Sensitive Protein Alignments at Tree-of-Life Scale Using DIAMOND

Dissertation

der Mathematisch-Naturwissenschaftlichen Fakultät
der Eberhard Karls Universität Tübingen
zur Erlangung des Grades eines
Doktors der Naturwissenschaften
(Dr. rer. nat.)

vorgelegt von
Dipl.-Inf. Benjamin Buchfink
aus Backnang

Tübingen
2023

Gedruckt mit Genehmigung der Mathematisch-Naturwissenschaftlichen Fakultät der Eberhard Karls Universität Tübingen.

Tag der mündlichen Qualifikation:	11.07.2023
Dekan:	Prof. Dr. Thilo Stehle
1. Berichterstatter:	Prof. Dr. Detlef Weigel
2. Berichterstatter:	Prof. Dr. Sven Nahnsen

Table of Contents

Zusammenfassung

Paarweises Alignment von Proteinsequenzen ist eine grundlegende Analysemethode der Bioinformatik. Seit dem Aufkommen von Next-Generation-Sequencing haben die anfallenden Sequenzdatenmengen einem exponentiellen Wachstum unterlegen. Insbesondere im Rahmen von Metagenomik-Studien wird DNA in der Größenordnung von Terabasen aus der Umwelt oder Wirtsorganismen sequenziert, und sensitives Sequenzalignment auf Proteinebene ist eine Schlüsselmethode, um die enorme Vielfalt mikrobiellen Lebens zu erforschen. Im Rahmen evolutionärer Studien ermöglicht es das paarweise Proteinalignment, entfernt verwandte Sequenzen zu erkennen, die über Milliarden von Jahren voneinander divergiert sind. Als Standardsoftware für Proteinalignment gilt NCBI BLAST, dessen rechnerische Leistungsfähigkeit nicht ausreichend ist, um gegenwärtig existierenden Datenmengen gerecht zu werden, und eine Rechenzeit von Jahrzehnten zur Auswertung großer Datensätze benötigen würde. Dies hängt mit der grundsätzlichen algorithmischen Schwere des Alignmentproblems auf Proteinen mit bis zu 20% Sequenzidentität hinunter ab, für das keine effizienten Lösungen bekannt sind. In dieser Arbeit wird der Proteinaligner DIAMOND vorgestellt, dessen erste Version bis zu 20.000-fach schneller als BLAST für das Alignment von Short Reads ist, sowie 650 bis 2.500-fach schneller in seinem sensitiven Modus, der bereits für ein breites Anwendungsspektrum geeignet ist. Darüber hinaus wird die Weiterentwicklung DIAMOND v2 vorgestellt, die einen Speedup von 80 bis 360 gegenüber BLAST bei vergleichbarer Sensitivität erreicht, sowie von 12 bis 15 gegenüber der Konkurrenzsoftware MMseqs2 und 6 bis 8 gegenüber der ersten Version von DIAMOND. Algorithmisch zentrale Aspekte sind zum einen mehrfache spaced Seeds, mit denen die Spezifität des Seedings entscheidend verbessert wird. Zum anderen ist dies die doppelte Indexierung von Query- und Targetsequenzen, die eine effiziente Verarbeitung der Seedtreffer unter verbesserter Ausnutzung der CPU-Caches ermöglicht. Zuletzt ist die konsequente Optimierung der Verabeitungspipeline durch SIMD-Instruktionen und geeignete Heuristiken wichtig, um die Performance wesentlich über das Niveau elementarer Ansätze hinaus zu steigern.

Summary

Pairwise alignment of protein sequences is a fundamental method of bioinformatics. Since the advent of Next Generation Sequencing, the arising quantities of sequence data have grown exponentially. In the context of metagenomics studies, DNA in the order of terabases is sequenced from the environment or host organisms, and sensitive protein alignment is a key method to investigate the enormous diversity of microbial life. In the context of evolutionary studies, pairwise protein alignment enables the detection of distantly related sequences that have diverged over billions of years. The standard software for protein alignment is NCBI BLAST, whose computational performance is not sufficient to process currently existing quantities of data, requiring decades to analyse large datasets. This is due to the fundamental algorithmic hardness of the alignment problem on proteins down to 20% sequence identity, for which no efficient solutions are known. In this work I present the protein aligner DIAMOND, whose first version was up to 20,000-fold faster than BLAST for short read alignment, and 650 to 2,500-fold faster in its sensitive mode that was already suitable for a broad range of applications. Furthermore, I present the enhanced version DIAMOND v2 which achieves a speedup of 80 to 360 vs BLAST at comparable sensitivity, a speedup of 12 to 15 vs the competitor tool MMseqs2, and a speedup of 6 to 8 vs the first version of DIAMOND. Central algorithmic aspects are multiple spaced seeds that drastically improve the specificity of seeding, and furthermore the double indexing of query and target sequences allowing the efficient processing of seed hits with improved use of the CPU caches. Lastly, the consequent optimisation of the processing pipeline using SIMD instructions and suitable heuristics is important to increase the performance substantially beyond the level of elementary approaches.

Acknowledgements

Thesis advisors

Hajk-Georg Drost

Daniel Huson

Thesis committee

Andreas Dräger

Sven Nahnsen

Nico Pfeifer

Detlef Weigel

Co-authors

Basem Al-Shayeb, Tomer Altman, Haim Ashkenazy, Artem Babaian, Jillian F. Banfield, Pierre Barbera, Rayan Chikhi, Fernando Dias, Robert Edgar, Andreas Grigorjew, Artur Gynter, John Kennedy, Anton Korobeynikov, Victor Lin, Dan Lohr, Dmitry Meleshko, Gherman Novakovsky, Marcos de la Peña, Klaus Reuter, Jeff Taylor, Alexandru Tomescu, Chao Xie

Drost lab members

Celine Behr, Erica Dinatale, Nikola Kalábová, Jasmin Katz, Dimitrios Koutsogiannis, Jaruwatana Lotharukpong, Lukas Maischak, Sergio Vasquez

Huson lab members

Sina Beier, Mohamed El Hadidi, Anna Gorska, Hans-Joachim Ruscheweyh, Rewati Tappu

Collaborators

Patrick Ettenhuber

Tech support

Joachim Sieler

1. Introduction

1.1 Overview

Sequence alignment is a fundamental operation in bioinformatics to identify evolutionarily related sequences and variation among organisms, supporting life scientists across a wide range of disciplines from agriculture to zoology. While the signal of DNA sequence comparisons gets blurry at around 60% sequence identity, the pairwise alignment of primary protein structures can detect sequences of common ancestry that have diverged over billions of years of evolution.

