Chapter 2

Protein-Coding Gene Families in Prokaryote Genome Comparisons

Dennis Carhuaricra-Huaman and João Carlos Setubal

Abstract

The identification of orthologous genes is relevant for comparative genomics, phylogenetic analysis, and functional annotation. There are many computational tools for the prediction of orthologous groups as well as web-based resources that offer orthology datasets for download and online analysis. This chapter presents a simple and practical guide to the process of orthologous group prediction, using a dataset of 10 prokaryotic proteomes as example. The orthology methods covered are OrthoMCL, COGtriangles, OrthoFinder2, and OMA. The authors compare the number of orthologous groups predicted by these various methods, and present a brief workflow for the functional annotation and reconstruction of phylogenies from inferred single-copy orthologous genes. The chapter also demonstrates how to explore two orthology databases: eggNOG6 and OrthoDB.

Key words Orthology inference, Orthology resources, Phylogeny inference, Functional annotation

1 Introduction

Homology describes an evolutionary relationship between genes and proteins. Two subtypes of homology relationships were recognized by Walter Fitch [1], orthology and paralogy. Orthology refers to genes originating from a speciation event, whereas paralogy refers to genes that arise from a gene duplication event. The original motivation behind identifying orthologs was to carry out phylogenetic reconstruction, since by definition they possess the same evolutionary history as the underlying species. However, nowadays, another motivation for distinguishing between orthologs and paralogs is the prediction of the function of newly annotated genes [2, 3]. Ortholog genes are more likely to have the same function because they were the same gene in the most recent common ancestor.

The identification of orthologs is essential in many applications, such as comparative genomics, phylogenetics, and functional annotation [4]. Currently, high-throughput sequencing methods have increased the availability of complete genomes (and their proteomes), and robust computational methods for identifying orthologs are crucial to explore the vast amount of genomic data [5].

Computational methods for identifying orthologous sequences can be in general divided into two groups: phylogeny-based and graph-based methods [6]. Phylogeny-based methods require aligning a group of homologous sequences, computing a phylogenetic tree, and inferring the type of evolutionary event represented by each internal node in the generated tree, a process that involves reconciling the gene tree with the species tree [7]. While this method tends to be more accurate, it requires more computational work and therefore is not easily scalable. On the other hand, graph-based approaches rely on sequence similarity searches and consider two sequences orthologous if they are each other's best hit in their respective proteomes [6, 8]. Orthologous Groups (OGs) (see below) are then built by running a clustering program on the underlying graph, in which a vertex is a sequence and there is an edge between two vertices if the respective sequences have similarity above the threshold used in the similarity search. Several software tools combine both methods using attributes of graph-based and tree-based methods in the inference of orthology relationships [9].

In this chapter, we cover the topic of computational inference of orthology from a practical point of view. There are basically two ways this can be done: (a) using orthology inference programs with user-defined proteomes and executed locally; (b) using existing orthology database resources on a website. We demonstrate the two methods using publicly available proteomes from ten prokaryotic species.

This chapter covers the case in which the genomes to be compared are distantly related. For closely related genomes (e.g., strains of the same species), we refer the reader to the chapter *Step-by-step Bacterial Genome Comparison* in this same volume.

2 Requirements and Assumptions

This chapter assumes basic knowledge of Unix/Linux and R. All analyses can be run on a desktop computer running Linux/Unix or Mac OSX. Most programs are run using bash shell commands. We show commands executed on the Linux shell preceded by the "$" symbol and the label *bash shell*. We also present R code, which can be executed in the RStudio environment (https://posit.co/download/rstudio-desktop/). R code sections are preceded by the label *R script*.

3 Datasets

Orthology programs use protein sequences as input to predict orthologous families. This chapter uses sets of protein sequences (proteomes) from ten species of prokaryotes, spanning six different phyla, to present different methods and software tools. Proteomes, which we define as a file with all predicted protein sequences from a given genome, can be found in public repositories, such as GenBank, and by GenBank convention have the extension ".faa." Table 1 provides the accession numbers in GenBank from where the proteomes used in this chapter were downloaded.

4 Software

Table 2 lists the software tools employed in this chapter, along with links to the corresponding websites where they can be downloaded.

5 Generating Orthologs by Various Methods

Two genes are homologous when they share a common ancestor gene. In a somewhat simplified scenario, two genes can have a common ancestor in two situations. The first is caused by speciation. Gene *a* in species *X* is homologous to gene *b* in species *Y* when *X* and *Y* have a common ancestor *Z*, and *Z* had a gene that can be said to be the ancestor of *a* and *b* (by virtue of the fact that *Z* is the ancestor of *X* and *Y*). In this case, we say that not only *a* and *b* are homologous, but they are also *orthologous*, to distinguish it from the second situation. That happens when *a* and *b* are the result of gene duplication. Duplicated genes are named *paralogs*; however, there can be two sub-cases, depending on when duplication takes place with respect to speciation. If duplication occurs *before* speciation, the resulting genes in the descendant species are called *out-paralogs*. If duplication occurs *after* speciation, the duplicated genes in a given species are called *in-paralogs*. When analyzing several genomes, as is the case in this chapter, genes that are orthologous among all pairs of genomes are grouped together as orthologous groups (OGs), also called orthogroups.

The inference of OGs can be challenging due to various evolutionary events, such as gene duplication, loss, or horizontal gene transfer (HGT). There are several programs to infer the available orthology methods. In this chapter, three different graph-based programs are used to predict OGs in our dataset: OrthoMCL (which uses Markov chains for clustering), the COG tool (which uses the BeT method for clustering), and OMA standalone (which uses cliques and hierarchical clustering methods). Additionally,

Table 1
Information on the proteomes used for presenting orthologous resources

Label	Species	Kingdom	Phyla	Class	Biosample accession
Bfragilis	*Bacteroides fragilis*	Bacteria	Bacteroidetes	Bacteroidia	SAMN16357367
Ecoli	*Escherichia coli*	Bacteria	Proteobacteria	Gammaproteobacteria	SAMN02604091
Hinfluenzae	*Haemophilus influenzae*	Bacteria	Proteobacteria	Gammaproteobacteria	SAMN02595602
Kgyiorum	*Kerstersia gyiorum*	Bacteria	Proteobacteria	Betaproteobacteria	SAMN10273024
Lacidophilus	*Lactobacillus acidophilus*	Bacteria	Firmicutes	Bacilli	SAMN02603216
Mgenitalium	*Mycoplasma genitalium*	Bacteria	Firmicutes	Tenericutes	SAMN02603983
Mjannaschii	*Methanococcus jannaschii*	Archea	Euryarchaeota	Methanococci	SAMN02603984
Mtuberculosis	*Mycobacterium tuberculosis*	Bacteria	Actinobacteria	Actinobacteridae	SAMEA3138326
Rradiobacter	*Rhizobium radiobacter*	Bacteria	Proteobacteria	Alphaproteobacteria	SAMN10169604
Synechocystis	*Synechocystis* sp.	Bacteria	Cyanobacteria	Cyanophyceae	SAMD00061113

OrthoFinder2, which combines graph-based and tree-based methods, is also used.

5.1 OrthoMCL

OrthoMCL [21] is a program for determining OGs in a given set of genomes. In addition to the original reference, a summary explanation of its method can be found in [22]. OrthoMCL distinguishes between orthologs and paralogs. Two genes from the same species will be included in the same OG only if their reciprocal BLAST similarity is larger than the similarity to any gene not belonging to that species. Those genes are considered "recent in-paralogs."

The standalone version of OrthoMCL can be downloaded from the website https://orthomcl.org/orthomcl/app/downloads/software/ and installed locally. However, it is not a simple program to run, because its pipeline requires the installation of other programs (BLAST, MySQL, MCL). In this chapter, we run OrthoMCL through the convenient Get_Homologues package [13], which offers three clustering methods, including OrthoMCL.

