OMAmer: tree-driven and alignment-free protein assignment to subfamilies outperforms closest sequence approaches

Victor Rossier[1,2,3], Alex Warwick Vesztrocy[1,2,3], Marc Robinson-Rechavi[4,5,*] and Christophe Dessimoz[1,2,3,5,6,*]

[1]Department of Computational Biology, University of Lausanne, Switzerland; [2]Center for Integrative Genomics, University of Lausanne, Switzerland; [3]SIB Swiss Institute of Bioinformatics, Lausanne, Switzerland; [4]Department of Ecology and Evolution, University of Lausanne, Switzerland; [5]Department of Genetics, Evolution, and Environment, University College London, UK; [6]Department of Computer Science, University College London, UK.

*Corresponding authors: Marc.Robinson-Rechavi@unil.ch & Christophe.Dessimoz@unil.ch

Abstract

Assigning new sequences to known protein families and subfamilies is a prerequisite for many functional, comparative and evolutionary genomics analyses. Such assignment is commonly achieved by looking for the closest sequence in a reference database, using a method such as BLAST. However, ignoring the gene phylogeny can be misleading because a query sequence does not necessarily belong to the same subfamily as its closest sequence. For example, a hemoglobin which branched out prior to the hemoglobin alpha/beta duplication could be closest to a hemoglobin alpha or beta sequence, whereas it is neither. To overcome this problem, phylogeny-driven tools have emerged but rely on gene trees, whose inference is computationally expensive.

Here, we first show that in multiple animal datasets, 19 to 68% of assignments by closest sequence are misassigned, typically to an over-specific subfamily. Then, we introduce OMAmer, a novel alignment-free protein subfamily assignment method, which limits over-specific subfamily assignments and is suited to phylogenomic databases with thousands of genomes. OMAmer is based on an innovative method using subfamily-informed *k*-mers for alignment-free mapping to ancestral protein subfamilies. Whilst able to reject non-homologous family-level assignments, we show that OMAmer provides better and quicker subfamily-level assignments than approaches relying on the closest sequence, whether inferred exactly by Smith-Waterman or by the fast heuristic DIAMOND.

OMAmer is available from the Python Package Index (as omamer), with the source code available at https://github.com/DessimozLab/omamer.

Introduction

Assigning new sequences to known protein families is a prerequisite for many functional, comparative, and evolutionary analyses (Glover et al., 2019). For instance, multiple sequence alignments and gene trees can be inferred from the expanded families to investigate selection pressure or to reconstruct the species tree (Fernández et al., 2019). Expression data and epigenetic markers can be compared between new sequences and reference proteins (Lowdon et al., 2016; Breschi et al., 2017; Roux et al., 2015). Functional knowledge can also be transferred from reference to new sequences assigned in the same family (Gabaldón and Koonin, 2013).

However, when gene duplication events have resulted in multiple copies per species, multiple "subfamilies" are generated, which can make placing a protein sequence into the correct subfamily challenging. Gene subfamilies are nested gene families defined after duplication events and organized hierarchically into gene trees. For example, the epsilon and gamma hemoglobin subfamilies are defined at the placental level, and nested in the adult hemoglobin beta subfamily at the mammal level (Opazo et al., 2008). Both belong to the globin family that originated in the LUCA (last universal common ancestor of cellular life).

Gene subfamily assignment is commonly achieved by looking for the most similar ("closest") sequence in a reference database, using a method such as BLAST (Altschul et al., 1990) or HMMER (Eddy, 2011), before assigning the query to the subfamily of the closest sequence identified. For example, EggNOG mapper uses reference subfamilies from EggNOG to functionally annotate millions of unknown proteins of genomes and metagenomes (Huerta-Cepas et al., 2017, 2019). Briefly, each query is assigned to the most specific gene subfamily of its closest sequence, inferred using DIAMOND (Buchfink et al., 2015), with functional annotations then transferred accordingly. This approach is straightforward because it relies on the long-standing practice of using the closest sequence to a query as the main source of information to infer its biological characteristics.

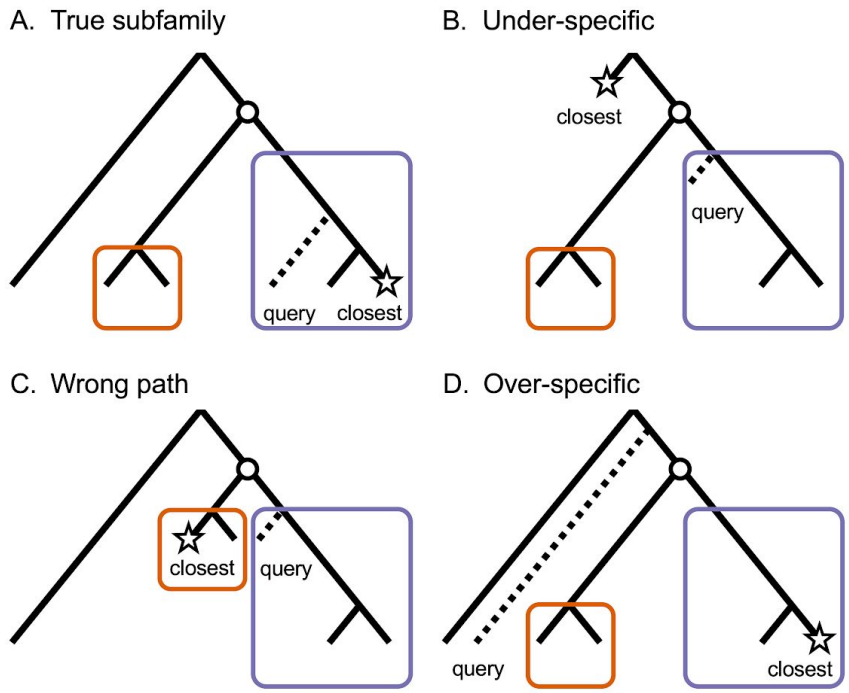


Fig. 1. The closest sequence to a query does not necessarily belong to the same subfamily. This figure conceptualizes the four possible closest sequence locations relative to the query. On each tree, the true position of the query is indicated by a dashed branch, while its closest sequence in the family is indicated by a star. The circle represents a duplication event leading to two subfamilies depicted as color boxes.

However, ignoring the protein family tree can be misleading because a query sequence does not necessarily belong to the same subfamily as its closest sequence (Fig. 1). For instance, if the query branched out from a fast evolving subtree, its closest sequence might not belong to that subtree, but to a more general subfamily, or even not be classifiable in any known subfamily (Fig. 1. B). Or, in case of asymmetric evolutionary rates between sister subfamilies, the closest sequence might belong to a different subfamily altogether (Fig. 1. C). The prospect of observing these two scenarios is sustained by the long-standing observation that duplicated proteins experience accelerated and often asymmetric evolution (Conant and Wolfe, 2008; Sémon and Wolfe, 2007). Indeed, because they initially share the function of their ancestor, one or both duplicates can evolve under relaxed selection and lose or gain (sub)functions.

Moreover, the closest sequence to the query can belong to an over-specific subfamily even without any departure from the molecular clock in the family tree (Fig. 1. D). Such cases may occur stochastically when the query branched out before the emergence of nested subfamilies. Indeed, each protein descending from the query divergence has, all else being equal, the same chance of being the closest sequence to the query. Since duplications are common in evolution (Conant and Wolfe, 2008), finding such nested subfamilies as close relatives to the query divergence is expected to be common.

To avoid such errors, protein subfamily assignment tools relying on gene trees have been proposed (Schreiber et al., 2014; Tang et al., 2019). In short, these start by assigning queries to families with pairwise alignments against Hidden Markov profiles of reference families. Then, fine-grained assignments to subfamilies are performed with tree placement tools, which typically attempt to graft the query on every branch of the tree until maximizing a likelihood or parsimony score (Barbera et al., 2018). However, gene tree inference is computationally expensive and therefore not scalable to the exponentially growing number of available sequences.

As a more scalable alternative to gene trees, the concept of hierarchical orthologous groups (HOGs) (Altenhoff et al., 2013) provides a precise definition of the intuitive notion of protein families and subfamilies. Each HOG is a group of proteins descending from a single speciation event and organized hierarchically. Moreover, they collectively provide the evolutionary history of protein families and subfamilies, like gene trees. While the oldest HOG in the family hierarchy ("root-HOG") is the family itself, the other nested HOGs are its subfamilies. Thus, HOGs up to 100,000 members and covering thousands of species are available in large-scale phylogenomic databases (Altenhoff et al., 2018; Huerta-Cepas et al., 2019; Kriventseva et al., 2019).