Since the advent of Next Generation Sequencing, the field of metagenomics has made heavy use of large-scale protein sequence comparisons to explore the vast biodiversity of microbial life that may comprise up to one trillion species (Locey and Lennon 2016). Efforts such as the Human Microbiome Project ("Human Microbiome Project" 2013) and the Tara Ocean Foundation (Sunagawa et al. 2015) have sequenced and studied terabases of DNA in order to advance our understanding of microbiology and drive biotechnological and medical applications. The bacterial communities inhabiting the human gut are known to have a strong effect on our overall health, and an imbalanced gut microbiome has been implicated as a factor in various diseases, including rheumatoid arthritis, diabetes, asthma, inflammatory bowel disease, chronic kidney disease, and even neurological disorders by the means of the "gut-brain axis" (Vijay and Valdes 2022; Mayer, Tillisch, and Gupta 2015). Clinical metagenomic sequencing is employed for infectious disease diagnostics, antibiotic resistance prediction, outbreak surveillance and epidemiology, human host response analysis, and oncology to study viruses associated with cancer (Chiu and Miller 2019).

Environmental habitats like soil or the ocean are abundantly populated by microorganisms which play an essential role in the nutrient cycles of these ecosystems and the health of our planet. A single gram of soil may contain billions of bacterial cells of 4,000 to 50,000 distinct species (Raynaud and Nunan 2014). Marine environments are dominated by microbes with respect to biomass and metabolic activity from photosynthesis to biodegradation (Kennedy et al. 2010). Metagenomic sequencing allows us to investigate the functioning of these communities comprised

of mostly unculturable organisms. Possible applications include the improvement of soil health and fertility for agricultural use, the bioremediation of contaminated soils and the screening of soil environments for molecules relevant to industrial processes and products such as cellulases, lipases or xylanases (Sabale, Suryawanshi, and Krishnaraj 2020). Oceanic microbes are a largely unexplored source of pharmaceutically relevant compounds that may exhibit anti-microbial, anti-viral, anti-tumour, anti-parasitic, anti-inflammation, or neurological activity (Trindade et al. 2015).

Beyond microorganisms, future ambitions include the Earth Biogenome Project (Lewin et al. 2018) which aims to sequence 1.8 million eukaryotic species within this decade. Large-scale comparative genomics and annotation will be the computational groundwork to make use of this dataset. Alignment sensitivity is crucial for comparing genomes across large evolutionary distances, as a large proportion of orthologous relations in the eukaryotic domain are highly diverged on the sequence level (Ruperti et al. 2023). The project is expected to advance a wide range of biotechnological and medical applications, such as drug discovery in marine eukaryotes or plants, the production of lignocellulose-derived biofuels (Lam et al. 2021), the cleaning of waste streams, the replacement of synthetic industrial materials by biology-based alternatives like bioplastics or spider silk (Ramezaniaghdam, Nahdi, and Reski 2022; Rosenboom, Langer, and Traverso 2022), or the production of food additives for meat substitutes (Voigt 2020; Lewin et al. 2018). Crops could be engineered that have higher yield, more nutritional value, are more resistant to heat, drought, or pathogens, and remove more CO_2 from the atmosphere and store it in the ground (Exposito-Alonso et al. 2020).

The Earth Biogenome dataset will also be an unprecedented opportunity to study questions of evolution on a large interspecific scale, and sensitive alignment in protein space will relate organisms and their genes across the tree of life. Studying the evolution of proteins can furthermore yield insights about their structure, function, and interactions. Gene age inference has been an important tool in studying developmental processes (Domazet-Lošo and Tautz 2010) (Quint et al. 2012). The emergence of cancer-associated genes has been strongly linked to the evolution of multicellularity and the origin of the first cell (Domazet-Loso and Tautz 2010), and the integration of phylogenetic and gene expression data has been used to study the

evolutionary patterns of cancer-associated pathways (L. Zhang et al. 2019). The recent breakthrough of AlphaFold2 (Jumper et al. 2021) demonstrated the use of evolutionary information at a large scale, where coevolving loci identified on the basis of multiple sequence alignments of related proteins enable the in-silico prediction of tertiary structure. The first step of the AlphaFold2 prediction is aligning the query sequence against MGnify (Mitchell et al. 2019) and the “Big Fantastic Database”, clustered databases each containing more than two billion protein sequences.

The insights gained from sensitive sequence alignment in protein space come at the expense of the computational effort needed to generate them. The standard tool for this task has been considered NCBI BLAST (Altschul et al. 1990) for more than 30 years. Given current database sizes (NCBI non-redundant, 514 million sequences as of November 2022), BLAST aligns proteins at a speed of 2.1

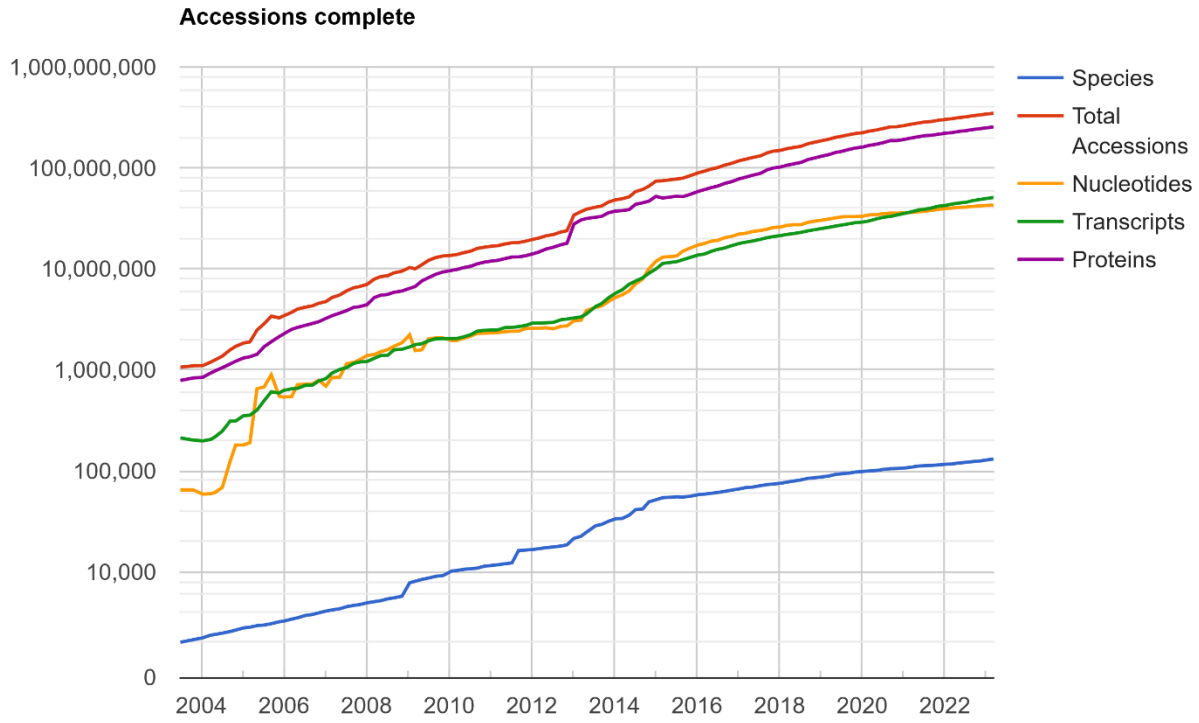


Figure 1. The growth of NCBI RefSeq since 2004 at logarithmic scale, from the U.S. National Center for Biotechnology Information.