In our example, Get_Homologues is executed in a directory (which we named prok_proteomes) that contains the full proteome

Table 2
List of bioinformatics tools used in this chapter

Tool	Description	References	Source
eggNOG-mapper	Orthologous inference and functional annotation	[10]	https://github.com/eggnogdb/eggnog-mapper
Egg-NOG v6	Orthology resource	[11]	http://eggnog6.embl.de/
GenBank	Genome database	[12]	https://www.ncbi.nlm.nih.gov/genbank/
get_homologues	Orthologous inference	[13]	https://github.com/eead-csic-compbio/get_homologues
ggtree	Phylogenetic visualization	[14]	https://github.com/YuLab-SMU/ggtree
IQ-TREE2	Phylogenetic inference	[15]	https://github.com/iqtree/iqtree2
MAFFT	Sequence alignment	[16]	https://mafft.cbrc.jp/alignment/software/
OMA	Orthologous inference	[17]	https://github.com/DessimozLab/OmaStandalone
OrthoDB	Orthology resource	[18]	https://www.orthodb.org/
OrthoFinder2	Orthologous inference	[9]	https://github.com/davidemms/OrthoFinder
seqkit	Sequence concatenation	[19]	https://bioinf.shenwei.me/seqkit/
trimAL	Sequence trimming	[20]	https://github.com/inab/trimal

sequences of the 10 prokaryote genomes previously downloaded from Genbank (Table 1).

Bash Shell

```
 $ get_homologues.pl -d ./prok_proteomes/ -M -C
30 -t 2
```

The flag "–M" specifies the use of OrthoMCL to compute orthologs. An output folder named "prok_proteomes_homologues" is created after the running is finished; it includes a subfolder that contains separate FASTA files for each OG. The option "–C" specifies the minimum alignment coverage percentage in pairwise BLAST; we set it at 30%. This value is considered low, but it is appropriate for distantly related prokaryotes, as is the case in our example. The parameter "–t" specifies the minimum number of sequences in each OG. Using the value 2, we exclude OGs formed by just one member (also known as singletons).

Another way of running Get_Homologues is by using option "–e." This option only computes OGs that do not contain in-paralogs or contain only single-copy genes from each taxon. The resulting OGs are called single-copy OGs, or SCOGs:

```
 $ get_homologues.pl -d ./prok_proteomes/ -M -C
30 -e
```

SCOGs are important as inputs for species phylogeny reconstruction, because their evolutionary history, in principle, does not include duplications.

The results of running Get_Homologues with the two commands above are shown in Table 3.

5.2 Cluster of Orthologous Groups

COGs stands for "cluster of orthologous groups," which is the name used by its authors [23] for OGs. The algorithm for determining COGs is called COGtriangles and is briefly explained in [22]. The algorithm has a preprocessing step in which all in-paralogs are determined and converted to single vertices. The COGtriangles algorithm was refined in terms of computational complexity by the EdgeSearch algorithm [24]. EdgeSearch is also one of the options of the Get_Homologues package [13].

The COGtriangles in the Get_Homologues package are run using the option "–G" shown as follows.

Bash Shell

```
 $ get_homologues.pl -d ./prok_proteomes_fasta -G
-C 30 -t 2
```

An output subfolder is created inside the "prok_proteomes_homologues" folder, containing separate FASTA files for each OG. Table 3 shows the number of OGs predicted by this program.

Get_Homologues can compare the output of the different methods implemented by the software. The script `compare_clusters.pl` identifies the common elements between the sets of clusters generated by OrthoMCL and COG tools. The command is shown below.

Bash Shell

```
 $ compare_clusters.pl -o sample_intersection -d
prok_proteomes_homologues/*_algOMCL_*,prok_pro-
teomes_homologues/*_algCOG_*
```

Table 3
The number of orthologous groups predicted by four different programs among the proteomes of ten prokaryotic species

	OrthoMCL	COG tool	OrthoFinder2	OMA
All OGs (containing two or more homologous genes)	3492	3121	4205	4389
Single-copy OGs (SCOGs) in all ten species	53	56	53	17

A folder called `sample_intersection` is created containing the list of OGs predicted by both methods. A comparison of the results of the two methods is shown in Fig. 1 using Venn diagrams. It is clear from the diagrams that there are significant differences in the results, especially in the case of non-single-copy-OGs. Which method is more reliable? We comment on this issue in Subheading 5.5.

5.3 OrthoFinder2

OrthoFinder2 [9] is a hybrid method that combines graph and tree-based approaches. In the first step, OrthoFinder2 incorporates BLAST score normalization to predict highly accurate OGs by taking into account gene length bias [25]. In the second step, these OGs are used to infer unrooted gene trees with dendro-BLAST; after that, these gene trees are used for the identification of gene duplication events. Ultimately, all of the phylogenetic data obtained is used to determine the complete set of orthologs between all species. OrthoFinder2 achieves very high ortholog inference accuracy on the Quest for Orthologs benchmarks [9, 26] (*see* Subheading 5.5).

Running OrthoFinder2 is a straightforward process that only requires a collection of proteomes (one for each species) in FASTA format. The command is shown below.

Bash Shell

```
$ orthofinder -f ./prok_proteomes_fasta
```

OrthoFinder2 generates a folder named “OrthoFinder” to store the output. The file `Statistics_Overall.tsv`, which contains the number of genes assigned and the number of OGs, is located within the subfolder Comparative_Genomics_Statistics. Table 3 summarizes the results.

5.4 OMA

Orthologous matrix (OMA) is a graph-based software that provides three different types of orthologs: pairwise orthologs, OMA groups, and hierarchical orthologous groups (HOGs), each type based on a different inference method [17, 27]. The OMA tool

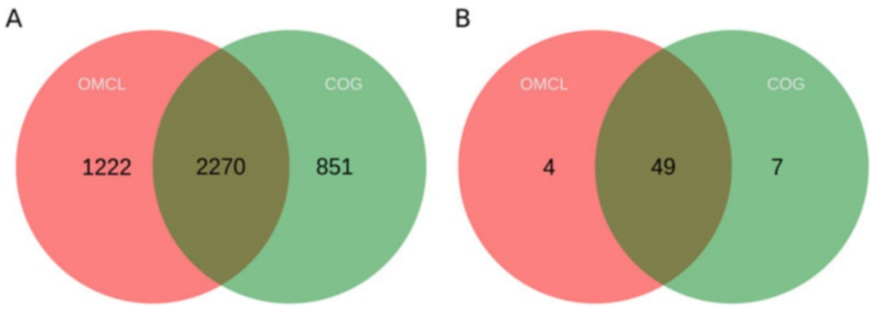


Fig. 1 Venn diagram depicting the overlap of orthologous groups predicted by OrthoMCL and COG tools in (**a**) OGs containing at least two sequences and (**b**) single-copy OGs containing genes in all ten species

begins by inferring pairwise orthologs, which are subsequently used to construct both HOGs and OMA Groups. To infer OGs, it first computes all-against-all Smith-Waterman alignments, saving only candidate pairs with sufficient score and overlap. OMA groups are clustered by identifying well-connected subgraphs ("cliques") corresponding to sequences that are strictly orthologous, excluding protein sequences involved in one-to-many and many-to-many relations.

The other clustering strategy performed by OMA software is hierarchical clustering. Hierarchical orthologous groups (HOGs) are formed starting with the most specific taxonomic level and merging groups progressively towards the root of the species tree [28]. These HOGs may include both orthologs and in-paralogs.

The OMA program is available as open-source software, OMA Standalone (https://omabrowser.org/standalone/#downloads), which is compatible with the public OMA browser (https://omabrowser.org).

To run OMA, all proteomes (.faa files) should be located in a folder called DB. OMA can be run by executing the following three commands.

Bash Shell

```
# Use option -p to create a default parameters file
$ oma -p
# Then use option -s to run the first part of the
OMA pipeline
$ oma -s
# Run the second part
$ oma
```

All resulting files will be located in a folder called "output." The *OrthologousGroupsFasta* subfolder will contain all OGs; these are one-to-one strict orthologs. Table 3 shows the OGs predicted by OMA in our dataset.

5.5 Performance of Orthology Inference Methods

Given that there are multiple orthology inference methods, it is natural to ask which one is the most accurate. This is a difficult question to answer. In what follows, we provide a simple comparison of the results we presented above, and then briefly describe a multi-group initiative to provide benchmarks for assessing the quality of orthology inference methods.

