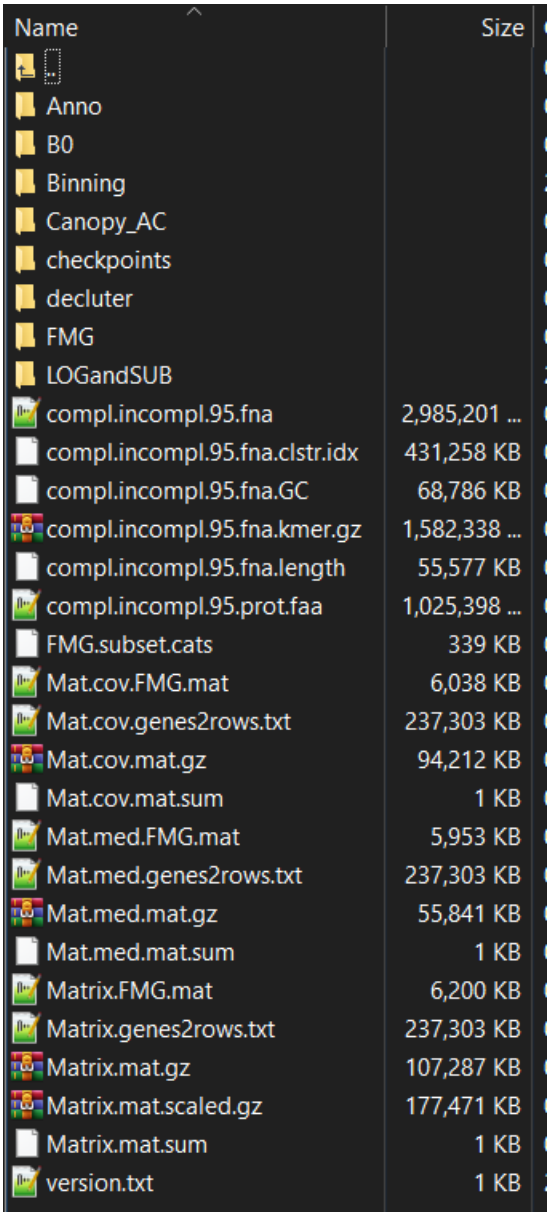


Figure 2 Typical file structure of MATAFILER gene catalog

Gene catalog

Gene catalogs, first described in work from the MetaHIT consortium [3], provide a set of non-redundant genes (clustered at 95% sequence identity) that are found across all samples. These represent therefore in theory all genes that microorganisms encode on their genomes. The gene abundance matrix records the abundance of these genes to single sample, and an abundance of 0 would mean this gene is not present in a given sample (typically gene catalogs are zero enriched).

The gene catalog algorithm implemented in MATAFILER is described in more detail in [4], a typical file structure is shown in (Fig2). *compl.incompl.95.fna* and *compl.incompl.95.prot.faa* are the nucleotide and AA representation of genes clustered at 95%. To capture further characteristics of these genes, their GC content (.GC), 4-mer frequencies (.kmer) and length (.length) are recorded and used in some analysis.

The central gene abundance matrix is stored in *Matrix.mat.gz*, (gzipped to save space), recording counts, i.e. reads that map, to each gene in each sample. *Matrix.mat.scaled.gz* is the same matrix, but normalized, that is each column sums to 1. *Matrix.mat.sum* is the sum of each sample, so the column sum of *Matrix.mat.gz*. Last, *Matrix.FMG.mat* is a subset of *Matrix.mat.gz*, containing only marker genes of bacteria, that is a set of 40 marker genes that are present in single copies within species [5]. Of this matrix, there are several variants, concerned with how a gene abundance is actually calculated: *Mat.cov.mat.gz* is the coverage (reads mapping to gene / length of gene) of each gene, *Mat.med.mat.gz* is the median coverage of all bp within a gene (which is theoretically more stable towards outliers). In practice, the normal *Matrix.mat.gz*

has proven to work well in most cases. However, these are rather technical files and summaries of these should be preferred for analysis. Two main information sources summarize the gene catalog: MGS (metagenomic species) and “bag-of-genes” interpretations of a metagenome (Fig. 3). MGS are in subfolder *Binning/*, bag-of-genes in folder *Anno/*.