Here, we first demonstrate on three animal genomes that 19 to 68% of assignments by closest sequence go to the incorrect, mostly over-specific, subfamilies. To overcome this problem, we introduce OMAmer, a novel alignment-free protein subfamily assignment method, which limits over-specific subfamily assignments and is suited to phylogenomic databases with thousands of genomes. We show that OMAmer is able to assign proteins to subfamilies more accurately than approaches relying on the closest sequence, whether inferred exactly by Smith-Waterman or by the fast heuristic DIAMOND. Furthermore, we show that by adopting efficient alignment-free *k*-mer based analyses pioneered by metagenomic taxonomic classifiers, and adapting them to protein subfamily-level classification, OMAmer is computationally faster and more scalable than DIAMOND.

Materials and methods

We start by introducing the OMAmer algorithm. Then, we describe the experiments used to validate and compare the accuracy, speed and memory consumption of OMAmer against closest sequence baselines. Finally, we describe the datasets and software parameters used in these experiments.

The OMAmer algorithm

In this section, we describe the two main algorithmic steps making OMAmer more accurate and faster than closest sequence approaches. First, to speed-up the protein assignment step, OMAmer preprocesses reference hierarchical orthologous groups (HOGs) into a *k*-mer table (Fig. 2). For each

family (root-HOG), this table stores the most likely ancestral subfamily (HOG) of each *k*-mer. Second, these subfamily-informed *k*-mers are used to reduce over-specific subfamily assignments (Fig. 3).

***K*-mer table precomputation**

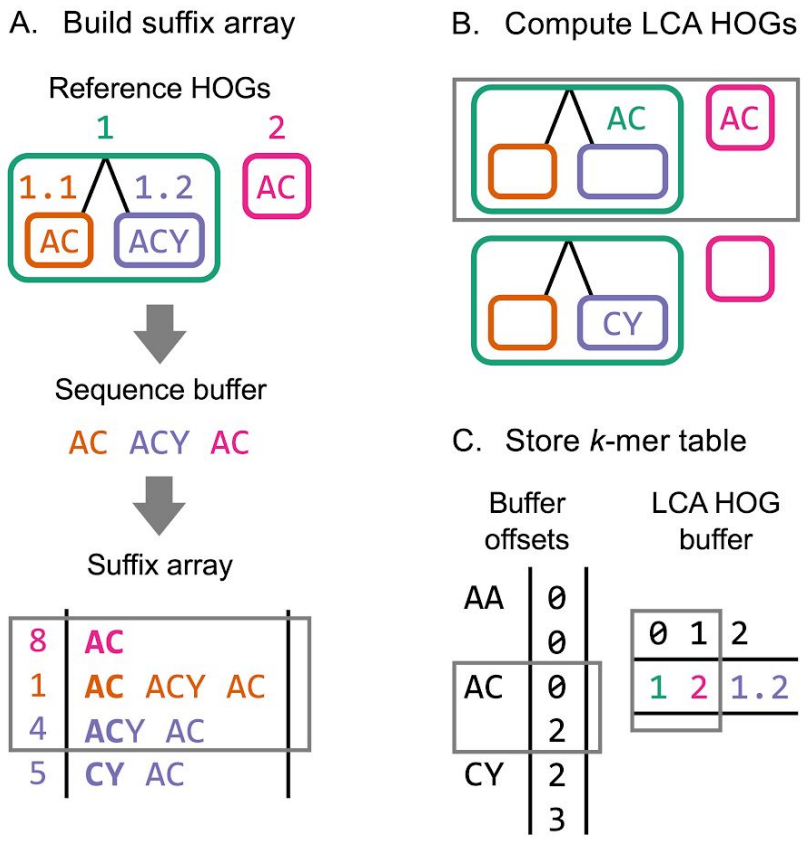


Fig. 2. OMAmer algorithm for compact *k*-mer table precomputation. A. To efficiently preprocess the *k*-mer table, a suffix array is first built from concatenated protein sequences of reference hierarchical orthologous groups (HOGs), encoding families (root-HOGs) and subfamilies (nested HOGs). Only suffixes starting with a *2*-mer are displayed. B. The last common ancestral HOG (LCA HOG) is computed for each *k*-mer and each root-HOG. For example, since both HOG 1.1 and 1.2 contain the "AC" *2*-mer, the LCA HOG for that *2*-mer in root-HOG 1 is root-HOG 1 itself. C. The final *k*-mer table includes two related arrays: a buffer storing LCA HOGs of each *k*-mer and an index storing LCA HOGs buffer offsets at the offset corresponding to the *k*-mer integer encoding (e.g. AA = 0, AC = 1, etc.). Grey boxes highlight the AC *2*-mer at each step.

To achieve a memory and time efficient preprocessing of reference HOGs, the *k*-mer table is built from the suffix array (Manber and Myers, 1993) of all concatenated reference proteins (Fig. 2. A.). Indeed, since suffix arrays are sorted alphabetically, all suffixes starting with a given *k*-mer are stored consecutively. This feature enables the identification of all proteins containing a given *k*-mer using binary search, without having to consider every single reference protein.

Then, the ancestral HOG of each *k*-mer is approximated within each family (root-HOG) as the last common ancestor HOG (LCA HOG) among all proteins with the given *k*-mer (Fig. 2. B). Essentially, this is the most specific HOG comprising all occurrences of the given *k*-mer within the family. Indeed, we assume that occurrences of the same *k*-mer in different members of a family mostly result from homology (*i.e.* same *k*-mer due to shared ancestry) rather than homoplasy (*i.e.* same *k*-mer arising independently). In the instances where the latter is true, the LCA HOG approximation will favor overly general assignments. Thus, compared to the homoplasy assumption that would favor over-specific assignments, this approach is more conservative.

Finally, to enable fast and memory efficient subfamily assignments, the resulting *k*-mer table is stored in compressed sparse row (CSR) format, consisting of two related arrays (Fig. 2. C). The LCA HOGs of each *k*-mer—one per root-HOG—are stored in a buffer array and retrieved through the index array. This array stores the buffer offsets corresponding

Family and subfamily assignment

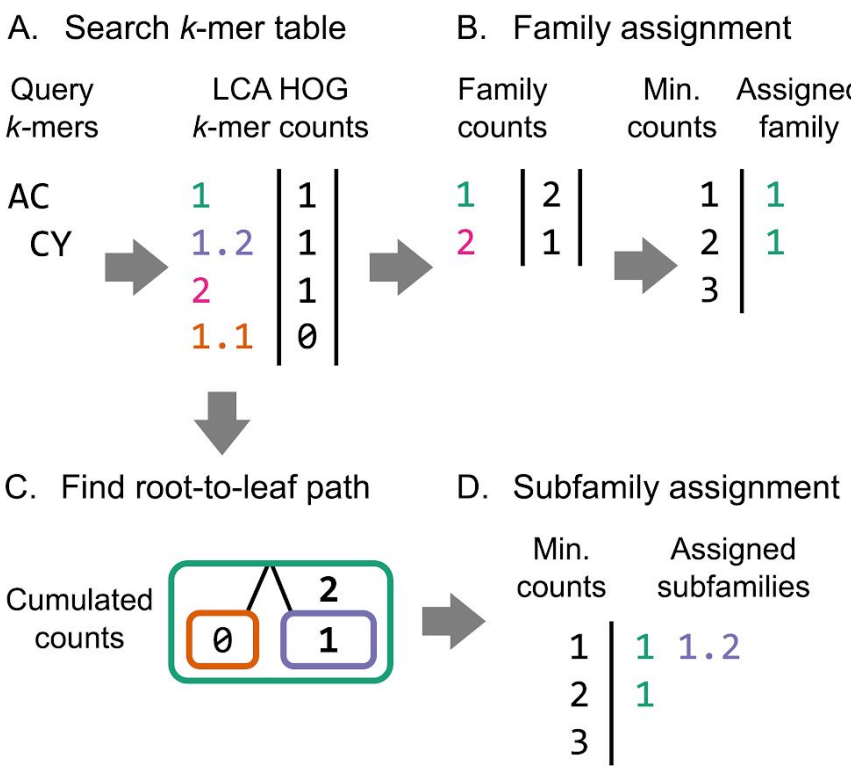


Fig. 3. OMAmer algorithm for protein family and subfamily assignment. A. Each query *k*-mer adds a count to each last common ancestor hierarchical orthologous group (LCA HOG)—precomputed in a *k*-mer table (e.g. Fig. 2)—containing the given *k*-mer. B. Simultaneously, family *k*-mer counts are computed as the sum of corresponding LCA HOG counts. Then, the query is assigned to the highest scoring family above a given threshold. C. Within that family, the highest scoring root-to-leaf path is computed from the LCA HOG counts formerly cumulated from leaves to root. D. The query is assigned to the most specific subfamily on that path with a score above a given threshold. For clarity, the *k*-mer count normalization is not shown here.