The Quest For Orthologs (QFO) consortium [29] offers an online benchmarking tool for orthology prediction (http://orthology.benchmarkservice.org). This benchmarking service has established a reference proteome dataset to enable comparisons of individual inference methods on a consistent set of species and proteins. The latest version of the database comprises 78 Uniprot reference proteomes from model organisms and species of biomedical interest (48 eukaryotes, 23 bacteria, and 7 archaea).

The benchmark service assesses the quality of predictions based on a species tree discordance test, the agreement with respect to reference phylogenies/orthologs, and a functional test. Each test measures the performance in terms of precision and recall. In simple terms, Precision measures the number of false positives, whereas recall measures the number of false negatives. A false positive is a gene that was wrongly considered as belonging to family *X*. A false negative is a gene that should have been assigned to family *X* but was not. Additional details can be found in [29].

In our dataset, the number of all orthologs (containing two or more species) predicted by OMA (4389 clusters) is higher than that predicted by OrthoMCL (3492) and COG (3121), but similar to OrthoFinder2 (4205). However, when we limit our observation to single-copy OGs, the pattern changes. Even though OMA yielded the highest number of orthologs, it inferred the fewest number of single-copy OGs in all ten species (17 SCOGs), whereas OrthoFinder2, OrthoMCL, and COG predicted 53, 53, and 56 SCOGs, respectively.

These results are consistent with previous studies. For example, Altenhoff et al. [17] compared several orthology methods on a reconstruction of the Lophotrochozoa phylogeny where OMA generally predicted more OGs than OrthoMCL and OrthoFinder2, but produced a smaller quantity of larger groups. This difference in group size distribution is likely the result of different trade-offs in terms of precision (proportion of predicted orthologs that are correct) and recall (proportion of true orthologs that are correctly predicted). A recent benchmarking study comparing 20 different orthology prediction methods highlighted the trade-off between precision and recall [26]. The OMA Groups method ranks as one of the best in precision but is worse in recall than most other methods. This means that genes are often missing from

OMA OGs, or OGs are more fragmented than they should be. OrthoMCL has good recall, though with lower accuracy than other methods. OrthoFinder2, Domanoid [30], and OrthoInspector [31] show a good balance between precision and recall in this benchmark (https://orthology.benchmarkservice.org/).

In terms of runtime, OMA is by far the most costly of the orthology methods tested, due to its reliance on full Smith–Waterman alignments and evolutionary distance in the all-against-all phase.

6 Obtaining Phylogenetic Trees from Orthogroups

Species phylogeny reconstruction is one of the main goals of orthology inference and can be performed using information from multiple genes (orthogroups) concatenated in a single matrix. For this purpose, it is important to consider only SCOGs because, as already mentioned, duplicated genes tend to bring noise to the desired vertical phylogenetic signal. There is usually no guarantee that SCOGs have not been subject to HGT; but determining whether a given OG was subject to HGT is a much more laborious effort than simply ensuring that an OG does not contain paralogs.

OMA groups are suitable for phylogenetic analyses since OMA generates OGs that contain only one copy per species, avoiding in-paralogs. OrthoMCL and OrthoFinder2 frequently include orthogroups with the presence of in-paralogs. OrthoFinder2 generates a separate directory with OGs for which only one of the copies is selected for the OG if recent in-paralogs have been inferred for that orthogroup. Note however that an OG that has been curated such that only one of the copies of paralog genes is kept is not really a single-copy OG. Nevertheless, if the paralog copy that is kept is indeed an ortholog with respect to the other genes in the OG, then that OG can be used for species phylogeny inference.

In this subsection, we compare the phylogenies produced from the sequence alignment of SCOGs predicted by the OrthoFinder2 and OMA methods.

6.1 OrthoFinder2

We use the 53 SCOG sequences predicted by OrthoFinder2 to infer a phylogenetic tree. After running OrthoFinder2 (Subheading 5.3), the SCOGs sequences are located in the folder OrthoFinder/Single_Copy_Orthologue_Sequences.

First, we navigate into the result folder.

Bash Shell

```
$ cd OrthoFinder/Single_Copy_Orthologue_Sequences
```

We rename the FASTA headers in all OG files using the column label in Table 1. As all protein sequences are sorted alphabetically, we use the following code:

```
$ for i in *.fa; do awk 'NR==FNR{names[NR]=$0;
next} /^>/{$1=">"names[++c]}1' label.tab $i > {i%.
*}_rn.fa; done
```

Then, we align all protein sequences in OGs using MAFFT as follows:

```
$ for i in *_rn.fa; do mafft --maxiterate 1000 --
localpair $i > ${i%%.*}_alig.fa; done
```

Poorly aligned regions are removed using trimAL as follows:

```
$ for i in *_alig.fa; do trimal -automated1 -in $i
-out ${i%%.*}_trim.fa; done
```

A supermatrix is then constructed by concatenating all alignments using the "concat" option of seqkit tool:

```
$ seqkit concat *_trim.fa > concat_orthofinder.fa
```

The resulting concat.fa FASTA file is used to construct the phylogenomic tree with IQTREE2 software using a maximum-likelihood approach:

```
# use LG as the substitution model (-m) and define
1000 as the number of bootstrap replicates (-bb)
$ iqtree2 -s concat_orthofinder.fa -m LG -bb 1000
-o "Mjannaschii"
```

The tree inferred can be visualized (Fig. 2a) with *ggtree* package using the following commands:

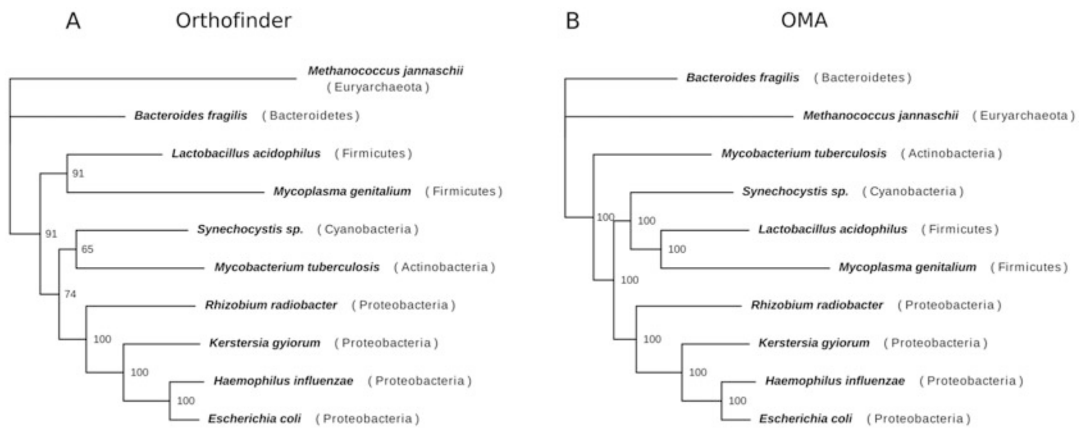

A
Orthofinder
Methanococcus jannaschii (Euryarchaeota)
Bacteroides fragilis (Bacteroidetes)
Lactobacillus acidophilus (Firmicutes)
Mycoplasma genitalium (Firmicutes)
Synechocystis sp. (Cyanobacteria)
Mycobacterium tuberculosis (Actinobacteria)
Rhizobium radiobacter (Proteobacteria)
Kerstersia gyiorum (Proteobacteria)
Haemophilus influenzae (Proteobacteria)
Escherichia coli (Proteobacteria)
91
91
65
74
100
100
100
B
OMA
Bacteroides fragilis (Bacteroidetes)
Methanococcus jannaschii (Euryarchaeota)
Mycobacterium tuberculosis (Actinobacteria)
Synechocystis sp. (Cyanobacteria)
Lactobacillus acidophilus (Firmicutes)
Mycoplasma genitalium (Firmicutes)
Rhizobium radiobacter (Proteobacteria)
Kerstersia gyiorum (Proteobacteria)
Haemophilus influenzae (Proteobacteria)
Escherichia coli (Proteobacteria)
100
100
100
100
100
100
100

Fig. 2 Phylogenetic tree constructed using maximum likelihood program based on the alignment of (**a**) 53 SCOGs predicted by OrthoFinder2 and (**b**) 17 SCOGs predicted by OMA. The tip labels represent the scientific name of the species and the respective phylum is shown between parentheses. Bootstrap values are represented by the number of nodes

R Script

```
library(ggtree)

# read tree file

tree <- read.tree("concat_orthofinder.fa.treefile")

# read metadata (table 1)

metadata <- read.table("metadata.tab", sep = "\t", header = T)

# Draw the tree displaying bootstrap value

gg <- ggtree(tree, layout = "rectangular",right = T)  +

  geom_nodelab(aes(label=label), size=2, hjust = -0.3)

# add tip labels

p1 <- gg %<+% metadata +

  geom_tiplab(aes(label=paste("bolditalic('", species,

"')~","'('~", Fila,"~')'")), parse=T,size=2.2, hjust = -0.02)

p1
```

6.2 OMA

In order to infer the phylogenetic tree of the ten species using OMA, first we need to extract all 17 SCOGs predicted by it (Subheading 5.4).