sequences/CPU hour[1], rendering it impractical for large-scale computations. For example, analysing the 40 million protein sequences of an ocean microbial reference catalogue created from Tara Oceans samples (Sunagawa et al. 2015) in this way

[1] Measured using a sample of 3,000 sequences from UniRef50 on an Intel Xeon Platinum 8360Y system with 64 cores.

would run for 34 years on a 64-core server. Meanwhile, ongoing sequencing efforts across the life sciences community have led to growth of publicly available data at a near-exponential rate **(Fig. 1)**. This is still only the beginning of the possibilities and ambitions that exist, representing only a small fraction of the planet's biodiversity. Computational efficiency is a key factor in addressing the challenges of biological big data which would otherwise overwhelm our research infrastructures and analytics pipelines.

Considerable effort has been spent on developing faster BLAST alternatives (Kent 2002; Edgar 2010; Kiełbasa et al. 2011; Ye, Choi, and Tang 2011; Zhao, Tang, and Ye 2012; Hauswedell, Singer, and Reinert 2014; Hauser, Steinegger, and Söding 2016), but the problem of protein alignment remains computationally hard and further improvements are necessary to keep up with the pace of incoming data. Additionally, these efforts were focused on gaining speed at the expense of sacrificing sensitivity, mostly disregarding the "twilight zone" of protein alignment at below 35% sequence identity (Rost 1999). Meanwhile, fully replacing BLAST with a substantially faster tool for detecting distant similarities down to 20% pairwise identity has remained a longstanding unsolved problem. The algorithmic hardness of this problem is particularly clear by the fact that even exhaustive Smith Waterman searches can be computed at a similar speed than BLAST (Rognes 2011), exposing a lack of efficient solutions.

In this work I will introduce the protein sequence aligner DIAMOND (Benjamin Buchfink, Xie, and Huson 2015; Benjamin Buchfink, Reuter, and Drost 2021) and discuss algorithms and code to solve the pairwise protein alignment problem more efficiently than possible with previous state-of-the-art software. I will show considerable advancements by combining algorithm design with an understanding of the underlying hardware. Fully making use of the capabilities of modern computing architectures and upholding technical code quality will result in actionable solutions to support the life science community.

1.2 Basics of protein alignment

The primary protein structure is defined by the linear chain of amino acid residues. The standard genetic code maps the coding space defined by 64 distinct codon triplets to the standard amino acid alphabet of size 20. Defining a pairwise

alignment of two sequences requires a measure of similarity for the amino acid residues, which is typically constructed based on a set of manually curated alignments (Henikoff and Henikoff 1992). From these alignments we can observe the target frequencies $(q_{i,j})_{i,j\in[1;20]}$ of substitutions of amino acid *i* by amino acid *j* as well as the background frequencies $(p_i)_{i\in[1;20]}$ of the occurrence of amino acid *i*. We then define a 20x20 scoring matrix whose entries are defined as

$$s_{i,j} = \frac{1}{\lambda}\ln\frac{q_{i,j}}{p_i p_j} \qquad i,j \in [1;20]$$

with λ being a scaling factor and the $s_{i,j}$ typically being rounded to the next integer. This *log-odds ratio* definition allows us to discretise the probability ratio between the foreground model (corresponding to alignments of homologous sequences) and the background model (corresponding to spurious similarities) and to quantify the statistical significance of alignments. The widely used BLOSUM substitution matrices were constructed in several flavours (e.g., BLOSUM62) denoting that only sequences of less than 62% pairwise identity were used for the initial seed alignments.

For the purpose of this thesis, I will only consider local alignments, meaning that only pairs of segments of the two sequences are aligned. In addition to the substitution scores given by the scoring matrix, we will manually define a gap open penalty *Q* and gap extension penalty *R* for scoring alignments with gaps. The optimal local alignment with affine-linear gap penalties of a query sequence *q* of length *m* and a database sequence *d* of length *n* is defined as the result of the Smith-Waterman-Gotoh algorithm (Gotoh 1982):

$$H_{i,j} = \begin{cases} \max\begin{cases} H_{i-1,j-1} + s_{q_i,d_j} \\ E_{i,j} \\ F_{i,j} \\ 0 \end{cases} \Bigg| \, i,j > 0 \\ 0, otherwise \end{cases}$$

$$E_{i,j} = \begin{cases} \max\begin{cases} H_{i,j-1} - Q \\ E_{i,j-1} - R \end{cases} \Big| \, j > 0 \\ 0, otherwise \end{cases}$$

$$F_{i,j} = \begin{cases} \max\begin{cases} H_{i-1,j} - Q \\ F_{i-1,j} - R \end{cases} \Big| \, i > 0 \\ 0, otherwise \end{cases}$$

$$S = \max_{i\in[1,m],j\in[1,n]} H_{i,j}$$

The final score *S* represents the score of the optimal local alignment between the sequences. This algorithm is a classic example of dynamic programming (DP) where the solution of a problem is recursively constructed based on the solution of smaller subproblems (here: subsequences). The concept of a single optimal local alignment can be generalized to that of a HSP (high-scoring segment pair) **(Fig. 2)** as any alignment of a pair of segments of the sequences whose score cannot be increased further by extending the segments in any direction, accounting for the fact that two

```
>hemoglobin, theta T2 [Mus musculus]
Sequence ID: NP_778165.1 Length: 145
Range 1: 4 to 135

Score:75.5 bits(184), Expect:7e-15,
Method:Compositional matrix adjust.,
Identities:47/138(34%), Positives:70/138(50%), Gaps:8/138(5%)

Query  5    TPEEKSAVTALWGKV--NVDEVGGEALGRLLVVYPWTQRFFESFGDLSTPDAVMGNPKVK  62
            + +++  V ALW K+  NV     EAL R  V +P T+ +F         D   G+ +VK
Sbjct  4    SQDDQWLVLALWKKMGSNVGIYTTEALERTFVAFPSTKTYFPHL------DLRPGSSQVK  57

Query  63   AHGKKVLGAFSDGLAHLDNLKGTFATLSELHCDKLHVDPENFRLLGNVLVCVLAHHFGKE  122
            AH +KV  A +    HLD+L  + + LS+LH  KL VDP NF+     L+  LA H+  +
Sbjct  58   AHAQKVADALTLATQHLDDLPASLSALSDLHAHKLCVDPANFQFFSCCLLVTLARHYPGD  117

Query  123  FTPPVQAAYQKVVAGVAN  140
            F+    A     V+G  +
Sbjct  118  FSGEKDACLLGHVSGSCD  135
```

***Figure 2**. A HSP computed by NCBI BLAST for the human hemoglobin subunit beta (NP_000509.1).*

sequences may in general share multiple independent regions that are locally similar.