Gene catalog annotations and summary tables

Functions

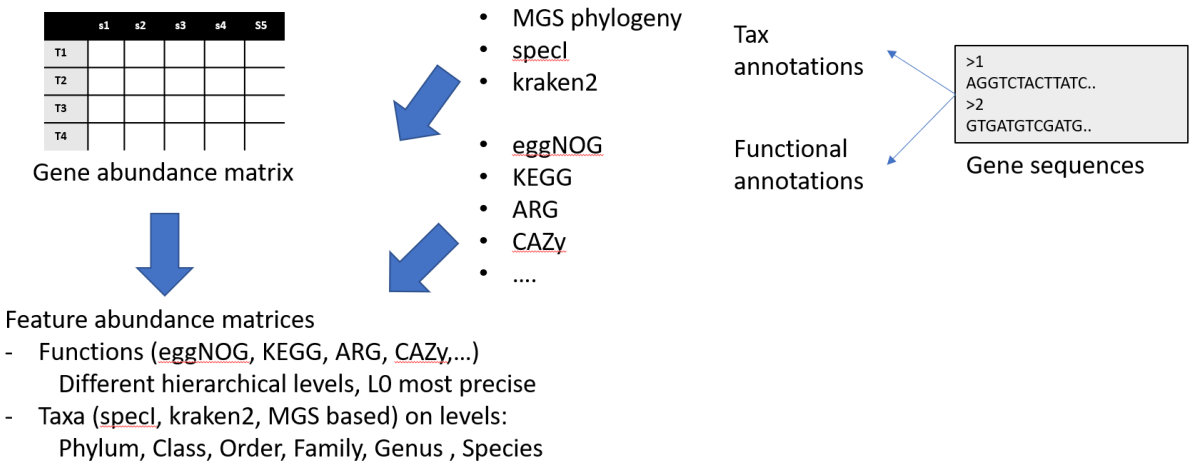


Figure 3 "Bag-of-genes" style summaries of genes and their abundances across samples. Feature abundance matrices always have the same structure, having as row names the taxa/function that is represented, column names the samples analysed, and entries the abundance in a given sample. “?” and “-1” correspond to unknown categories (?, e.g. some bacteria don’t have a genus level name, but a gene can be associated to them) or unassigned reads (-1, some genes are not taxonomically/functionally annotated at all) in a sample.

In the folder *Anno/*, two subfolders and divisions can be found, *Anno/Tax/* and *Anno/Func/*. Depending on the algorithms and reference databases chosen these will contain differing files, but in principle the following files are usually found in the *Anno/Func/* folder:

```
ABRcL0.txt*                        DIAass_KGM.srt.gz.stone*
ABRcL1.txt*                        DIAass_NOG.srt.gz*
CNT_1e-08_25/                      DIAass_NOG.srt.gzgeneAss.gz*
CZyL0.txt*                         DIAass_NOG.srt.gz.stone*
CZyL1.txt*                         DIAass_TCDB.srt.gz*
CZyL2.txt*                         DIAass_TCDB.srt.gzgeneAss.gz*
CZyL3.txt*                         DIAass_TCDB.srt.gz.stone*
CZyL4.txt*                         KGML0.txt*
CZyL5.txt*                         NOGL0.txt*
CZyL6.txt*                         NOGL1.txt*
DIAass_ABRc.srt.gz*                NOGL2.txt*
DIAass_ABRc.srt.gzgeneAss.gz*      TCDBL0.txt*
DIAass_ABRc.srt.gz.stone*          TCDBL1.txt*
DIAass_CZy.srt.gz*                 TCDBL2.txt*
DIAass_CZy.srt.gzgeneAss.gz*       TCDBL3.txt*
DIAass_CZy.srt.gz.stone*           TCDBL4.txt*
DIAass_KGM.srt.gz*                 TCDBL5.txt*
DIAass_KGM.srt.gzgeneAss.gz*       TCDBL6.txt*
```

The prefix denotes the reference databases (*ABR*=antibiotic resistance genes, *CZy*=CAZy, *KGM*=KEGG, *NOG*=eggNOG, *TCDB*=transporter database), folder *CNT_1e-08_25* and *DIAass_* (Diamond assignments) are primary annotations that are saved for technical purposes only. "L" in the file name denotes the hierarchical level of the feature abundance matrix, going from precise (L0) to most broad summary (L1,…). Feature abundance matrices have the standard annotation format.