Bash Shell

```
# We navigate into the result folder

$ cd Output/OrthologousGroupsFasta

# Create a directory to copy all 17 SCOG to there

$ mkdir OG_10species

# copy all OGs that contain 10 protein sequences into the “OG_10species” directory

$ for i in *.fa; do occurrences=$(grep -c ">" "$i"); if ((
occurrences == 10 )); then cp "$i" OG_10species; fi; done
```

We then navigate to the folder “OG_10species.” From there, the commands are basically the same as in Subheading 6.1 (OrthoFinder2), which we give in full for convenience.

Bash Shell

```
$ cd OG_10species

$ for i in *.fa; do mafft --maxiterate 1000 --localpair $i >
${i%%.*}_alig.fa; done

$ for i in *_alig.fa; do trimal -automated1 -in $i -out
${i%%.*}_trim.fa; done

$ seqkit concat *_trim.fa > concat_OMA.fa

$ iqtree2 -s concat_OMA.fa -m LG -bb 1000 -o “Mjannaschii”
```

R Script

```
library(ggtree)

# Read the treefile produced by IQTREE2

tree <- read.tree("cancat_OMA.fa.treefile")

# Read metadata (table 1)

metadata <- read.table("metadata.tab", sep = "\t", header = T)

# Draw the tree

gg <- ggtree(tree, layout = "rectangular", right = T)  +

  geom_nodelab(aes(label=label), size=2,hjust = -0.3)

# add tip labels

p2 <- gg %<+% metadata +

  geom_tiplab(aes(label=paste("bolditalic('", species,

"')~","'('~", Fila,"~')'")), parse=T,size=2.2, hjust = -0.02)

p2
```

The tree obtained from the OMA orthologs is shown in Fig. 2b.

The phylogenetic tree generated using 17 OGs predicted by OMA shows a topology similar to that constructed using 53 OGs predicted by OrthoFinder2. The innermost clade contains *E. coli* and *H. influenzae*. Since both belong to the class Gammaproteobacteria, this was expected. *K. gyiorum* (betaproteobacteria) and *R. radiobacter* (alphaproteobacteria) are also more closely related as they belong to the phylum proteobacteria. Likewise, the two species of the phylum Firmicutes (*Mycoplasma genitalium* and *L. acidophilus*) group together. On the other hand, while *Synechocystis* sp. (cyanobacteria) and Firmicutes form a well-supported cluster species in the OMA-based tree, in the OrthoFinder2 tree, we observe that *Synechocystis* sp. and *Mycobacterium tuberculosis* (Actinobacteria) cluster together, although with a low bootstrap support. Bootstrap values were used to estimate the branch support of the phylogenetic tree using 1000 replicates. Bootstrapping is a computational strategy to measure how strongly the sequence data supports the phylogeny. The closer the bootstrap value is to 100, the better supported is the node or branch.

7 eggNOG-Mapper

eggNOG-mapper [10] is a gene function annotation tool. It utilizes the vast eggNOG database [11] of orthologous groups, which spans thousands of bacterial, archaeal, and eukaryotic organisms. (This database is covered in more detail in Subheading 8; eggNOG is an acronym for evolutionary gene genealogy Non-supervised Orthologous Groups.) To achieve this, the tool uses precomputed phylogenies for each OG to enhance orthology assignments, making it possible to annotate a gene by transferring annotations from close orthologs. eggNOG-Mapper is able to differentiate between orthologous and paralogous gene groups, which is important for functional assignment purposes [2].

eggNOG-mapper was specifically designed to annotate vast collections of sequences, primarily focusing on protein-coding genes from genomes, metagenomes, and transcriptomes. The functional attributes assigned to query sequences consist of COG functional categories [32], gene ontology terms [33], KEGG pathways [34], and carbohydrate-active enzymes (CAZymes) terms [35]. eggNOG-mapper is available online (http://eggnog-mapper.embl.de/) and can also be used in a standalone version (https://github.com/eggnogdb/eggnog-mapper).

Three options for the initial sequence-mapping step are available: Diamond [36], MMseqs2 [37], and HMMER3 [38]. Diamond is the default mode and the best, considering memory and speed. Compared to the other two modes, HMMER3 mode is slower and requires the download of large databases. Nevertheless, utilizing HMM-based searches may be helpful in detecting distant homology relationships.

We use eggNOG-mapper to annotate single-copy OGs obtained as described above (Subheadings 5.1, 5.2, 5.3 and 5.4).

Bash Shell

```
# The following code is an example of OMA output.
# Navigate into the directory containing 17 OGs in all ten
species:
$ cd OG_10species
# Merge all sequences in one file:
$ cat *.faa > OGs_OMA.faa
# Run eggNOG mapper:
$ emapper.py -i OGs_OMA.faa -o eggnog_OMA
```

eggNOG-mapper by default will run Diamond BLASTP. The option "–m" changes the search program to MMseqs2 (–m mmseqs) or HMMER (–m hmmer).

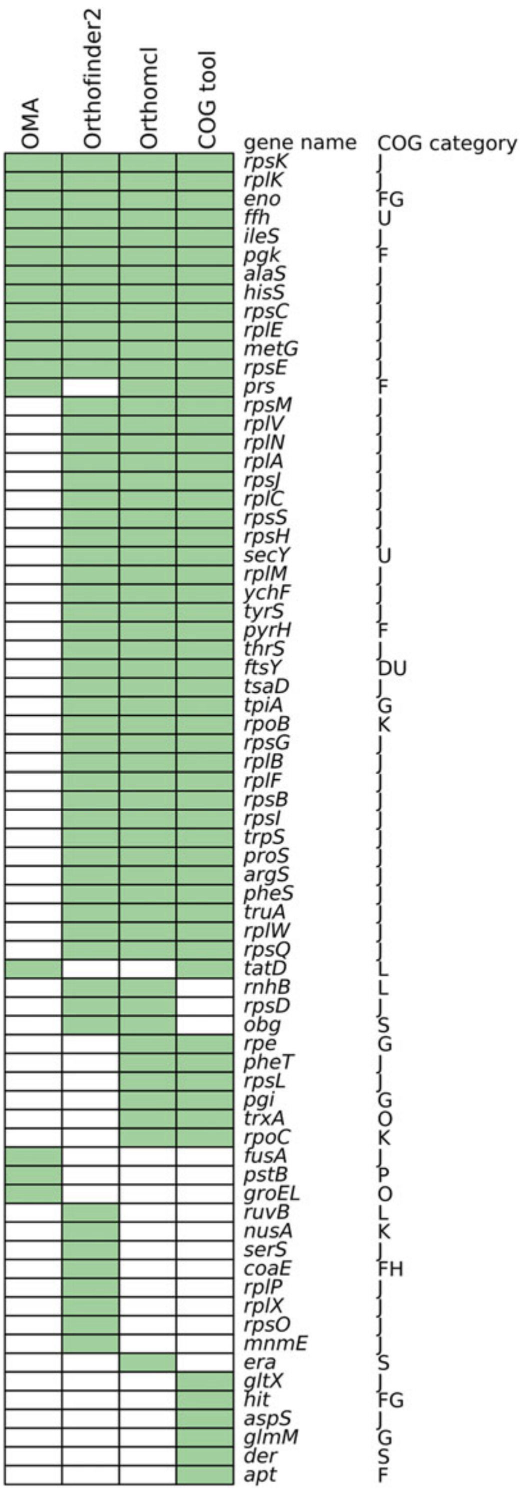
The output consists of three files. The most important is the annotation file (outputname.emapper.annotations), which provides the functional predictions for each query (COG category, KEGG pathway, OG terms, and CAZy terms) in TSV format. A comparative list of predicted single-copy OGs is shown in Fig. 3.