To avoid redundant accesses to the *k*-mer table, LCA HOG and family (root-HOG) *k*-mer counts are collected simultaneously by searching query *k*-mers—ignoring multiple occurrences of the same *k*-mer—in the precomputed *k*-mer table (Fig. 3. A and B). In particular, root-HOG counts are the sum of corresponding LCA HOG counts. Then, to make thresholds comparable across queries, each root-HOG score is calculated as the root-HOG counts divided by the number of *k*-mers in the query. Thus, the same threshold can be used on normalized scores for different queries. Finally, to further reduce unnecessary computation, each query is assigned to a single root-HOG before being placed in more specific HOGs (Fig. 3. B). Briefly, the family with the highest score above a given threshold is selected.

Then, refining the query assignment to nested HOGs starts by finding the highest scoring root-to-leaf path within the root-HOG (Fig. 3. C). First, LCA HOG *k*-mer counts are cumulated from leaves to root by adding the count of the highest scoring subtree to the current LCA HOG count at each multifurcation. Second, the highest scoring HOG is retained at each multifurcation when traversing the root-HOG from root to leaves. HOG scores are computed by dividing each HOG *k*-mer count by the number of *k*-mers in the query, from which was subtracted the number of *k*-mers shared with parent LCA HOGs. Finally, the query is assigned to the most specific HOG on that path with a score above a given threshold (Fig. 3. D). Thus, over-specific assignments can be avoided with an appropriate threshold.

Accuracy experiments

In this section, we describe the experiments conducted to evaluate the accuracy of OMAmer compared to closest (most similar) sequence baseline methods: Smith-Waterman (Smith and Waterman, 1981) and DIAMOND (Buchfink et al., 2015). Since placement in subfamilies initially requires accurate family-level assignments, we started by evaluating OMAmer at the family level. Second, to evaluate the impact of ignoring the phylogeny on subfamily assignments by closest sequences, we estimated the frequency of each closest sequence configuration, as depicted in Fig. 1. Third, we benchmarked subfamily-level assignments against closest sequence baselines. Finally, we broke down the validation results of OMAmer by closest sequence configuration.

Family-level validation

To simulate newly sequenced genomes, positive query sets were constructed as the sets of proteins from a given species contained in reference hierarchical orthologous groups (HOGs). The proteins of that species were hidden in the reference database used before the *k*-mer index precomputation.

Since query proteins do not necessarily have homologous counterparts in the reference families (e.g. “orphan” genes, contamination, horizontal gene transfer), validating family assignments also

required negative sets of non-homologous queries. Therefore, negative query sets were built with two approaches, while always matching the size of their corresponding positive set. In the first approach, random proteins were simply simulated with UniProtKB amino acid frequencies (release 2020_01) (UniProt Consortium, 2019) and sequence lengths of positive queries. The second approach was designed to resemble events of contamination or of horizontal gene transfer. Each negative query was randomly selected from a unique clade-specific family lying outside the taxonomic scope of reference families. In practice, clade-specific families were randomly selected among HOGs without parent (root-HOGs) at a given taxonomic level.

The resulting family assignments were compared with the truth set, and classified into true positives (TPs), false negatives (FNs) and false positives (FPs) for various score thresholds. FPs included negative queries assigned to a family as well as positive queries assigned to the wrong family. The remaining positive queries were divided into TPs and FNs depending on whether the score for their family of origin passed the threshold, or not. Finally, precision, recall and accuracy (F1) were computed from TPs, FNs and FPs (Table. 1), defined according to the score threshold.

In the following experiments, to assess subfamily-level assignment separately from family-level assignment, we focused on the query sequences assigned to the correct family (*i.e.* the set of TPs at the threshold where F1 is maximal [$F1_{max}$] for family assignment). Moreover, non-overlapping family-level TPs between methods being compared were further filtered out (sets of overlapping TPs are shown in Supp. Fig. 1.).

Table 1. Formulae of validation measures

Measure	Formula
Precision	$\frac{\#TPs}{(\#TP + \#FPs)}$
Recall	$\frac{\#TPs}{(\#TP + \#FNs)}$
Accuracy	$2x \frac{(precision * recall)}{(precision + recall)}$

#: number, TPs: true positives, FPs: false positives, FNs: false negatives.

Quantification of subfamily assignment errors by closest sequences

As a reference to find the closest sequence, we used Smith-Waterman local alignments (Smith and Waterman, 1981). Being an exact algorithm, Smith-Waterman is guaranteed to find the highest scoring match. Although there are cases where the closest sequence does not have the highest score, these cases typically arise when a query has only few detectable homologs (Koski and Golding, 2001). But since we only used Smith-Waterman alignments to identify the closest sequence within a

subfamily, this case does not apply. Thus, in the present context, we consider matches with highest scoring Smith-Waterman alignment to be reasonable proxies of the closest sequences.

Then, we classified each query according to the location of its closest sequence (Fig. 1.) as follows: a "true subfamily" configuration arises when the most specific HOG of the closest sequence is the same as the query one. An "over-specific" configuration arises when the most specific HOG of the query is ancestral to the most specific HOG of the closest sequence. Conversely, an "under-specific" configuration arises when the most specific HOG of the closest sequence is ancestral to that of the query. The last case is the "wrong-path" configuration, in which the most specific HOG of the query and of the closest sequence are in different parts of the family tree.

Subfamily-level validation

TPs, FNs and FPs were assessed at this level using two different approaches. The first approach takes the view that an assignment to a subfamily also implies assignment to its "parental" subfamilies—if there are any. For instance, let us consider a nested gene family of alcohol dehydrogenases. Under this view, an assignment to the specific "alcohol dehydrogenase 1C" is also implicitly an assignment to "alcohol dehydrogenase 1", as well as to "alcohol dehydrogenase". In this case, if a method incorrectly assigns the protein to the subfamily "alcohol dehydrogenase 1B", in addition to counting a FP (the gene is not a true member of subfamily "B") and a FN (the gene is missing from subfamily "C"), we also count one TP for correctly assigning to the parental sub-HOG "alcohol dehydrogenase 1". In effect, the prediction is regarded as being only partially wrong. Note that there is no TP counted for correctly implying an assignment to the root-HOG (alcohol dehydrogenase), because the present analysis only seeks to assess within-family placement.

The second approach takes the more stringent view that there are no implicit predictions of parental subfamilies, therefore no reward is given for partial correctness. Thus, in the previous example, there would be no TP counted—only one FP and one FN.

For both validation approaches, precision, recall and accuracy (F1) were computed from TPs, FPs, and FNs using the same formulae as at the family-level (Table 1).

Performance experiments

To benchmark the computational performance of OMAmer and DIAMOND, we measured real and CPU time, as well as the maximum resident set size (memory) using the GNU time command. All timing was performed on machines containing identical hardware (dual-socket Intel Xeon E5-2660, 64GB of RAM), with sole-use at the time of computation. Single threaded versions of both methods were used, with timing repeated 10 times in order to ensure stability.

Databases of increasing size (20 to 200 proteomes, in steps of 20) were generated from Metazoan proteomes, with each including all of the previous and an extra 20 randomly selected species. The

full proteomes of the initial 20 were used to query the databases of increasing size in order to gauge the scaling characteristics.

Software and datasets

OMAmer was compared with two closest sequence methods lying at different extremes of the speed-accuracy tradeoff: DIAMOND (v0.9.24.125) and Smith-Waterman, respectively. Due to the computational cost of performing Smith-Waterman alignments, we used pre-computed alignments from OMA (June 2019) (Altenhoff et al., 2018). DIAMOND databases were built with default parameters, and search-es for the most similar sequence were performed with effectively no significance requirement (E-value set to 1e6). The OMAmer *k*-mer table was built with a *k*-mer size of 6.

OMAmer directly yields family and subfamily predictions. For Smith-Waterman and DIAMOND, each query was assigned to the family and most specific subfamily of its most similar reference protein. To obtain multiple precision-recall values, predictions were computed for multiple score thresholds: E-values of 1e-322 to 1e6 for DIAMOND, alignment scores of 1 to 5,000 for Smith-Waterman and normalized *k*-mer counts of 0.001 to 0.996 for OMAmer.