Taxonomy

Taxonomic abundance matrices derived from gene abundances and their taxonomic annotation are stored in *Anno/Tax/* folder. *krak* and *specI,* as prefix, serves to define the method (sometimes in subfolders), L denotes taxonomic level, or taxonomic level is directly defined. In general, I recommend to use *specI* abundances, but this is also a bit a matter of taste.

Metagenomic assembled genomes (MAGs) and metagenomic species (MGS)

Directory: [gene_catalog]/Binning/

MATAFILER compiles from the single sample assemblies and gene catalogs MAGs, that are clustered at species level into MGS. Briefly, this is based on MetaBAT2 [6] and canopy clustering [7], which are quality filtered and compared among samples to avoid chimeric bins, described in [4]. MAGs are available in primary data folders (so not the gene catalog folder), MGS are available in the *Binning/* dir. A summary stat of MetaBAT2 bins that can be found in each sample is stored in *MAG.MB2.assStat.summary* file.

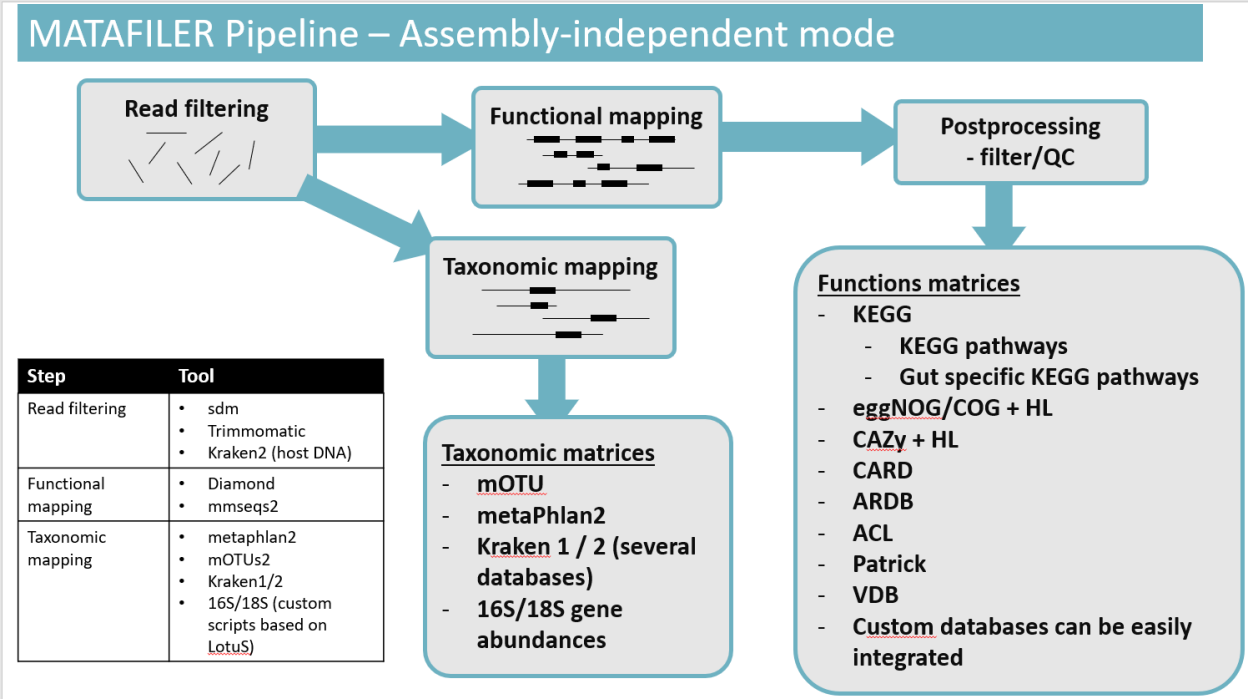
The genes (from the gene catalog) associated to each MGS are stored in *MB2.clusters.ext.can.Rhcl*, a taxonomy is provided *kraken2.LCA* , checkM [8] is used to assess the quality and likely contamination in each MGS and stored in file *MB2.clusters.ext.can.Rhcl.cm*. The directory *Binning/RhclClust/* contains the proteins (.faa) found in each MGS, separately stored for each MGS.