A total of 71 different OGs were predicted: 12 OGs were predicted by all four methods and 53 OGs were predicted by at least two methods. Most OGs were assigned to the COG category of “translation, ribosomal structure, and biogenesis” (J).

For better visualization, we plot the single-copy orthogroups predicted by each of the four orthology inference methods as a function of the COG category (Fig. 4).

R Script

```
library(ggplot2)

library(dplyr)

library(tidyverse)

#read the tsv output file from eggNOG-mapper after removing all

comment lines

oma<- read.table("mapper_OMA.tab", header = T, sep = "\t")

oma_g <- as.data.frame(table(oma$COG_category))

of<- read.table("mapper_of.tab", header = T, sep = "\t")

of_g <- as.data.frame(table(of$COG_category))

omcl <- read.table("mapper_OMCL.tab", header = T, sep = "\t")

omcl_g <- as.data.frame(table(omcl$COG_category))

cog <- read.table("mapper_COG.tab", header = T, sep = "\t")

cog_g <- as.data.frame(table(cog$COG_category))

# combine all outputs in one table

combined <- rbind(oma_g,of_g, omcl_g, cog_g)
# Add a new column with the method name

combined <- transform(combined, method =

rep(c("OMA","Orthofinder", "OrthoMCL", "COG"), times =

c(nrow(oma_g),nrow(of_g), nrow(omcl_g), nrow(cog_g))))

a <- combined %>% group_by(method) %>%

    mutate(freq = Freq*100/sum(Freq))

# Set categories position on x axis

positions <- c("J", "K","L","F","G","P","FG","FH","O","U",
"DU")

# Draw the barplot

ggplot(a, aes(fill=method, y=freq, x=Var1)) +

geom_bar(position = position_dodge2(width = 1.4, preserve =

"single"), stat="identity", colour="black", size=0.2)+

  scale_x_discrete(limits = positions) +

  xlab("COG category") + ylab("% of proteins")+

  theme_bw()+

  scale_fill_brewer(palette = "Set3")
```



OMA	Orthofinder2	Orthomcl	COG tool	gene name	COG category
🟩	🟩	🟩	🟩	*rpsK*	J
🟩	🟩	🟩	🟩	*rplK*	J
🟩	🟩	🟩	🟩	*eno*	FG
🟩	🟩	🟩	🟩	*ffh*	U
🟩	🟩	🟩	🟩	*ileS*	J
🟩	🟩	🟩	🟩	*pgk*	F
🟩	🟩	🟩	🟩	*alaS*	J
🟩	🟩	🟩	🟩	*hisS*	J
🟩	🟩	🟩	🟩	*rpsC*	J
🟩	🟩	🟩	🟩	*rplE*	J
🟩	🟩	🟩	🟩	*metG*	J
🟩	🟩	🟩	🟩	*rpsE*	J
🟩		🟩	🟩	*prs*	F
	🟩	🟩	🟩	*rpsM*	J
	🟩	🟩	🟩	*rplV*	J
	🟩	🟩	🟩	*rplN*	J
	🟩	🟩	🟩	*rplA*	J
	🟩	🟩	🟩	*rpsJ*	J
	🟩	🟩	🟩	*rplC*	J
	🟩	🟩	🟩	*rpsS*	J
	🟩	🟩	🟩	*rpsH*	J
	🟩	🟩	🟩	*secY*	U
	🟩	🟩	🟩	*rplM*	J
	🟩	🟩	🟩	*ychF*	J
	🟩	🟩	🟩	*tyrS*	J
	🟩	🟩	🟩	*pyrH*	F
	🟩	🟩	🟩	*thrS*	J
	🟩	🟩	🟩	*ftsY*	DU
	🟩	🟩	🟩	*tsaD*	J
	🟩	🟩	🟩	*tpiA*	G
	🟩	🟩	🟩	*rpoB*	K
	🟩	🟩	🟩	*rpsG*	J
	🟩	🟩	🟩	*rplB*	J
	🟩	🟩	🟩	*rplF*	J
	🟩	🟩	🟩	*rpsB*	J
	🟩	🟩	🟩	*rpsI*	J
	🟩	🟩	🟩	*trpS*	J
	🟩	🟩	🟩	*proS*	J
	🟩	🟩	🟩	*argS*	J
	🟩	🟩	🟩	*pheS*	J
	🟩	🟩	🟩	*truA*	J
	🟩	🟩	🟩	*rplW*	J
	🟩	🟩	🟩	*rpsQ*	J
🟩			🟩	*tatD*	L
	🟩	🟩		*rnhB*	L
	🟩	🟩		*rpsD*	J
	🟩	🟩		*obg*	S
		🟩	🟩	*rpe*	G
		🟩	🟩	*pheT*	J
		🟩	🟩	*rpsL*	J
		🟩	🟩	*pgi*	G
		🟩	🟩	*trxA*	O
		🟩	🟩	*rpoC*	K
🟩				*fusA*	J
🟩				*pstB*	P
🟩				*groEL*	O
	🟩			*ruvB*	L
	🟩			*nusA*	K
	🟩			*serS*	J
	🟩			*coaE*	FH
	🟩			*rplP*	J
	🟩			*rplX*	J
	🟩			*rpsO*	J
	🟩			*mnmE*	J
		🟩		*era*	S
			🟩	*gltX*	J
			🟩	*hit*	FG
			🟩	*aspS*	J
			🟩	*glmM*	G
			🟩	*der*	S
			🟩	*apt*	F

Fig. 3 List of SCOGs predicted by OMA, OrthoFinder2, OrthoMCL, and COGtriangle programs including annotation using COGs and KEGG orthology (KO), inferred by eggNOG-mapper. Some SCOGs were annotated into two different COG categories

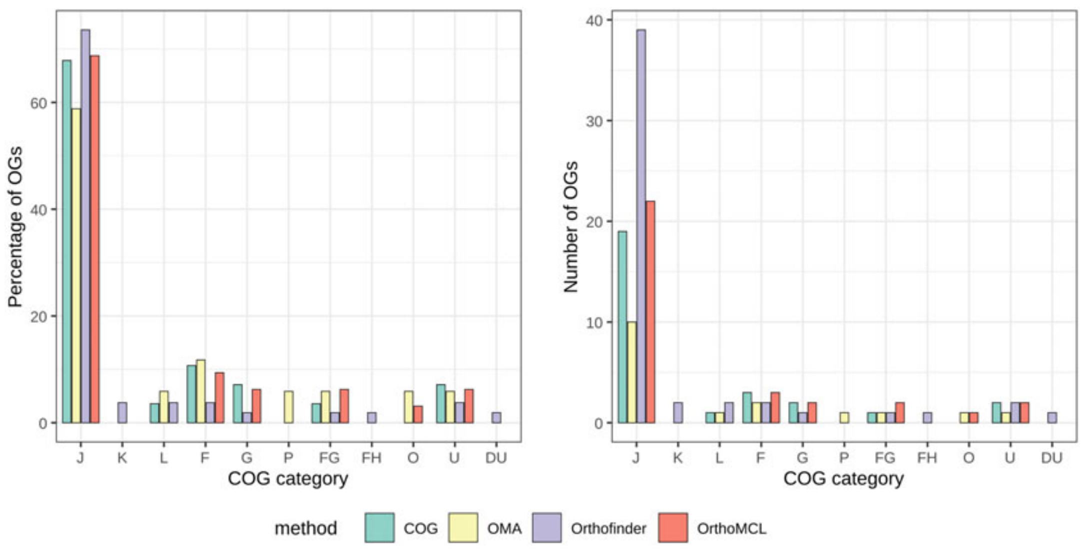


Fig. 4 Barplot comparing the proportion of proteins among different COG categories annotated by eggNOG-mapper with four different orthology inference methods. Data are shown using percentage (left panel) and absolute values (right panel)

We can see in Fig. 4 that most proteins (between 60% and 70%) in OGs were annotated with COG Category J (translation, ribosomal structure, and modification), encoding for ribosomal proteins and other components of the translation machinery, and this was consistent for all four orthology methods (Fig. 4). This is expected, because gene conservation across phylogenetically diverse bacteria and archaea (as is the case for our group of ten species) is known to occur only for genes that are related to fundamental life processes, such as the ones included in category J [23, 39]. All methods predicted OGs related to replication functions (COG Category L), with the exception of OrthoMCL. OrthoFinder2 was the only one that predicted OGs with proteins associated with transcriptional functions (COG Category K: Transcription). Although studies analyzing genomes of bacteria and archaea differ in the number and groups of predicted universal single-copy genes [40–42], these studies report that the vast majority of OGs are annotated within COG Category J and fewer than a handful are annotated as part of categories L, G, H, or O [43], consistent with what we observed here.