Assessing the statistical significance of an alignment is typically done based on the e-value (Karlin and Altschul 1990; Altschul et al. 1990):

$$evalue(S) = Kmne^{-\lambda S}$$

K is a statistical parameter of the scoring matrix, *m* is the length of the query sequence, and *n* is the total length of the search database (in amino acid residues). The expected value of the number of HSPs of score *S* or higher occurring for random query and database sequences of lengths *m* and *n* is a powerful statistical tool to distinguish between true alignments resulting from homology and spurious similarities that inevitably occur when searching through large sequence databases. Tools like BLAST and DIAMOND will by default filter their alignment output by an e-value threshold to guide users in considering only hits that are statistically significant.

1.3 Prior work

NCBI BLAST (Altschul et al. 1990, 1997) is (to the best of my knowledge) the most highly cited bioinformatics tool of all time and still considered the “gold

standard" tool for pairwise protein alignment. Remarkably, BLAST has not been fully replaced by any other tool to date, despite the fact that its core algorithm has not changed for 30 years and the heavy computational burden it imposes. This speaks to the hardness of the problem it solves and its seeming resistance to better algorithmic solutions.

BLAST was one of the first tools to pioneer the seed-and-extend approach of computing alignments as opposed to rigorous Smith-Waterman searches. Seeding here refers to matching short exact subsequences, which are then extended to longer, inexact alignments. The core idea of BLAST is to extract all 3-mers[2] (seeds) from the query sequence and store them in an index data structure along with their neighbouring 3-mers. Two 3-mers are considered neighbours if their alignment score according to the substitution matrix is above a certain threshold. The BLAST algorithm scans linearly over the database sequences, looks up the 3-mers in the query index and triggers an extension if two 3-mer hits are found on the same diagonal[3] within a 40-letter window. If an ungapped extension at the seed hit location satisfies a minimum score threshold, a gapped alignment anchored at the seed hit is computed using the X-drop approach, meaning that the computation of paths in the dynamic programming matrix will be abandoned if the score drops below a certain difference to the maximum along the path.

Using the default thresholds, the BLAST algorithm comes close to the optimal sensitivity achievable by a full Smith-Waterman computation. The seeding strategy in this setup however produces many orders of magnitude more false positives hits than it finds significant alignments. For comparing two random sequences of length 300, the expected value of the number of (ungapped) extensions triggered by the BLAST algorithm is approximately 7.4, making it barely more efficient than a brute-force all-vs-all comparison[4]. In fact, it has been shown that a well-optimized and vectorized Smith-Waterman search can almost reach the search speed of BLAST (Rognes 2011). While most other alignment tools in use can be fast due to effective filtration of the search space by sufficiently long and specific seeds, the low filtration efficiency of a seeding sensitive enough to detect pairwise similarities down to 20%

[2] A *k*-mer is defined as a subsequence of *k* consecutive letters.

[3] For query position *i* and target position *j*, the diagonal is defined as *i* - *j*.

[4] An estimate for the expected value was empirically determined by running BLAST on randomly generated protein sequences and having it output the number of extensions computed (unpublished).

sequence identity precludes a fast solution of this problem. Consequently, BLAST has not been fully replaced by any other software over more than three decades, and the life sciences community is still investing considerable resources into operating it every day.

FASTA (Pearson and Lipman 1988) was an even earlier tool that uses short *k*-mers (usually *k*=2 for proteins) to filter sequences for regions of interest and determine initial diagonals as anchor points for extension.

Many faster BLAST alternatives were proposed over the years. Since the full search sensitivity down to the twilight zone is not necessary for many downstream applications, it is an attractive option to sacrifice sensitivity to gain speed. This is mostly accomplished by choosing longer, more specific seeds in combination with a basic design that most alignment tools follow:

1. Index the seeds of the reference database in some suitable index data structure e.g. hash tables, suffix arrays, or the FM-Index (Ferragina and Manzini 2000).
2. Read linearly over the query sequences, extract seeds and look them up in the index.
3. Compute gapped extensions at the seed hit locations.

Tools that fall into this category include BLAT (Kent 2002), USEARCH (Edgar 2010), LAST (Kiełbasa et al. 2011), RAPSearch (Ye, Choi, and Tang 2011), SANS (Koskinen and Holm 2012), RAPSearch2 (Zhao, Tang, and Ye 2012), SANSparallel (Somervuo and Holm 2015), SWORD (Vaser, Pavlović, and Šikić 2016), MMseqs (Hauser, Steinegger, and Söding 2016), MMseqs2 (Steinegger and Söding 2017) and TOPAZ (Medlar and Holm 2018).

BLAT uses *k*-mers of length of 3-7 that are indexed in a table or hash table, also requiring multiple hits per target sequence to increase specificity. **USEARCH** uses a similar strategy of ranking target sequences based on the counts of seed hits and only evaluating the most promising candidates. **LAST** leverages the suffix array data structure to use seeds of variable length, limiting the search space by increasing the seed length for each lookup position until the hit count is below a user-defined threshold. **RAPSearch** uses the suffix array in a similar way as LAST combined with a reduced alphabet, while **RAPSearch2** indexes 6-mers in a hash table. **SANS** (“suffix array neighbourhood search”) also uses a database suffix array to scan for

seeds that inexactly match the query, so do its successor tools **SANSparallel** and **TOPAZ**. **SWORD** indexes database *k*-mers of length 3-5 in a hash table and reportedly achieves a speedup of 8-16 over BLAST with comparable sensitivity. Similar to BLAST, it also indexes neighbouring seeds and requires at least two seed hits on the same diagonal for triggering an extension. Speed is gained by ranking the target sequences based on their seed hits for each query and only considering a certain number of best candidates. **MMseqs** indexes database 6-mers in a hash table and ranks target sequences by their prefilter score derived from seed hits, while the search sensitivity is controlled by also conducting inexact seed lookups. **MMseqs2** simplified this idea by scanning for two adjacent seed hits on the same diagonal ("double-match criterion"), which can in practice be evaluated faster due to a more cache-friendly implementation. It offers a wide range of sensitivity settings and reportedly comes close to the sensitivity of BLAST at a 36-fold speedup.