To make family-level assignments comparable and well differentiated from subfamily-assignments, we selected OMA HOGs defined at the Metazoa taxonomic level as families, and their nested HOGs as subfamilies. To avoid low-confidence families, we further filtered out Metazoa HOGs with less than six proteins. We picked Metazoa because it is one of the largest clades in OMA. Then, we selected three species as experiment targets. Platypus, spotted gar and amphioxus were picked because they stand as outgroups of large clades in OMA and thus display some variability in divergence ages to reference species (Supp. Table 1). Clade-specific root-HOGs used to build the negative query set were picked at the Bacteria taxonomic level.

The reference dataset included 603,607 proteins from 188 species organized in 115,782 HOGs and including 5,296 root-HOGs (families). The query datasets included 2,948, 3,985 and 3,920 proteins of platypus, spotted gar, and amphioxus species, respectively. 2,616, 3,421 and 3,267 queries belonged to a subfamily (a nested HOG) in addition to the family (root-HOG).

Results

Before addressing the problem of subfamily placement, we first consider the problem of sequence placement at the overall family level (*i.e.* identifying the correct root hierarchical orthologous group, or “root-HOG”). Then, we present our analyses of the subfamily placement problem in four parts: First, we quantify the different types of errors resulting from the closest protein criterion. Second, we show that OMAmer overcomes many of these errors, resulting in higher accuracy than closest

sequence approaches. Third, we show that this accuracy improvement is mainly achieved by avoiding over-specific sequence classification. And fourth, we compare the computational cost and scaling of OMAmer and DIAMOND.

At the overall family level, efficient sequence placement is a largely solved problem

Before placing sequences within subfamilies, queries must first be assigned to families. We evaluated this using DIAMOND and OMAmer, assessing the ability of the methods to either correctly place a protein in its correct family, or refrain from placing a sequence for which no homologous family is present in the reference database (see *Methods*).

Both methods delivered near perfect results in placing platypus and spotted gar proteins ($F1_{max}$ > 0.97; Supp. Fig. 2). The methods did not perform as well on the amphioxus genome (OMAmer $F1_{max}$ = 0.83-0.88; DIAMOND $F1_{max}$ = 0.91-0.93; Supp. Fig. 2), but this is an outgroup to all other chordates in OMA, with a divergence of more than 500 MY (Peterson and Eernisse, 2016) to the closest species sampled (*i.e.* all vertebrates and urochordates), presenting a worst-case scenario relative to placement of a new genome in an already sampled clade. Still, this first analysis indicates that, with reference genomes within the same phylum, assigning protein sequences at the family-level is a largely solved problem.

The closest sequence to a query is often not in the same subfamily

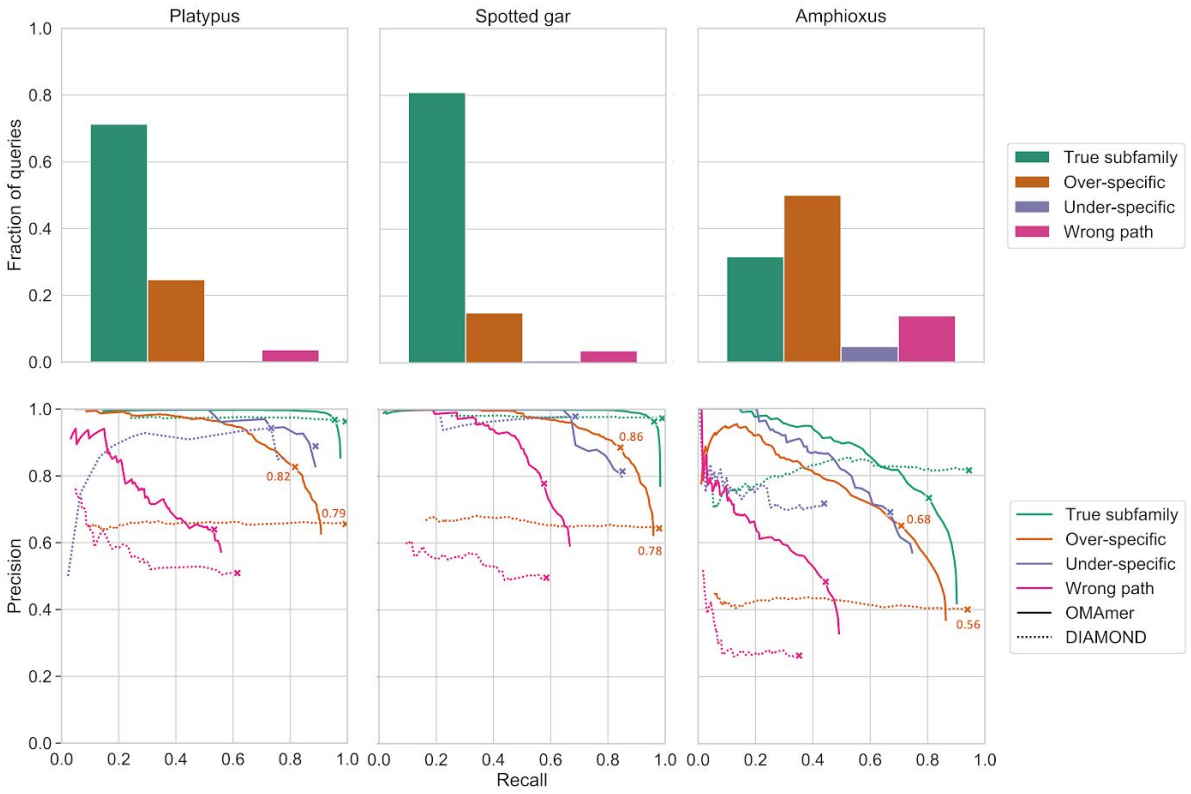


Fig. 4. Frequency of closest sequence configurations defined in Fig. 1 and OMAmer accuracy for each. A. The closest sequence to a query was often found in another subfamily. Smith-Waterman alignments were used as proxies for closest sequences. B. "Over-specific" configurations were especially well dealt with by OMAmer. Each curve displays the range of trade-offs between precision and recall when varying a score threshold. They were computed by breaking down queries by closest sequence configurations as in panel A, before the validation procedure itself (the relaxed one). These results are consistent with the stringent validation procedure (Supp. Fig. 4). Crosses indicate the location of $F1_{max}$ values. Over-specific $F1_{max}$ values are specifically annotated.

For a large proportion of query sequences (19-68%), the closest counterpart (inferred as the highest scoring Smith-Waterman match, see *Methods*) belongs to a different subfamily (Fig. 4. A). In such cases, the closest sequence most often belongs to a more specific subfamily (15-50% of all queries). These results highlight the need to account for the phylogeny, especially in the presence of many nested subfamilies.

OMAmer is more precise in subfamily placement

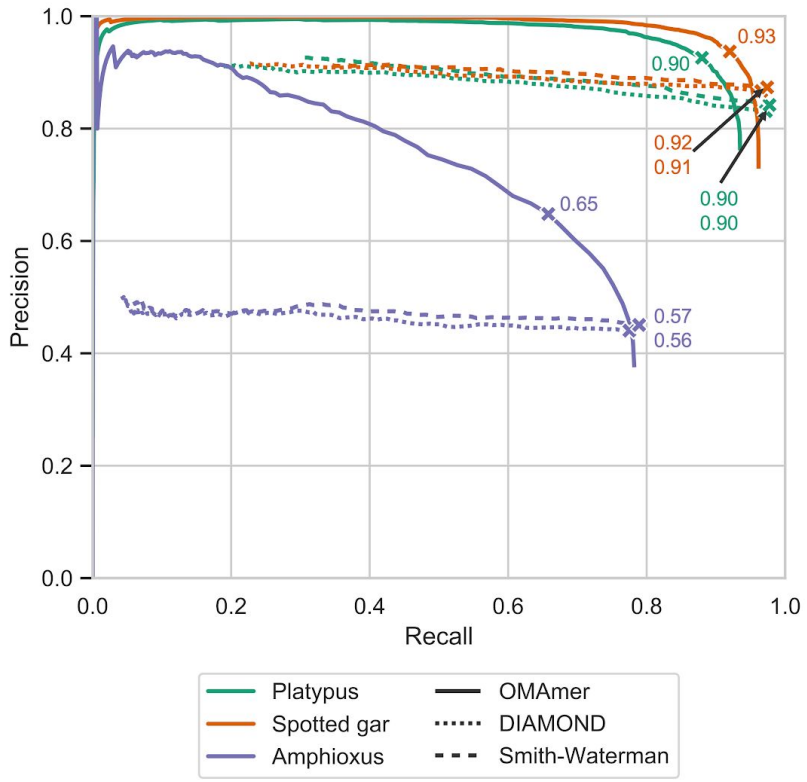


Fig. 5. Comparison of subfamily assignments with OMAmer and by closest sequence (DIAMOND and Smith-Waterman). Each curve displays the range of trade-offs between precision and recall when varying a score threshold. These results were computed using a relaxed validation procedure and are consistent with the stringent procedure (Supp. Fig. 3). $F1_{max}$ values are annotated at their locations indicated by crosses.