8 Obtaining Orthologs from Ortholog Databases

In this section, we briefly describe a few resources that offer pre-computed OGs. The text that follows complements and updates the article by Setubal and Stadler [44] (*see* Subheading 3.2 in that article).

Orthology databases vary in the quantity and diversity of the species they represent. Some databases, like eggNOG [45] and OrthoDB [46], cover a wide range of species, including viral sequences, while others, such as TreeFam [47], are more specialized to certain clades.

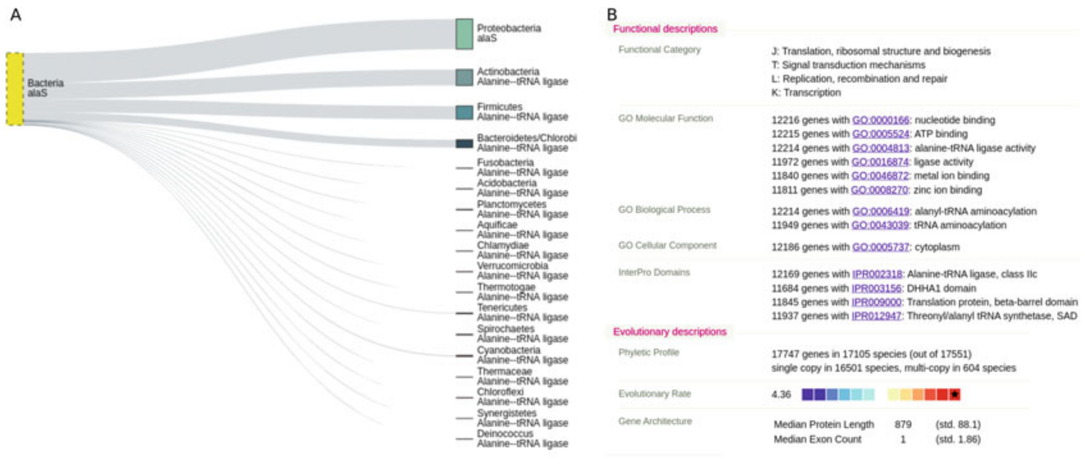
Orthology resources offer diverse ways of information exploration via web interfaces for manual inspections essential for routine use by non-experts or using programmatical access interfaces. The search can be performed using the gene name (e.g., *alaS*), GeneID (e.g., 948,665), or the protein sequence. Here, we will use the protein AlaS that was predicted as a single-copy orthologous group present in all ten species of our dataset to test two orthology resources: OrthoDB v.11 and eggNOG v.6.

According to the UniProtKB database (entry: P00957), AlaS is an alanyl tRNA ligase that catalyzes the attachment of L-alanine to tRNA (Ala). The Gene ontology annotation for this protein is aminoacyl-tRNA ligase (molecular function, GO:0004812) and is involved in the protein biosynthesis process (biological process, GO:0006412).

8.1 OrthoDB

The OrthoDB database contains pre-computed orthology data at various levels of taxonomic distance. The most recent version (OrthoDB v.11) provides analysis and annotation of over 100 million genes, sampling a broad range of species diversity, including prokaryotes (18,158 genomes), eukaryotes (1973 genomes), and viruses (7962 genomes). OrthoDB is based on the OrthoLoger software (https://orthologer.ezlab.org). It relies on the bidirectional best hit method, calculated using MMseqs2 [37]. Orthologous genes are clustered following a hierarchical approach guided by a user-provided organism taxonomy. A distinctive feature of OrthoDB is that it also provides evolutionary information for OGs, such as the rate of sequence divergence.

Users can search OrthoDB (https://www.orthodb.org/) by conducting a basic text search, utilizing identifiers from diverse databases, or inputting a protein sequence. When using the term "alaS," the database yields 612 groups, which correspond to results across various taxonomic levels. At the bacterial level, the alaS group contains 17,747 genes in 17,105 species (out of 17,551 species in the database), which indicates that in-paralogs are included in OGs. The group hierarchy splits OGs into different sub-taxonomic levels, which is displayed with a Sankey flow diagram (Fig. 5a). Single-copy genes were identified in 16,501 bacterial species and multi-copy genes in 604 species. We navigate into each of the nine bacterial species of our dataset and confirm the single-copy presence of AlaS in the OrthoDB database. In the search at the archaea level, OrthoDB returns the presence of


A
Bacteria alaS
Proteobacteria alaS
Actinobacteria Alanine--tRNA ligase
Firmicutes Alanine--tRNA ligase
Bacteroidetes/Chlorobi Alanine--tRNA ligase
Fusobacteria Alanine--tRNA ligase
Acidobacteria Alanine--tRNA ligase
Planctomycetes Alanine--tRNA ligase
Aquificae Alanine--tRNA ligase
Chlamydiae Alanine--tRNA ligase
Verrucomicrobia Alanine--tRNA ligase
Thermotogae Alanine--tRNA ligase
Tenericutes Alanine--tRNA ligase
Spirochaetes Alanine--tRNA ligase
Cyanobacteria Alanine--tRNA ligase
Thermaceae Alanine--tRNA ligase
Chloroflexi Alanine--tRNA ligase
Synergistetes Alanine--tRNA ligase
Deinococcus Alanine--tRNA ligase
B

Functional descriptions

Functional Category	J: Translation, ribosomal structure and biogenesis T: Signal transduction mechanisms L: Replication, recombination and repair K: Transcription
GO Molecular Function	12216 genes with GO:0000166: nucleotide binding 12215 genes with GO:0005524: ATP binding 12214 genes with GO:0004813: alanine-tRNA ligase activity 11972 genes with GO:0016874: ligase activity 11840 genes with GO:0046872: metal ion binding 11811 genes with GO:0008270: zinc ion binding
GO Biological Process	12214 genes with GO:0006419: alanyl-tRNA aminoacylation 11949 genes with GO:0043039: tRNA aminoacylation
GO Cellular Component	12186 genes with GO:0005737: cytoplasm
InterPro Domains	12169 genes with IPR002318: Alanine-tRNA ligase, class IIc 11684 genes with IPR003156: DHHA1 domain 11845 genes with IPR009000: Translation protein, beta-barrel domain 11937 genes with IPR012947: Threonyl/alanyl tRNA synthetase, SAD

Evolutionary descriptions

Phyletic Profile	17747 genes in 17105 species (out of 17551) single copy in 16501 species, multi-copy in 604 species
Evolutionary Rate	4.36
Gene Architecture	Median Protein Length 879 (std. 88.1) Median Exon Count 1 (std. 1.86)

Fig. 5 AlaS family at Bacteria level in OrthoDB. (**a**) Interactive group hierarchy split into different taxonomic groups with a Sankey flow diagram. (**b**) Functional annotations of the orthologous group, including COG categories, GO terms, InterPro domains, and evolutionary information. Both figures are screenshots of the search results page for AlaS protein at the OrthoDB website (https:/www.orthodb.org/) [18]