Tools that deviate from this standard algorithm design include **PLAST** (Nguyen and Lavenier 2009) which proposed the use of two index data structures for both queries and targets and computing alignments in seed-by-seed order, reportedly achieving a speedup of 3-6 vs BLAST with similar sensitivity. **PSimScan** (Kaznadzey et al. 2013) is more similar to the original BLAST algorithm as it indexes batches of queries instead of the database using 5-mers. It reportedly is 5-100 times faster than BLAST but does not match its full sensitivity. **Lambda** (Hauswedell, Singer, and Reinert 2014) also makes use of double-indexing, indexing the query seeds in a trie and the targets in a suffix array. **GHOSTX** (Suzuki et al. 2014) uses double-indexing with suffix arrays and reportedly achieved a speedup of 131 to 165-fold of BLASTX for short read alignment only. **PAUDA** (Huson and Xie 2014) is not an alignment tool in itself, but solves protein alignment by reducing the protein sequences to a four-letter alphabet and using the Bowtie2 DNA aligner (Langmead and Salzberg 2012) to align the reduced sequences.

The only of these tools that strive for search sensitivity close to BLAST are FASTA, PLAST, SWORD and MMseqs2. Most of the time, better performance is paid for by reduced sensitivity. Despite the considerable effort that was invested into

developing all of these tools, wide adoption by the research community[5] has only been achieved by FASTA, BLAST, BLAT, USEARCH, LAST and MMseqs2.

[5] Defined here as >500 literature citations according to Google Scholar.

2. Algorithms

2.1 DIAMOND version 0

In this subchapter I will describe the algorithms of the original DIAMOND software as published in November 2014 (Benjamin Buchfink, Xie, and Huson 2015). I will refer to this software as DIAMOND v0 in accordance with the public release version numbers. At the time, Next Generation Sequencing (NGS) technology was relatively new and heavily employed in metagenomics studies to produce large quantities of sequencing reads from both environmental and host-associated samples. At the same time, researchers lacked the means for efficient computational analysis of such vast amounts of data. For one such study conducted by the U.S. Department of Energy Joint Genome Institute (Mackelprang et al. 2011), the authors produced 190 million Illumina reads from Alaskan permafrost and computed for two weeks on a supercomputer (totalling 800,000 CPU hours) to annotate these reads against the KEGG database (Ogata et al. 1999) using BLASTX (Jansson 2011). Spurred by this analytics bottleneck was the development of the DIAMOND[6] protein aligner, among other tools.

2.1.1 Alphabet reduction

An alphabet reduction maps the 20 standard amino acid residues to a smaller alphabet by grouping similar residues (Edgar 2004). Reducing the protein sequences in this way prior to extracting seeds is commonly used by many alignment tools, as it is known to improve the sensitivity/specificity trade-off (Ye, Choi, and Tang 2011). The original version of DIAMOND v0 used a hand-crafted reduction of size 11, while later versions used the Murphy10 reduction (Murphy, Wallqvist, and Levy 2000).

2.1.2 Spaced seeds

Spaced seeds are extracted out of a sequence according to a pattern of "match" and "don't care" positions (Ma, Tromp, and Li 2002) (e.g. "111101110111"), skipping over some letters as opposed to a simple *k*-mer consisting of *k* consecutive letters in the sequence. They are well known to exhibit better sensitivity for the same

[6] An acronym for "Double Indexed AlignMent Of NGS Data", name conceived by Daniel Huson.

value of *k* than a corresponding consecutive seed (here *k* is the number of match/1 positions in the seed, also denoted as the weight of the spaced seed). What makes them most interesting is the fact that a set of such patterns or seed shapes can be designed in a way to complement each other (Ilie and Ilie 2007; M. Li et al. 2004). This effect can be quantified as shown by **(Fig. 3).** We observe that the sensitivity of one spaced seed of weight 11 is roughly equivalent to the sensitivity of a set of two spaced seeds of weight 12, or four spaced seeds of weight 13, and so on.