Solving this problem is the primary aim of OMAmer. We compared OMAmer with closest sequence methods (DIAMOND and Smith-Waterman) using two validation procedures: relaxed and stringent. OMAmer systematically achieved, or equaled, the highest accuracy ($F1_{max}$) across species, with both relaxed (Fig. 5.) and stringent (Supp. Fig. 3) validation procedures. Specifically, increases in $F1_{max}$ values between OMAmer and closest sequence methods ranged from 0.00 to 0.09 with the relaxed procedure and from 0.01 to 0.22 with the stringent one. Moreover, OMAmer $F1_{max}$ values were obtained from large precision gains for limited recall costs compared to closest sequence base-lines. Finally, score thresholds at $F1_{max}$ between platypus and spotted gar were highly congruent (0.13, 0.12 for both relaxed and stringent validation procedures), while amphioxus displayed lower optimal score thresholds (relaxed: 0.05, stringent:0.07).

Furthermore, OMAmer provides a genuine precision-recall trade-off, providing users with the possibility of obtaining very high precision, at the cost of lower recall. Such trade-off is not possible with closest sequence methods: varying the E-value threshold has very limited impact on precision (Fig. 5 and Supp. Fig. 3).

OMAmer deals especially well with over-specific closest sequences

As previously mentioned, over-specific placement is the most frequent mistake when only relying on assignments by closest sequences (Fig. 4. A). Since OMAmer was specifically designed to deal with such cases using subfamily-informed *k*-mers mapping toward ancestral subfamilies, we investigated whether this feature would explain OMAmer performance. Therefore, we reproduced the subfamily-level validation procedure with queries partitioned between the types of closest sequence configuration depicted in Fig. 1. and quantified in Fig. 4 A.

As expected, OMAmer was especially accurate—compared to DIAMOND—for queries in the "over-specific" configuration across species, with both the relaxed (Fig. 4. B) and stringent (Supp. Fig. 4) validation procedures. Specifically, for these queries, increases in $F1_{max}$ values between OMAmer and DIAMOND ranged from 0.03 to 0.12 with the relaxed procedure and from 0.45 to 0.51 with the stringent one. Finally, OMAmer displayed a proportion of over-specific assignments—defined at $F1_{max}$ and between validation approaches—0.07 to 0.37 lower than Smith-Waterman and DIAMOND (Supp. Fig. 5).

This performance for queries in the "over-specific" configuration was achieved while sacrificing very little accuracy for queries in the "true subfamily" configuration (Fig. 4. B and Supp. Fig. 4). Thus, the specificity of the OMAmer algorithm—subfamily-informed *k*-mers mapping toward ancestral subfamilies—probably explains its overall higher accuracy compared to closest sequence baselines (Fig. 5 and Supp. Fig. 3.).

Finally, despite their small number, queries in the “wrong-path” configuration were also placed more accurately by OMAmer. “Under-specific” configurations were too few to draw any conclusion.

OMAmer displays near-constant run time with the number of reference genomes

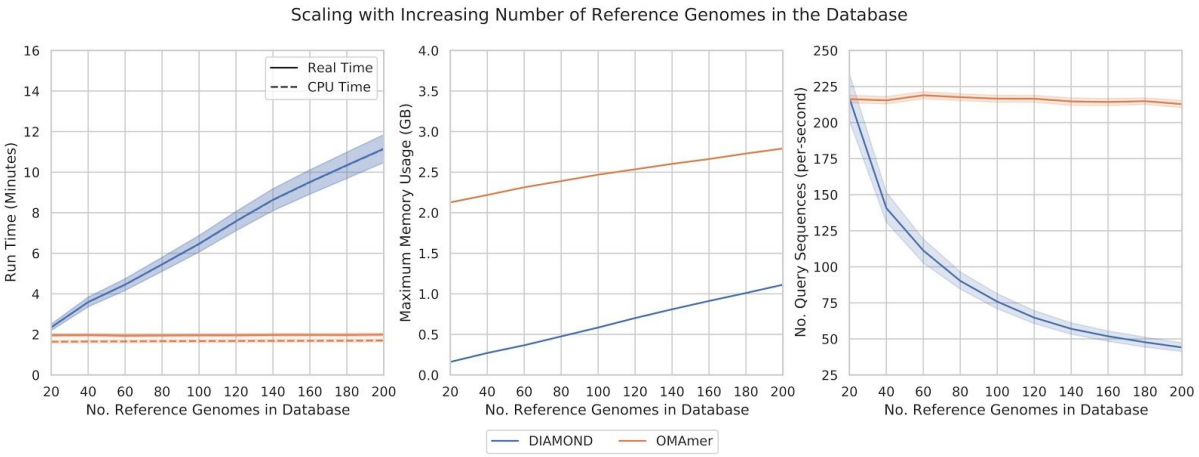


Fig. 6. Comparison of the computational performance of family and subfamily assignments between OMAmer and DIAMOND. (Left) run time (CPU and real) showing a sub-linear increase in run time for DIAMOND and near-constant time for OMAmer. (Middle) maximum memory usage of the two methods. (Right) number of queries processed per second. Error bars shown for 95% confidence interval, estimated using 10,000 bootstraps.

In an empirical scaling analysis, we varied the number of reference genomes in the database whilst querying a number of full-proteomes (see *Methods* for details). DIAMOND achieved sub-linear scaling, whereas OMAmer exhibited near-constant run time when increasing the number of reference genomes in the database (Fig. 6, left). Both methods, however, exhibited a similar increase in maximum memory usage (Fig. 6, middle), with OMAmer initially using over 2GB and DIAMOND using less than 256MB on a database of 20 reference genomes. In order to achieve this performance, OMAmer only stores *k*-mers once per LCA HOG. This does require extra computation, with the overhead being reflected in its memory usage and time to build the database (Supp. Fig. 6), taking between 15-20 minutes in comparison to 1-2 minutes for DIAMOND.

To put the timing into context, OMAmer is processing more than 200 query sequences per second (Fig. 6, right). DIAMOND starts with similar performance, before trailing off to less than 50 with the largest number of reference genomes

Discussion

In this study, we demonstrate that considering the phylogenetic relations between orthologous groups is essential for the problem of subfamily assignment. Indeed, although alignment-free,

OMAmer generally outperforms closest (most similar) sequence approaches, even when inferred by the exact Smith-Waterman algorithm. In particular, OMAmer systematically achieved, or equaled Smith-Waterman, for the best precision-recall trade-off ($F1_{max}$).

However, the main advantage of OMAmer is its control over assignment precision through the setting of specific thresholds that refrain over-specific placements. By contrast, relying on the closest sequence does not provide the ability for any precision-recall trade-off. Each assignment is bound to the most specific subfamily of the closest sequence, and varying the E-value threshold has a large impact on recall but almost none on precision. Thus, while closest sequence approaches are useful for cases where high recall is the overriding priority, OMAmer is more flexible and applicable in a broad range of contexts.

In addition to providing robust subfamily assignments, OMAmer scales in near-constant run time with the number of reference genomes. This is achieved with alignment-free sequence comparisons against hierarchical orthologous groups (HOGs) instead of exact or even approximate alignments against protein sequences. Indeed, in addition to removing the computational burden of sequence alignment, merging sequence information in HOGs drastically reduces the number of comparisons. This is especially true since the number of reference HOGs increases more slowly than proteins with the number of reference genomes.

Large-scale sequencing projects of genomes or metagenomes add difficulties such as chimeric assemblies or contaminations, thus mixing gene families from different species. OMAmer was designed as a starting point for the integration of such heterogeneous data. Thus, instead of constraining subfamily assignments along the known taxonomy of query genomes, OMAmer performs taxonomically blind assignments. We hope that this feature will enable diverse applications of OMAmer: the detection of contamination and horizontal gene transfers, the binning of protein level metagenomic assemblies (Steinegger et al., 2018), and with some algorithmic adaptations, directly placing reads to skip genome assembly and annotation.