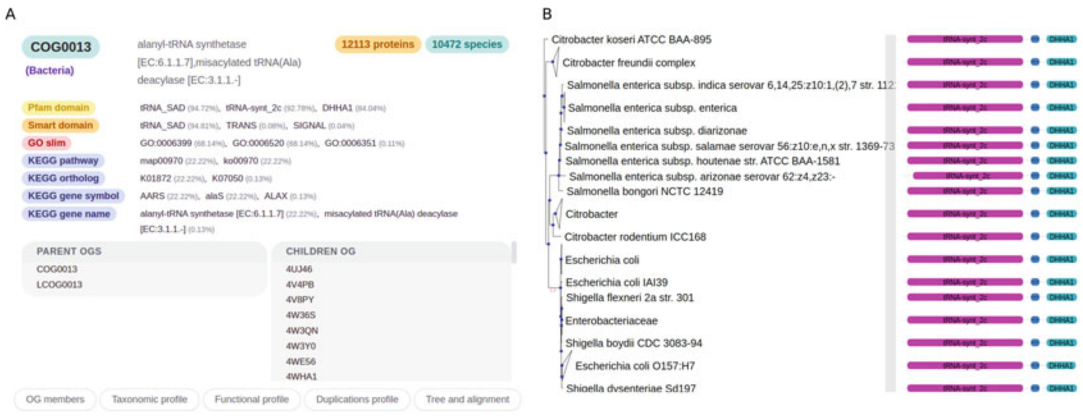
1592 genes in 602 species (out of 607 species in the database), and just 52 species contain single-copy *alaS* gene, including *Methanococcus jannaschii*.

OrthoDB also provides a functional description of the protein family including COG categories, Gene Ontology terms, and InterPro domains, as well as evolutionary descriptions including the number of copies per organism, evolutionary rate, and gene architecture (Fig. 5b).

8.2 eggNOG

The eggNOG database offers comprehensive functional information and orthology data for organisms across all domains of life. eggNOG computes all-against-all Smith–Waterman alignments carried out by the SIMAP project [48], and best hits are stored and indexed in a relational database. The current database version (version 6) contains information on 12,535 organisms and 17 million orthologous groups [11].

The user has to first enter a search term (e.g., AlaS or alanyl-tRNA ligase). The OGs matching the query are shown as a list of summary cards, where the most important functional annotations are displayed. Select the card in which the taxonomic level (e.g., bacteria) of the orthologous group should be searched. The orthogroup (COG0013) of *alaS* gene provided by eggNOGG contains 12,113 proteins in 10,472 bacterial species, which indicates that in-paralogs are included in the OG (Fig. 6a). The result includes functional annotation, such as KEGG, Gene Ontology, PFAM, and SMART, the list of species with multiple copies of the gene (duplication profile), and interactive visualization of the taxonomic distribution of OGs and phylogenetic tree (Fig. 6b).


A

COG0013 (Bacteria)

alanyl-tRNA synthetase [EC:6.1.1.7],misacylated tRNA(Ala) deacylase [EC:3.1.1.-]

12113 proteins 10472 species

Pfam domain	tRNA_SAD (94.72%), tRNA-synt_2c (92.78%), DHHA1 (84.04%)
Smart domain	tRNA_SAD (94.81%), TRANS (0.08%), SIGNAL (0.04%)
GO slim	GO:0006399 (68.14%), GO:0006520 (68.14%), GO:0006351 (0.11%)
KEGG pathway	map00970 (22.22%), ko00970 (22.22%)
KEGG ortholog	K01872 (22.22%), K07050 (0.13%)
KEGG gene symbol	AARS (22.22%), alaS (22.22%), ALAX (0.13%)
KEGG gene name	alanyl-tRNA synthetase [EC:6.1.1.7] (22.22%), misacylated tRNA(Ala) deacylase [EC:3.1.1.-] (0.13%)

PARENT OGS
COG0013
LCOG0013

CHILDREN OG
4UJ46
4V4PB
4V8PY
4W36S
4W3QN
4W3Y0
4WE56
4WHA1

OG members | Taxonomic profile | Functional profile | Duplications profile | Tree and alignment

B
Citrobacter koseri ATCC BAA-895
Citrobacter freundii complex
Salmonella enterica subsp. indica serovar 6,14,25:z10:1,(2),7 str. 112
Salmonella enterica subsp. enterica
Salmonella enterica subsp. diarizonae
Salmonella enterica subsp. salamae serovar 56:z10:e,n,x str. 1369-73
Salmonella enterica subsp. houtenae str. ATCC BAA-1581
Salmonella enterica subsp. arizonae serovar 62:z4,z23:-
Salmonella bongori NCTC 12419
Citrobacter
Citrobacter rodentium ICC168
Escherichia coli
Escherichia coli IAI39
Shigella flexneri 2a str. 301
Enterobacteriaceae
Shigella boydii CDC 3083-94
Escherichia coli O157:H7
Shigella dysenteriae Sd197
tRNA-synt_2c
DHHA1

Fig. 6 An eggNOG summary card of the AlaS orthogroup (COG0013) in bacteria domain. (**a**) list of functional annotations from different sources (KEGG, GO, etc.) is summarized in cards. Expandable cards display detailed information about species members of the orthogroup, taxonomic profile, and interactive visualization tool for phylogenetic trees. (**b**) Interactive visualization of phylogenetic tree indicating speciation or duplication events on nodes coupled to a schematic representation of the gene domain structure. Both figures are screenshots of the search results page for AlaS protein (http:/eggnog6.embl.de/) [11]

9 Conclusion

Orthology computation based on protein-coding gene datasets is at the heart of most genome comparisons; it is also a nontrivial computational task, largely because gene histories can be quite complicated. Various practical methods have been developed in the last several years, and the orthology computation problem is likely to remain an active area of research for many years. In this chapter, we have provided a simple, practical guide for orthology computation using four programs, which are among the most popular methods for this task. Our principal aim was to provide readers with a simple framework that they could apply to their own datasets.

Acknowledgments

This work was supported in part by a CNPq senior researcher fellowship to JCS.

References

1. Fitch WM (1970) Distinguishing homologous from analogous proteins. Syst Zool 19:99–113
2. Altenhoff AM, Studer RA, Robinson-Rechavi-M, Dessimoz C (2012) Resolving the ortholog conjecture: orthologs tend to be weakly, but significantly, more similar in function than paralogs. PLoS Comput Biol 8:e1002514. https://doi.org/10.1371/journal.pcbi.1002514