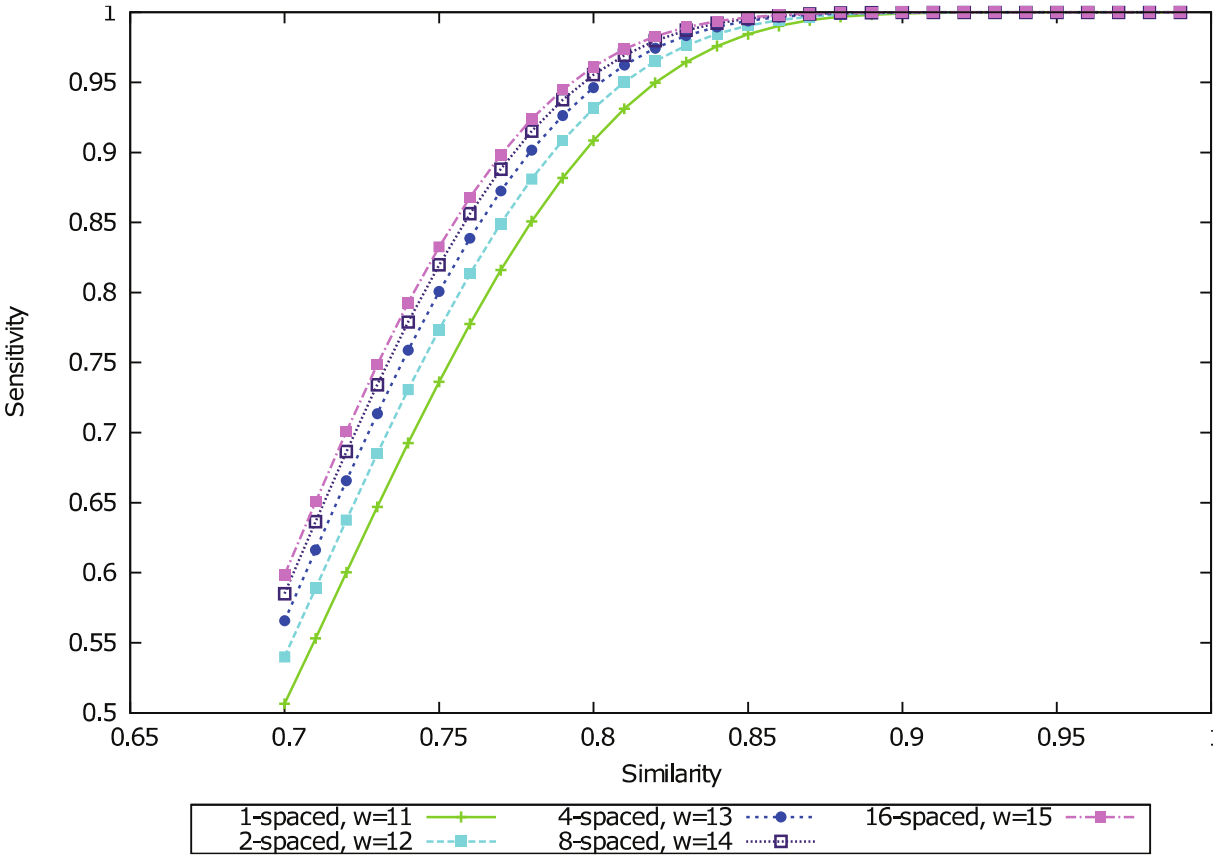


Figure 3.The sensitivity of sets of 1 to 16 spaced seeds with increasing weight depending on the similarity of a homologous region of length 70, from (Ilie et al. 2011).

Doubling the number of spaced seeds and simultaneously increasing their weight by one thus maintains equal sensitivity, which is a powerful property of multiple spaced seeds: while twice the number of seeds will double the computational cost, an increase of the weight by one is expected to reduce the number of false positive hits by a factor of roughly the size of the alphabet. Thus, multiple spaced seeds are a key instrument to increase the filtration efficiency of the seeding and speed up sensitive protein alignment where the biggest challenge is to process an overwhelming number of random hits. The initial version of DIAMOND v0 used a set of 4 seed

shapes of weight 12 for its default mode and 16 seed shapes of weight 8 for its sensitive mode.

2.1.3 Double indexing

The basic seed-and-extend paradigm is implemented most of the time as outlined in the last chapter: an index is built for the database seeds, the queries are scanned linearly, the seeds are looked up in the index, and extensions are computed at the seed hit locations. This approach works well as long as seeds can be chosen sufficiently long and specific such that most of the search space is already eliminated at this stage, leaving only a few candidate locations for further evaluation. This is the case for typical read mappers designed to map reads against a reference genome of the same organism.

In contrast, sensitive protein alignment across the tree of life presents a different problem class. While we did not venture to achieve the sensitivity of BLAST when creating DIAMOND v0, the sensitive alignment mode attempts to provide good sensitivity for hits of 40% sequence identity and above, which is already much more demanding than the typical task of a read mapper. When DIAMOND operates in sensitive alignment mode, it may need to process 200 million seed hits to align a single query protein sequence, with only 1 in 4,000 seed hits yielding a significant alignment[7]. When dealing with false positive hits of this magnitude, memory latency incurred from querying the seed index becomes the limiting factor of the computation. The standard seed lookup approach as described above exhibits a quasi-random memory access pattern to an index data structure and sequence database many gigabytes in size, with little chance of successful utilization of the CPU cache memories. Since main memory loads may have latencies of hundreds of clock cycles on modern CPUs[8], this approach is poorly adapted to modern hardware architectures.

[7] Numbers empirically determined by aligning a sample of 1,000 sequences from UniRef50 against the NCBI non-redundant database (514 million entries), unpublished.

[8] For example, the latency was measured at 105 ns on the AMD 3rd generation EPYC Milan architecture (https://www.anandtech.com/show/16529/amd-epyc-milan-review/4).

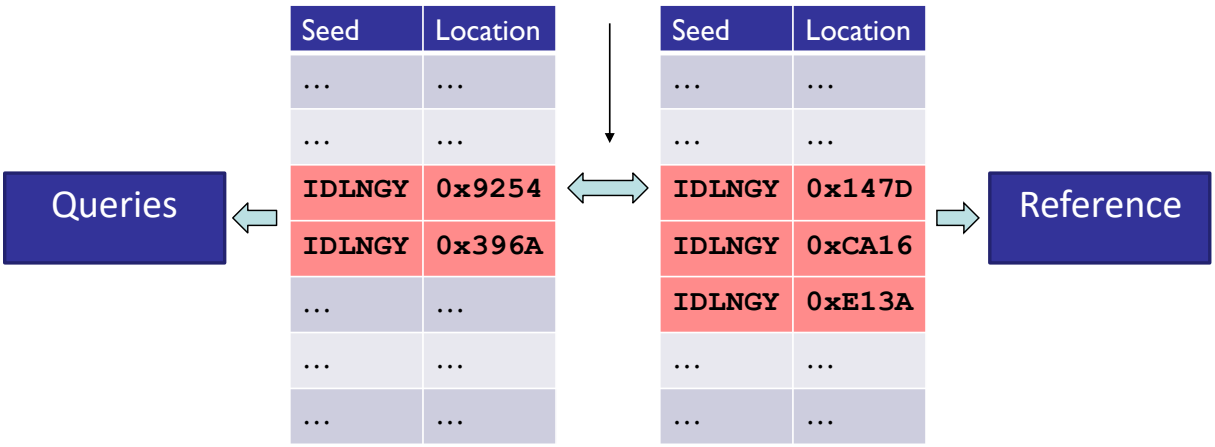


Figure 4. Double indexing builds a pair of indexes for the queries and targets and processes them together in seed-by-seed order.

The idea of double-indexing attempts to restructure this computation and determine and process seed hits in a cache-aware manner. DIAMOND v0 builds two lists of (seed, location)-pairs for the query and target sequences respectively. These lists are then sorted on the seed and processed in seed-by-seed order. This approach corresponds to the classic sort-merge join algorithm known from database systems and was first suggested (to my knowledge) in the context of sequence alignment by PLAST (Nguyen and Lavenier 2009). The algorithm reads the index data structures linearly, which can be efficiently handled by the hardware prefetcher, and completely avoids a random memory access pattern with respect to the index. Further, when evaluating the hits of one particular seed, all query and target sequences containing it are processed together. As seeds are chosen reasonably long in a multiple spaced seeds configuration, this sequence data associated with a single seed will usually be limited in size and can therefore be cached and reused by the CPU for the full all-vs-all comparison of the query and target locations **(Fig. 4)**.