The OMAmer algorithm builds upon some key ideas of the metagenomic software Kraken, which classifies reads into the species taxonomy (Wood and Salzberg, 2014). Indeed, this task is analogous to protein subfamily assignments for two reasons. First, some prior knowledge, shaped as labelled reference sequences, is preprocessed before the assignment itself. Second, this prior knowledge is organized hierarchically in a tree graph. Thus, instead of relying on closest sequences, such methods of taxonomic classification exploit semi-phylogenetic information to improve their predictions. While MEGAN introduced the key idea of taking the LCA taxon among significant BLAST hits (Huson et al., 2007), Kraken scaled up the approach by preprocessing LCA taxa in a database of taxonomically-informed *k*-mers (Wood and Salzberg, 2014).

While inspired by Kraken, the OMAmer algorithm features some key algorithmic innovations to fit the case of assigning proteins to subfamilies. One difference lies in the types of events used to define clades or subtrees. Indeed, while taxa are defined by speciation nodes in Kraken, subfamilies are defined by duplication nodes in OMAmer. This is an important difference because duplication patterns are variable across protein families, whereas the reference taxonomy is the same for different genes and genomes in Kraken. Second, the dual problem of first placing sequences within families, followed by subfamily-level assignment is specific to OMAmer. Finally, by breaking down the computation of subfamily-informed *k*-mers by family, OMAmer indirectly overcomes one of Kraken's main weaknesses: its high memory footprint (Breitwieser et al. 2018).

Although placing proteins at the overall family level appears to be a generally solved problem in our analyses, we start to see some degradation with the amphioxus sequences (last common ancestor to vertebrates >500MY [Peterson and Eernisse, 2016]). We expect further degradation for cases where query genomes are even farther from the reference genomes, because relying on *k*-mer exact matches is likely to be less sensitive than Smith-Waterman alignments to detect distant homologs. Some avenues to increase OMAmer sensitivity in absence of closely related reference species could be explored: the use of a reduced alphabet, which compresses the mutual information of sequences being compared (Edgar 2004); or spaced seeds, *i.e.* non-contiguous *k*-mers, that have shown an increased sensitivity in metagenomics classification (Břinda et al. 2015). On the other hand, adding such very distant genomes is expected to be much rarer than adding genomes to an already sampled clade. This is especially true for the increase of sequences through projects such as i5k (insect genomes) (i5K Consortium, 2013) or the Vertebrate Genomes Project (Koepfli et al., 2015), where duplications and thus subfamilies are common and a solid backbone of reference genomes are available. OMAmer is especially well positioned to help classify the genes from such projects, which will present a challenge for slower or less precise methods.

Another avenue to improve OMAmer accuracy will be the development of a probabilistic score which can control for variation in the size of gene families and subfamilies, as well as their *k*-mer frequencies. Such a score could provide a stronger discriminative power between families, as well as a finer-grain criterion to stop placement within the hierarchy of subfamilies.

Meanwhile, one compelling application of OMAmer will be processing of the large number of genomes which will be produced by initiatives under the Earth BioGenome project (Lewin et al. 2018), which collectively aims to sequence all 1.5 million known eukaryotic species within the coming decade.

Acknowledgements

We thank Adrian Altenhoff for fruitful discussions in the conception and validation of OMAmer. Computations were performed at the Vital-IT Center for high performance computing of the SIB Swiss Institute of Bioinformatics, as well as the Wally and Axiom clusters of the University of Lausanne.

Funding

This work was supported by the Swiss National Foundation grant No 167276, as part of the National Research Program 75 “Big Data”, as well as Swiss National Foundation grant No 183723.

Conflict of Interest: none declared.

References

Altenhoff,A.M. *et al.* (2013) Inferring hierarchical orthologous groups from orthologous gene pairs. *PLoS One*, **8**, e53786.

Altenhoff,A.M. *et al.* (2018) The OMA orthology database in 2018: retrieving evolutionary relationships among all domains of life through richer web and programmatic interfaces. *Nucleic Acids Res.*, **46**, D477–D485.

Altschul,S.F. *et al.* (1990) Basic local alignment search tool. *J. Mol. Biol.*, **215**, 403–410.

Barbera,P. *et al.* (2018) EPA-ng: Massively Parallel Evolutionary Placement of Genetic Sequences. *Syst. Biol.*

Breitwieser,F.P. *et al.* (2018) KrakenUniq: confident and fast metagenomics classification using unique k-mer counts. *Genome Biol.*, **19**, 198.

Breschi,A. *et al.* (2017) Comparative transcriptomics in human and mouse. *Nat. Rev. Genet.*, **18**, 425–440.

Buchfink,B. *et al.* (2015) Fast and sensitive protein alignment using DIAMOND. *Nat. Methods*, **12**, 59–60.

Conant,G.C. and Wolfe,K.H. (2008) Turning a hobby into a job: how duplicated genes find new functions. *Nat. Rev. Genet.*, **9**, 938–950.

Eddy,S.R. (2011) Accelerated Profile HMM Searches. *PLoS Comput. Biol.*, **7**, e1002195.

Fernández,R. *et al.* (2019) Orthology: definitions, inference, and impact on species phylogeny inference. *arXiv [q-bio.PE]*.

Gabaldón,T. and Koonin,E.V. (2013) Functional and evolutionary implications of gene orthology. *Nat. Rev. Genet.*, **14**, 360–366.

Glover,N. *et al.* (2019) Advances and Applications in the Quest for Orthologs. *Mol. Biol. Evol.*, **36**, 2157–2164.

Huerta-Cepas,J. *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.

Huerta-Cepas,J. *et al.* (2017) Fast Genome-Wide Functional Annotation through Orthology Assignment by eggNOG-Mapper. *Mol. Biol. Evol.*, **34**, 2115–2122.

Huson,D.H. *et al.* (2007) MEGAN analysis of metagenomic data. *Genome Res.*, **17**, 377–386.

i5K Consortium (2013) The i5K Initiative: advancing arthropod genomics for knowledge, human health, agriculture, and the environment. *J. Hered.*, **104**, 595–600.

Koepfli,K.-P. *et al.* (2015) The Genome 10K Project: a way forward. *Annu Rev Anim Biosci*, **3**, 57–111.

Koski,L.B. and Golding,G.B. (2001) The closest BLAST hit is often not the nearest neighbor. *J. Mol. Evol.*, **52**, 540–542.

Kriventseva,E.V. *et al.* (2019) OrthoDB v10: sampling the diversity of animal, plant, fungal, protist, bacterial and viral genomes for evolutionary and functional annotations of orthologs. *Nucleic Acids Res.*, **47**, D807–D811.

Lewin,H.A. *et al.* (2018) Earth BioGenome Project: Sequencing life for the future of life. *Proc. Natl. Acad. Sci. U. S. A.*, **115**, 4325–4333.

Lowdon,R.F. *et al.* (2016) Evolution of Epigenetic Regulation in Vertebrate Genomes. *Trends Genet.*, **32**, 269–283.

Manber,U. and Myers,G. (1993) Suffix Arrays: A New Method for On-Line String Searches. *SIAM J. Comput.*, **22**, 935–948.

Opazo,J.C. *et al.* (2008) Differential loss of embryonic globin genes during the radiation of placental mammals. *Proc. Natl. Acad. Sci. U. S. A.*, **105**, 12950–12955.

Peterson,K.J. and Eernisse,D.J. (2016) The phylogeny, evolutionary developmental biology, and paleobiology of the Deuterostomia: 25 years of new techniques, new discoveries, and new ideas. *Org. Divers. Evol.*, **16**, 401–418.

Roux,J. *et al.* (2015) What to compare and how: Comparative transcriptomics for Evo-Devo. *J. Exp. Zool. B Mol. Dev. Evol.*, **324**, 372–382.

Schreiber,F. *et al.* (2014) TreeFam v9: a new website, more species and orthology-on-the-fly. *Nucleic Acids Res.*, **42**, D922–D925.

Sémon,M. and Wolfe,K.H. (2007) Consequences of genome duplication. *Curr. Opin. Genet. Dev.*, **17**, 505–512.

Smith,T.F. and Waterman,M.S. (1981) Identification of common molecular subsequences. *J. Mol. Biol.*, **147**, 195–197.