3. Huerta-Cepas J, Forslund K, Coelho LP et al (2017) Fast genome-wide functional annotation through orthology assignment by eggNOG-mapper. Mol Biol Evol 34:2115–2122. https://doi.org/10.1093/molbev/msx148
4. Gabaldón T, Koonin EV (2013) Functional and evolutionary implications of gene orthology. Nat Rev Genet 14:360–366. https://doi.org/10.1038/nrg3456
5. Nevers Y, Defosset A, Lecompte O (2020) Orthology: promises and challenges. In: Pontarotti P (ed) Evolutionary biology – a transdisciplinary approach. Springer, Cham, pp 203–228
6. Altenhoff AM, Glover NM, Dessimoz C (2019) Inferring orthology and paralogy. In: Anisimova M (ed) Evolutionary genomics: statistical and computational methods. Springer, New York, pp 149–175
7. Fernández R, Gabaldon T, Dessimoz C (2020) Orthology: definitions, prediction, and impact on species phylogeny inference. 2.4:1
8. Wolf YI, Koonin EV (2012) A tight link between orthologs and bidirectional best hits in bacterial and archaeal genomes. Genome Biol Evol 4:1286–1294. https://doi.org/10.1093/gbe/evs100
9. Emms DM, Kelly S (2019) OrthoFinder: phylogenetic orthology inference for comparative genomics. Genome Biol 20:238. https://doi.org/10.1186/s13059-019-1832-y
10. Cantalapiedra CP, Hernández-Plaza A, Letunic I et al (2021) eggNOG-mapper v2: functional annotation, orthology assignments, and domain prediction at the metagenomic scale. Mol Biol Evol 38:5825–5829. https://doi.org/10.1093/molbev/msab293
11. Hernández-Plaza A, Szklarczyk D, Botas J et al (2023) eggNOG 6.0: enabling comparative genomics across 12 535 organisms. Nucleic Acids Res 51:D389–D394. https://doi.org/10.1093/nar/gkac1022
12. Benson DA, Cavanaugh M, Clark K et al (2018) GenBank. Nucleic Acids Res 46:D41–D47. https://doi.org/10.1093/nar/gkx1094
13. Contreras-Moreira B, Vinuesa P (2013) GET_HOMOLOGUES, a versatile software package for scalable and robust microbial pangenome analysis. Appl Environ Microbiol 79:7696–7701. https://doi.org/10.1128/AEM.02411-13
14. Yu G, Smith DK, Zhu H et al (2017) ggtree: an r package for visualization and annotation of phylogenetic trees with their covariates and other associated data. Methods Ecol Evol 8:28–36. https://doi.org/10.1111/2041-210X.12628
15. Minh BQ, Schmidt HA, Chernomor O et al (2020) IQ-TREE 2: new models and efficient methods for phylogenetic inference in the genomic era. Mol Biol Evol 37:1530–1534. https://doi.org/10.1093/molbev/msaa015
16. Katoh K, Standley DM (2013) MAFFT multiple sequence alignment software version 7: improvements in performance and usability. Mol Biol Evol 30:772–780. https://doi.org/10.1093/molbev/mst010
17. Altenhoff AM, Levy J, Zarowiecki M et al (2019) OMA standalone: orthology inference among public and custom genomes and transcriptomes. Genome Res 29:1152–1163. https://doi.org/10.1101/gr.243212.118
18. Kuznetsov D, Tegenfeldt F, Manni M et al (2023) OrthoDB v11: annotation of orthologs in the widest sampling of organismal diversity. Nucleic Acids Res 51:D445–D451. https://doi.org/10.1093/nar/gkac998
19. Shen W, Le S, Li Y, Hu F (2016) SeqKit: a cross-platform and ultrafast toolkit for FASTA/Q file manipulation. PLoS One 11: e0163962. https://doi.org/10.1371/journal.pone.0163962
20. Capella-Gutiérrez S, Silla-Martínez JM, Gabaldón T (2009) trimAl: a tool for automated alignment trimming in large-scale phylogenetic analyses. Bioinformatics 25:1972–1973. https://doi.org/10.1093/bioinformatics/btp348
21. Li L, Stoeckert CJ, Roos DS (2003) OrthoMCL: identification of ortholog groups for eukaryotic genomes. Genome Res 13: 2178–2189. https://doi.org/10.1101/gr.1224503
22. Setubal JC, Stadler PF (2018) Gene phylogenies and orthologous groups. Methods Mol Biol Clifton NJ 1704:1–28. https://doi.org/10.1007/978-1-4939-7463-4_1
23. Tatusov RL, Koonin EV, Lipman DJ (1997) A genomic perspective on protein families. Science 278:631–637. https://doi.org/10.1126/science.278.5338.631
24. Galperin MY, Kristensen DM, Makarova KS et al (2019) Microbial genome analysis: the COG approach. Brief Bioinform 20:1063–1070. https://doi.org/10.1093/bib/bbx117
25. Emms DM, Kelly S (2015) OrthoFinder: solving fundamental biases in whole genome comparisons dramatically improves orthogroup inference accuracy. Genome Biol 16:157. https://doi.org/10.1186/s13059-015-0721-2

26. Nevers Y, Jones TEM, Jyothi D et al (2022) The quest for orthologs orthology benchmark service in 2022. Nucleic Acids Res 50:W623–W632. https://doi.org/10.1093/nar/gkac330
27. Zahn-Zabal M, Dessimoz C, Glover NM (2020) Identifying orthologs with OMA: a primer. F1000Research 9:27. https://doi.org/10.12688/f1000research.21508.1
28. Altenhoff AM, Gil M, Gonnet GH, Dessimoz C (2013) Inferring hierarchical orthologous groups from orthologous gene pairs. PLoS One 8:e53786. https://doi.org/10.1371/journal.pone.0053786
29. Linard B, Ebersberger I, McGlynn SE et al (2021) Ten years of collaborative progress in the quest for orthologs. Mol Biol Evol 38:3033–3045. https://doi.org/10.1093/molbev/msab098
30. Persson E, Kaduk M, Forslund SK, Sonnhammer ELL (2019) Domainoid: domain-oriented orthology inference. BMC Bioinf 20:523. https://doi.org/10.1186/s12859-019-3137-2
31. Nevers Y, Kress A, Defosset A et al (2019) OrthoInspector 3.0: open portal for comparative genomics. Nucleic Acids Res 47:D411–D418. https://doi.org/10.1093/nar/gky1068
32. Galperin MY, Wolf YI, Makarova KS et al (2021) COG database update: focus on microbial diversity, model organisms, and widespread pathogens. Nucleic Acids Res 49:D274–D281. https://doi.org/10.1093/nar/gkaa1018
33. The Gene Ontology Consortium (2019) The gene ontology resource: 20 years and still GOing strong. Nucleic Acids Res 47:D330–D338. https://doi.org/10.1093/nar/gky1055
34. Kanehisa M, Goto S, Sato Y et al (2014) Data, information, knowledge and principle: back to metabolism in KEGG. Nucleic Acids Res 42:D199–D205. https://doi.org/10.1093/nar/gkt1076
35. Drula E, Garron M-L, Dogan S et al (2022) The carbohydrate-active enzyme database: functions and literature. Nucleic Acids Res 50:D571–D577. https://doi.org/10.1093/nar/gkab1045
36. Buchfink B, Xie C, Huson DH (2015) Fast and sensitive protein alignment using DIAMOND. Nat Methods 12:59–60. https://doi.org/10.1038/nmeth.3176
37. Steinegger M, Söding J (2017) MMseqs2 enables sensitive protein sequence searching for the analysis of massive data sets. Nat Biotechnol 35:1026–1028. https://doi.org/10.1038/nbt.3988
38. Mistry J, Finn RD, Eddy SR et al (2013) Challenges in homology search: HMMER3 and convergent evolution of coiled-coil regions. Nucleic Acids Res 41:e121. https://doi.org/10.1093/nar/gkt263
39. Koonin EV (2003) Comparative genomics, minimal gene-sets and the last universal common ancestor. Nat Rev Microbiol 1:127–136. https://doi.org/10.1038/nrmicro751
40. Wu M, Eisen JA (2008) A simple, fast, and accurate method of phylogenomic inference. Genome Biol 9:R151. https://doi.org/10.1186/gb-2008-9-10-r151
41. Wu D, Jospin G, Eisen JA (2013) Systematic identification of gene families for use as "markers" for phylogenetic and phylogeny-driven ecological studies of bacteria and archaea and their major subgroups. PLoS One 8:e77033. https://doi.org/10.1371/journal.pone.0077033
42. Lan Y, Rosen G, Hershberg R (2016) Marker genes that are less conserved in their sequences are useful for predicting genome-wide similarity levels between closely related prokaryotic strains. Microbiome 4:18. https://doi.org/10.1186/s40168-016-0162-5
43. Wang S, Ventolero M, Hu H, Li X (2022) A revisit to universal single-copy genes in bacterial genomes. Sci Rep 12:14550. https://doi.org/10.1038/s41598-022-18762-z
44. Setubal JC, Stadler P (2018) Gene phylogenies and orthologous groups. In: Methods in molecular biology, vol 1704. Springer New York, pp 1–28
45. Huerta-Cepas J, Szklarczyk D, Heller D et al (2019) eggNOG 5.0: a hierarchical, functionally and phylogenetically annotated orthology resource based on 5090 organisms and 2502 viruses. Nucleic Acids Res 47:D309–D314. https://doi.org/10.1093/nar/gky1085
46. Zdobnov EM, Kuznetsov D, Tegenfeldt F et al (2021) OrthoDB in 2020: evolutionary and functional annotations of orthologs. Nucleic Acids Res 49:D389–D393. https://doi.org/10.1093/nar/gkaa1009
47. Schreiber F, Patricio M, Muffato M et al (2014) TreeFam v9: a new website, more species and orthology-on-the-fly. Nucleic Acids Res 42:D922–D925. https://doi.org/10.1093/nar/gkt1055
48. Arnold R, Goldenberg F, Mewes H-W, Rattei T (2014) SIMAP – the database of all-against-all protein sequence similarities and annotations with new interfaces and increased coverage. Nucleic Acids Res 42:D279–D284. https://doi.org/10.1093/nar/gkt970