2.1.4 Hamming distance filter

As outlined before, sensitive protein alignment is dominated by the occurrence of false positive seed hits, and their rapid elimination is crucial to achieving high throughput. Full extension by the means of Smith Waterman of every seed hit candidate location is prohibitively expensive, and fast heuristics are essential. Before evaluating a seed hit by gapped extension, we will at least require that an ungapped extension at the hit location satisfies a certain minimum threshold score. Even this computation is relatively expensive however, since for each pair of letters to be scored, a separate load from the scoring matrix is required. DIAMOND v0 therefore

disregards amino acid substitution scores in the first stage of evaluating seed hits, and instead counts the number of identical letters over a 48-letter window aligned at the seed hit location. This count is required to be above a certain threshold parameter, which is a fixed constant to be empirically determined in order to reach a given target sensitivity[9]. The hamming distance operation can be efficiently computed using vector instruction sets, loading 16 (for SSE2) or 32 (for AVX2) sequence letters into one SIMD register. My implementation uses the `pcmpeqb` instruction to test for equality, the `pmovmskb` instruction to move the comparison result to an integer mask, and the `popcnt` instruction to count the number of bits set to 1. The timing for this code is about 16 picoseconds per letter compared to 170 picoseconds per letter for my own vectorized implementation of ungapped extension using the BLOSUM62 matrix (created for DIAMOND v2), or 630 picoseconds per letter for scalar ungapped extension[10].

2.1.5 Gapped extension

SIMD vectorisation is essential to achieve high performance when computing gapped extensions. The initial version of DIAMOND v0 used my own implementation of a modified SWIPE (Rognes 2011) approach to compute gapped extensions that are anchored at seed hit locations. The SWIPE approach vectorises the Smith Waterman algorithm by aligning multiple target sequences against the same query. This algorithm in its published form precludes the computation of banded[11] or anchored alignments, since each column of the dynamic programming (DP) matrix has to span over the whole query sequence by design. I have generalised this algorithm to compute alignments of multiple independent query/target pairs by using score profiles that are arrays of substitution scores along the query sequences against a fixed residue and interleaving the scores from these profiles using a vectorised matrix transpose (refer to (Benjamin Buchfink et al. 2023) for a more detailed description). This redesign enables the computation of banded and anchored alignments. The initial version of DIAMOND v0 computed its extensions

[9] DIAMOND has one such constant hard-coded for each of the sensitivity modes that can be chosen on the command line, and optionally allows the user to set it manually.

[10] The timings were measured using microbenchmarks included in DIAMOND on an AMD Ryzen Threadripper 2970WX system, unpublished.

[11] The Smith Waterman algorithm normally computes a DP matrix of size m*n (m = length of the query, n = length of the target). A banded alignment will only evaluate the cells (i, j) of the DP matrix between two diagonals, i.e., for which $d_0 \leq i - j \leq d_1$ holds.

anchored at seed hits with a static band depending on the length of the query, which saves considerable time compared to a full-matrix Smith Waterman that incurs a quadratic cost in the lengths of the sequences.

2.2 DIAMOND version 2

In this subchapter I will describe the algorithms of the DIAMOND v2 software as published in (Benjamin Buchfink, Reuter, and Drost 2021). While DIAMOND v0 had already gained a substantial user community over the years, it still fell short of being suitable as a full replacement for BLAST. DIAMOND v0 operates efficiently not least due to a seeding configuration that still allows reasonable filtration of the search space, thereby compromising with respect to alignment sensitivity.

To make full use of protein sequence comparisons, sensitivity has to be maximised, as related sequences diverge over long evolutionary timescales up to a point of being undetectable on the sequence level, while function and structure may still be conserved (Ruperti et al. 2023). The general pairwise alignment formulation permits the detection of pairwise sequence similarities down to approximately 20% sequence identity. As discussed in the “prior work” subchapter, seeding with respect to this target sensitivity as implemented by BLAST is barely more efficient than brute-force, all-vs-all comparison and would overwhelm any naïve alignment algorithm design. For DIAMOND v2, we set out to solve the technical challenges associated with this problem class and fully replace BLAST for sensitively detecting remote homologs across the tree of life.

I will sketch the most important algorithmic aspects of DIAMOND v2 in this chapter but refer to (Benjamin Buchfink, Reuter, and Drost 2021) for a more elaborate and exhaustive discussion.

2.2.1 Double Indexing and multiple spaced seeds

DIAMOND v2 uses the same basic design as described in the previous chapter of double indexing in combination with multiple spaced seeds. When attempting to increase sensitivity, seeds inevitable must be chosen shorter, and the amount of false positive hits increases exponentially. The synergy of double indexing and multiple spaced seeds is key to coping with this problem: the cache-aware double indexing reduces the cost of seed lookups and allows us to cheaply use a

larger number of multiple spaced seeds, which in turn lets us achieve the same sensitivity with longer and more specific seeds, limiting the amount of false positives.

In addition to the default mode using 2 spaced seeds of weight 10 and the sensitive mode using 16 spaced seeds of weight 8, for DIAMOND v2 we have added the very-sensitive mode using 14 spaced seeds of weight 7 and the ultra-sensitive mode using 64 spaced seeds of weight 7.

2.2.2 Hamming distance filter

Shorter seeds necessarily have more occurrences in the query and target sequences, increasing the amount of data associated with a seed that must be cached and processed together. This data may well exceed the cache capacity available on the CPU, and optimal use of the different levels of the cache hierarchy is desirable. The hamming distance filter (see previous subchapter) functions at the first stage in the pipeline and processes the largest amount of data as it ingests the raw seed hits. We have optimised this stage using a loop-tiling strategy (Wolfe 1989), recursively partitioning the seed hit data into work packages such that cache reuse across the different levels of the cache hierarchy is maximised. To avoid cache pollution by subsequent computations of the pipeline in this stage, seed hits that pass the hamming distance filter are buffered prior to further processing.

2.2.3 Ungapped extension

A minimum ungapped extension score is required for any seed hit that passes through the hamming distance filter stage, which was naively implemented in DIAMOND v0 with scalar instructions. We have vectorised this operation for DIAMOND v2 using AVX2 instructions to align one query sequence against 32 targets simultaneously, based on a vectorized 32x32 matrix transpose operation to interleave target letters for one query position into an AVX2 register. The matrix transposition is implemented as a series of 160 `vpunpckh` and `vpunpckl` instructions, using the `vperm2i128` instruction for 128-bit operations[12]. Substitution scores against target letters interleaved into one register are then loaded using the `vpshufb` instruction.

[12] Matrix transposition using SIMD instructions was described previously, e.g. in the SWIPE (Rognes 2011) source code.

2.2.4 Chaining

As a prerequisite of computing a banded Smith-Waterman alignment as opposed to evaluating the full DP matrix, the seed hits must be used to infer regions of interest and the most likely trajectory of the optimal local alignment. To this end, DIAMOND v2 will chain the diagonal segments obtained from ungapped extensions at the seed hit locations to longer gapped alignments, effectively computing a dynamic program on the level of these diagonal segments instead of amino acid residues. Such a chaining can be seen as a scaffold for the final alignment to be computed and is used to determine the band geometry for a banded Smith Waterman alignment.