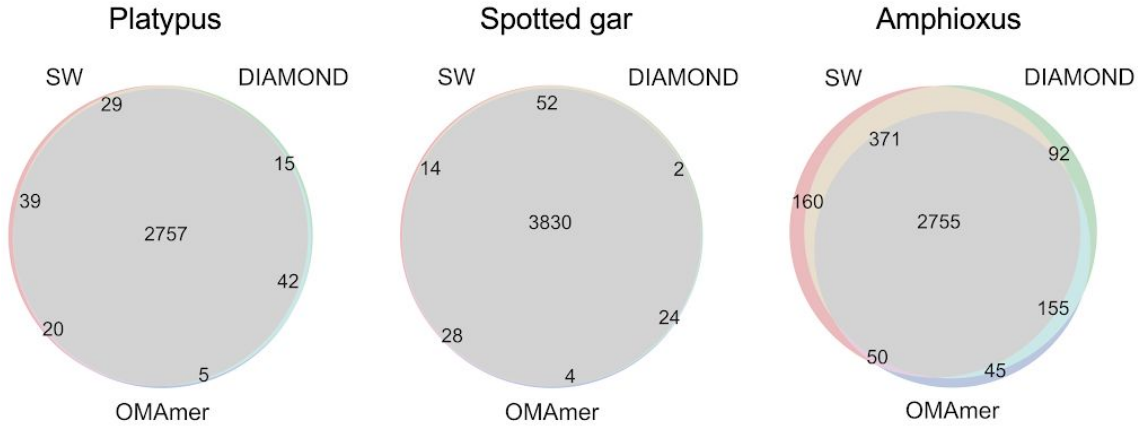
Steinegger,M. *et al.* (2018) Protein-level assembly increases protein sequence recovery from metagenomic samples manyfold. *bioRxiv*, 386110.

Tang,H. *et al.* (2019) TreeGrafter: phylogenetic tree-based annotation of proteins with Gene Ontology terms and other annotations. *Bioinformatics*, **35**, 518–520.

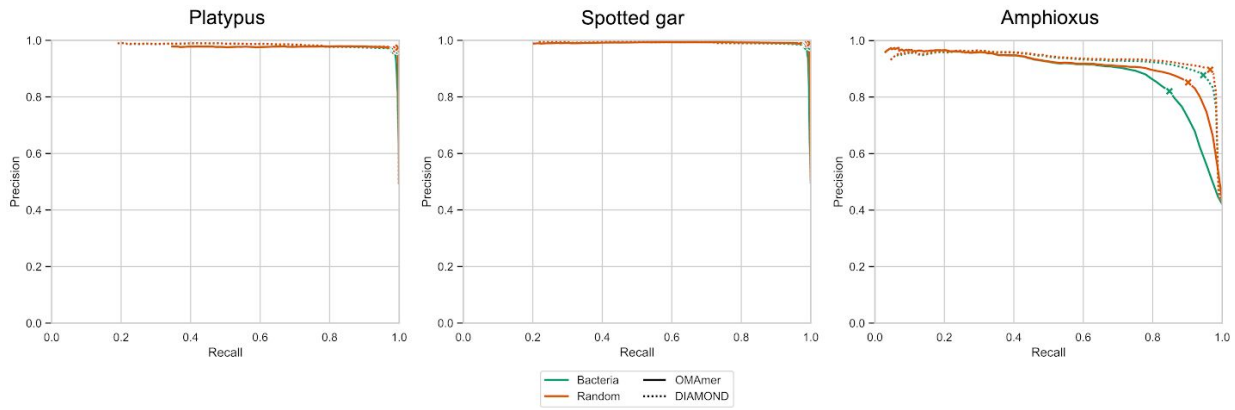
UniProt Consortium (2019) UniProt: a worldwide hub of protein knowledge. *Nucleic Acids Res.*, **47**, D506–D515.

Wood,D.E. and Salzberg,S.L. (2014) Kraken: ultrafast metagenomic sequence classification using exact alignments. *Genome Biol.*, **15**, R46.

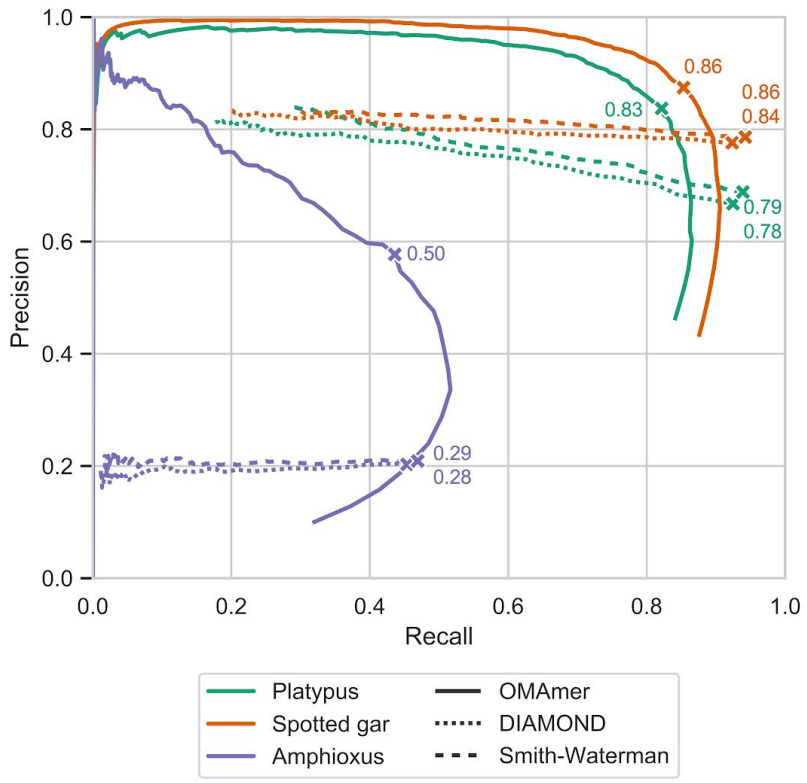
Supplementary material



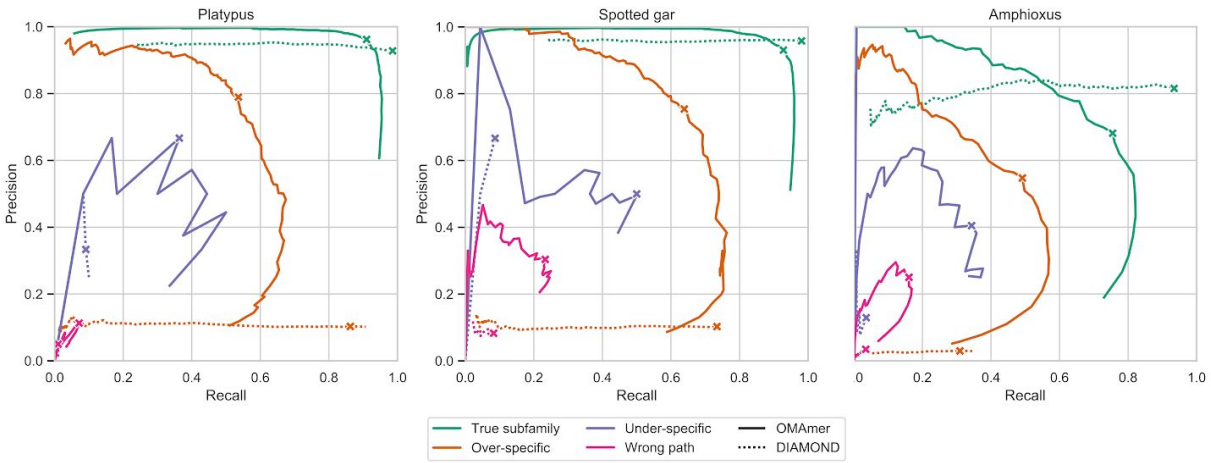
Supp. Fig. 1. Overlaps of family-level TP queries between methods. TP sets were defined at $F1_{max}$ for DIAMOND and OMAmer and at the minimum score (1) for Smith-Waterman alignments. These queries were used to assess subfamily assignment.