File in *[gene_catalog]/Binning/*	**Purpose**
MB2.clusters.ext.can.Rhcl.matL0.txt	abundance of MGS across samples

MB2.clusters.ext.can.Rhcl	Assignments of gene-catalog-genes to single MGS, a merge of MetaBat2, canopy clustering and custom post-filtering
MB2.clusters.ext.can.Rhcl.cm	CheckM qualities of MGS genomes
GTDBTK.tax	GTDB tax assignments -> this is the most modern algorithm
kraken2.LCA	Kraken2 tax assignments
between_phylo/phylo/IQtree_allsites.treefile	De novo phylogeny of MGS

Principal MGS abundance files can be found in the *[gene_catalog]/Anno/tax/* folder. The MGS based analysis is still under active development, with additional functional annotations and GMM derived functions currently being included in the MATAFILER pipeline.

Assembly independent metagenomics



Taxonomic and functional annotations can also be derived from raw (or filtered) metagenomic reads. Standard metagenomic software for deriving taxa abundances falls into this category. For example, kraken2 [9] can be used on reads directly, mOTUs2 [10] is in principle similar to the specI approach and metaphlan2 [11] is widely used. All these approaches suffer in precision due to having to depend on

short reads, but in turn do not suffer from potential biases introduced during assembly. In general, assembly can be considered more precise, as modern metagenomic assemblers are very precise [12], but an assembly based metagenomic analysis is computationally more demanding. mOTUs2, kraken2 and metaphlan2 taxa abundance matrices based on raw reads are available in MATAFILER.

Literature

1 Bahram M, Hildebrand F, Forslund SK, *et al.* Structure and function of the global topsoil microbiome. *Nature* 2018;**560**:233–7. doi:10.1038/s41586-018-0386-6

2 Mende DR, Sunagawa S, Zeller G, *et al.* Accurate and universal delineation of prokaryotic species. *Nat Methods* 2013;**10**:881–4. doi:10.1038/nmeth.2575

3 Qin J, Li R, Raes J, *et al.* A human gut microbial gene catalogue established by metagenomic sequencing. *Nature* 2010;**464**:59–65. doi:10.1038/nature08821

4 Hildebrand F, Moitinho-Silva L, Blasche S, *et al.* Antibiotics-induced monodominance of a novel gut bacterial order. *Gut* 2019;**68**:1781–90. doi:10.1136/gutjnl-2018-317715

5 Sunagawa S, Mende DR, Zeller G, *et al.* Metagenomic species profiling using universal phylogenetic marker genes. *Nat Methods* 2013;**10**:1196–9. doi:10.1038/nmeth.2693

6 Kang DD, Li F, Kirton E, *et al.* MetaBAT 2: An adaptive binning algorithm for robust and efficient genome reconstruction from metagenome assemblies. *PeerJ* 2019;**2019**:1–13. doi:10.7717/peerj.7359

7 Nielsen HB, Almeida M, Juncker AS, *et al.* Identification and assembly of genomes and genetic elements in complex metagenomic samples without using reference genomes. *Nat Biotechnol* 2014;**32**:822–8. doi:10.1038/nbt.2939

8 Parks DH, Imelfort M, Skennerton CT, *et al.* CheckM: assessing the quality of microbial genomes recovered from. *Cold Spring Harb Lab Press Method* 2015;**1**:1–31. doi:10.1101/gr.186072.114

9 Wood DE, Lu J, Langmead B. Improved metagenomic analysis with Kraken 2. *Genome Biol* 2019;**20**:1–13. doi:10.1186/s13059-019-1891-0

10 Milanese A, Mende DR, Paoli L, *et al.* Microbial abundance, activity and population genomic profiling with mOTUs2. *Nat Commun* 2019;:1–11. doi:10.1038/s41467-019-08844-4

11 Truong DT, Franzosa EA, Tickle TL, *et al.* MetaPhlAn2 for enhanced metagenomic taxonomic profiling. *Nat Methods* 2015;**12**:902–3. doi:10.1038/nmeth.3589

12 Vollmers J, Wiegand S, Kaster AK. *Comparing and evaluating metagenome assembly tools from a microbiologist's perspective - Not only size matters!* 2017. doi:10.1371/journal.pone.0169662