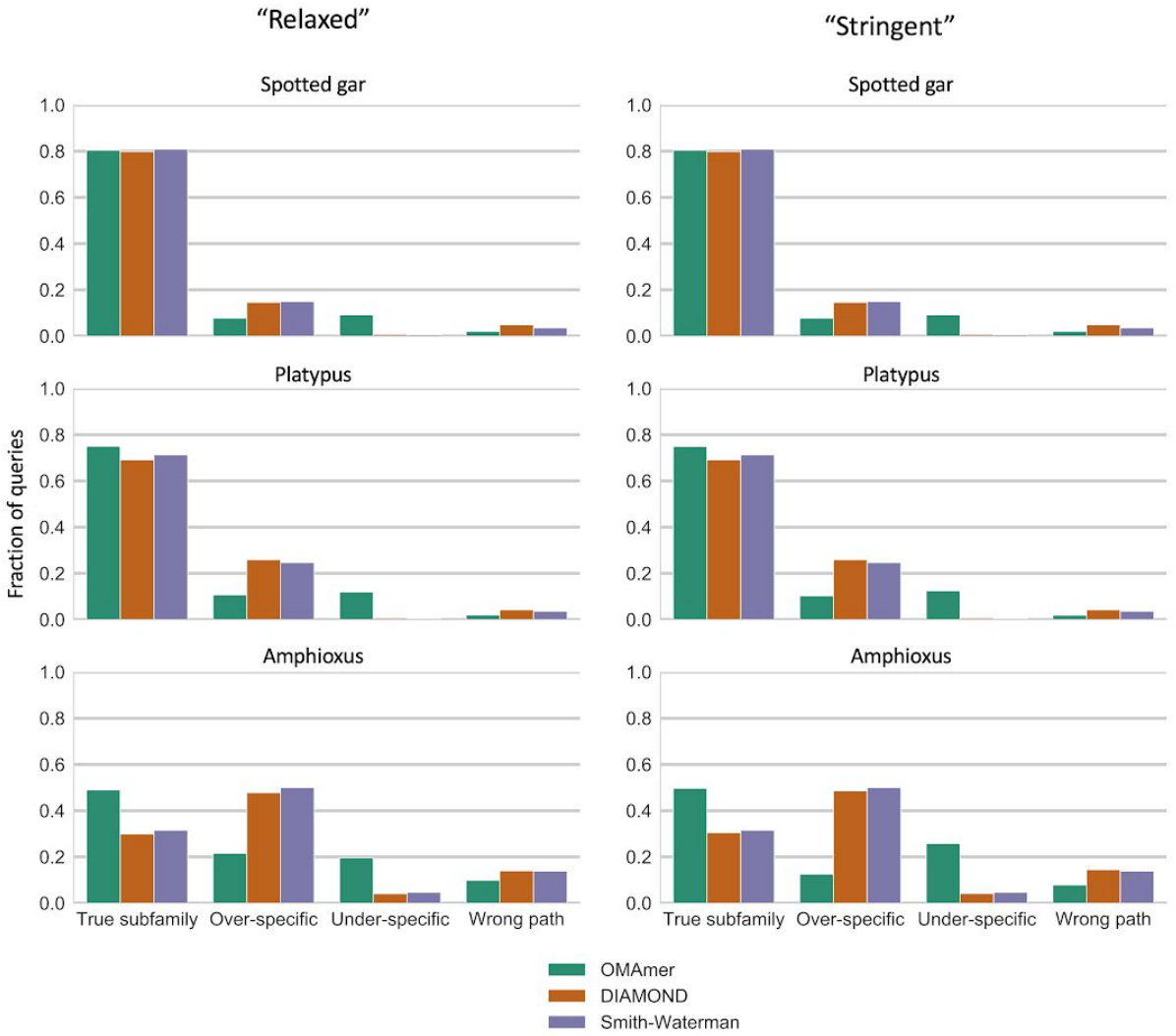
Supp. Fig. 2. Comparison of family assignments between OMAmer and DIAMOND across negative datasets. Each curve displays the range of trade-offs between precision and recall when varying a score threshold. The curves labeled *Bacteria* refer to analyses using bacteria-specific sequences as negatives whereas those labeled *Random* refer to using random sequences as negatives. Crosses indicate the location of F1max values.



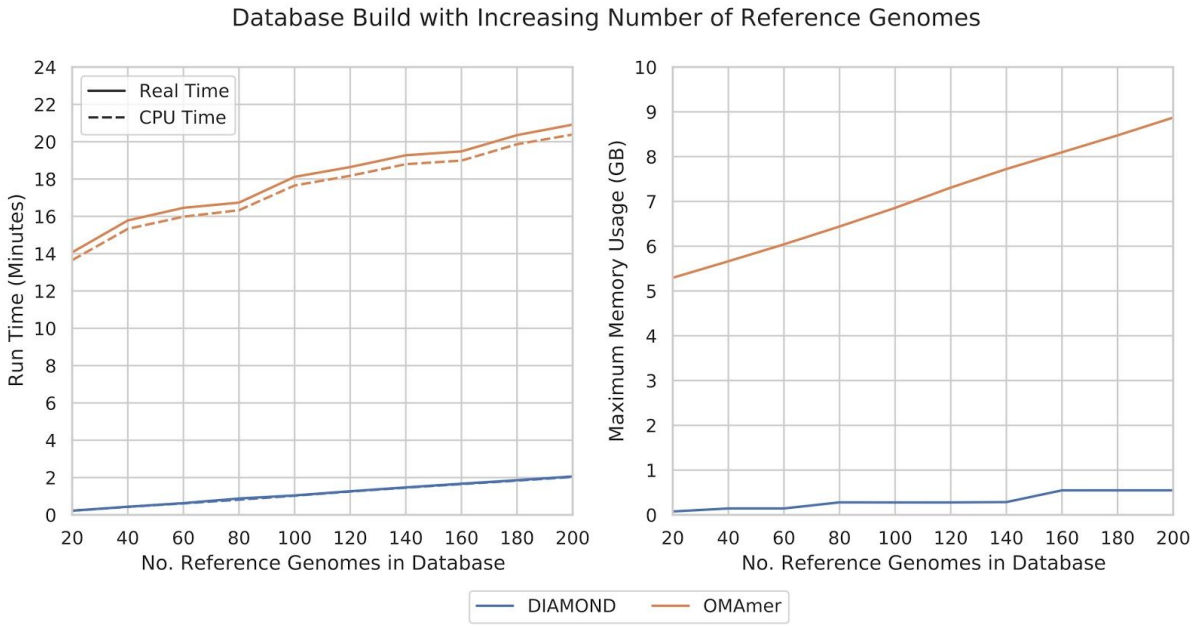
Supp. Fig. 3. Comparison of subfamily assignments with OMAmer and by closest sequence (Smith-Waterman and DIAMOND). Each curve displays the range of trade-offs between precision and recall when varying a score threshold. These results were computed using a stringent validation procedure. $F1_{max}$ values are annotated with their locations indicated by crosses.



Supp. Fig. 4. "Over-specific" configurations were especially well dealt with by OMAmer. Each curve displays the range of trade-offs between precision and recall when varying a score threshold. They were computed by breaking down queries by closest sequence configurations as in panel A, before the validation procedure itself (stringent one). Crosses indicate the location of $F1_{max}$ values.



Supp. Figure 5. Comparison of placement configurations between subfamily assignments of OMAmer, DIAMOND and Smith-Waterman and OMAmer at $F1_{max}$. "Relaxed" and "stringent" refers to validation procedures (see *Methods*).



Supp. Figure 6. Run time (left) and maximum memory usage (right) during database build for DIAMOND and OMAmer. Whilst OMAmer is slower due to the increased pre-processing to enable constant lookup time, the increase in time is sublinear with the number of reference genomes in the resulting database.

Supp. Table 1. Species used as queries in accuracy experiments.

Species	**Scientific name**	**LCA clade**	**Divergence age (mya)**	**Genome N50 (kb)**
Spotted Gar	*Lepisosteus oculatus*	*Neopterygii*	320 (Betancur-R *et al.*, 2017)	68 (Ensembl LepOcu1 assembly)
Platypus	*Ornithorhynchus anatinus*	*Mammalia*	250 (Upham *et al.*, 2019)	612 (Ensembl OANA5 assembly)
Amphioxus	*Branchiostoma floridae*	*Chordata*	600 (Peterson and Eernisse, 2016)	2600 (Putnam *et al.*, 2008)

Betancur-R,R. *et al.* (2017) Phylogenetic classification of bony fishes. *BMC Evol. Biol.*, **17**, 162.
Upham,N.S. *et al.* (2019) Inferring the mammal tree: Species-level sets of phylogenies for questions in ecology, evolution, and conservation. *PLoS Biol.*, **17**, e3000494.
Peterson,K.J. and Eernisse,D.J. (2016) The phylogeny, evolutionary developmental biology, and paleobiology of the Deuterostomia: 25 years of new techniques, new discoveries, and new ideas. *Org. Divers. Evol.*, **16**, 401–418.
Putnam,N.H. *et al.* (2008) The amphioxus genome and the evolution of the chordate karyotype. *Nature*, **453**, 1064–1071.