2.2.5 Gapped alignment using Smith Waterman

Targets that passed through the filtering pipeline and satisfied internal sensitivity-dependent threshold values in each stage are subjected to banded Smith Waterman alignment to produce the final output. As for most of the performance-critical code in DIAMOND v2, SIMD-vectorisation is essential to achieve good performance in this stage. The fastest vectorisation algorithm that has been described in the literature is SWIPE (Rognes 2011), which in its basic design is only suitable to compute full-matrix local alignments. For DIAMOND v2, I have designed and implemented a modified version of the SWIPE algorithm to allow banded alignments. While the simple SWIPE approach iterates over the complete query sequence for each column of the DP matrix computed, my implementation will overlay the columns of the individual targets based on their query ranges (as implied by the band defined for the target) and for each column only evaluate the minimum query range enveloping these combined query ranges. Geometrically, this cannot be fitted ideally most of the time causing extra cells to be computed as overhead, but we empirically observed an efficiency of 80%-90% when processing larger databases. The main difference of this approach to the approach described in the previous subchapter is that no matrix transposition is needed to load the target scores, so the performance is closer to the original SWIPE implementation.

3. Results and discussion

3.1 DIAMOND version 0

3.1.1 Results

We benchmarked DIAMOND v0.4.7 against BLAST (v2.2.28+) and RAPSearch2 (v2.18) (Zhao, Tang, and Ye 2012) on a total of six short-read and long-read datasets in translated search (BLASTX) mode. We considered RAPSearch2 the state-of-the-art tool at the time and therefore omitted benchmarks against any other tools. The benchmark datasets were five million Illumina reads (100bp) from the Human Microbiome Project ("Human Microbiome Project" 2013), six million Illumina reads (average length 114bp) from permafrost (Mackelprang et al. 2011), long-read metagenomic datasets sequenced using Ion Torrent, 454 Titanium, and Sanger sequencing, as well as 30,000 ORFs predicted from a microbial assembly (Wilson et al. 2014). We searched these datasets using BLAST (v2.2.28+), DIAMOND (v0.4.7) and RAPSearch2 (v2.18) against the NCBI non-redundant database containing 25.9 million protein sequences at the time on a 64-core AMD Opteron server with 512 GB of RAM. DIAMOND and RAPSearch2 were run in two different modes, indicated as *fast* and *sensitive*.

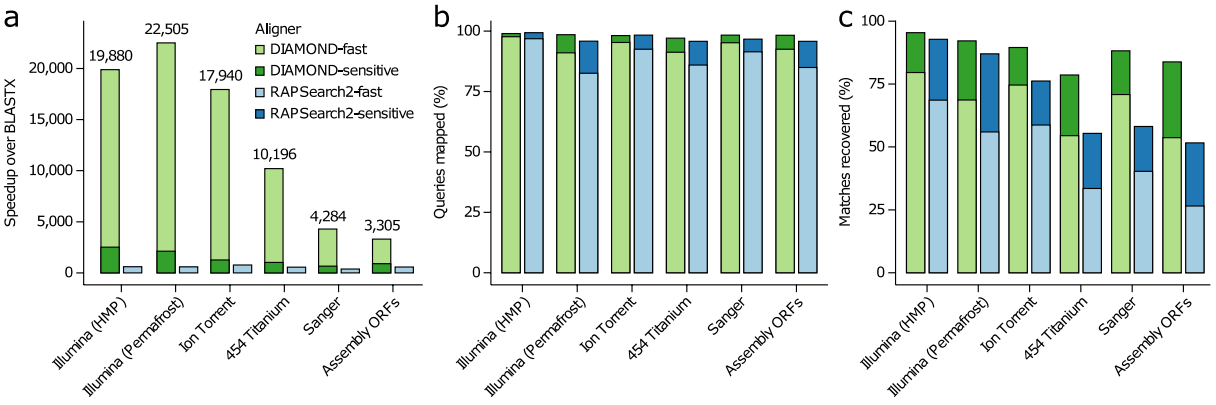


Figure 5. Benchmark of DIAMOND and RAPSearch2 against BLAST on six datasets. (a) the speedup over BLASTX (b) the percentage of query sequences that were aligned relative to BLAST (c) the percentage of alignments that were found relative to BLAST. See appendix (p. 56) for the source data of this figure. From (Benjamin Buchfink, Xie, and Huson 2015) *(Figure created by C. Xie).*

We configured the tools to report the best 250 alignments per query sequence and used the BLAST result as the ground truth for evaluating the other two tools **(Fig. 5)**. DIAMOND-fast was 20,000-fold faster than BLAST on the short-read datasets

and at least 3,000-fold faster on the long-read datasets, and 6 to 38-fold faster than RAPSearch2-fast. DIAMOND-sensitive was 650 to 2,500-fold faster than BLAST and 23 to 58-fold faster than RAPSearch2-sensitive. DIAMOND-fast mapped 91% (Illumina permafrost) to 98% (Illumina human microbiome) of the query sequences relative to BLAST, while DIAMOND-sensitive mapped more than 97% of queries on all datasets. The percentage of the BLAST alignments recovered was in the range of 53% to 79% for DIAMOND-fast and 78% to 95% for DIAMOND-sensitive. In both modes, DIAMOND consistently outperformed RAPSearch2 in terms of sensitivity.

3.1.2 Discussion

Analysing short-read data in the order of terabases produced by metagenomics projects used to be a major computational bottleneck for such studies, which was addressed by DIAMOND v0 aligning short reads 20,000-fold (fast mode) to 2,000-fold (sensitive mode) faster than BLAST at close to its full sensitivity. While not recovering all alignments found by BLAST, for the purpose of functional annotation it is usually sufficient to find some of the best hits for a query, making the fast mode of DIAMOND v0 an attractive option for rapid annotation.

The performance of the tool was generally much better on short reads due to the fact that the protein alignment problem becomes a lot less challenging on shorter sequences (here less than 40 amino acids in length). This happens because alignments on these short sequences must be above roughly 60% sequence identity in order to be statistically significant (as measured by the e-value), which in turn makes it easier to find longer and more specific seed hits. The more challenging task of finding distant similarities down to 20% identity goes away on these short reads, making the problem accessible to efficient solutions.

In practice, this version of DIAMOND has also proven itself useful to annotate reads from long-read technology, assembled contigs, or full-length proteins in BLASTP mode. We have shown that its fast mode generally has good sensitivity for alignments above 60% sequence identity, and its sensitive mode for alignments above 40% identity[13]. While not fully penetrating the twilight zone of protein alignment, these levels of sensitivity have been considered sufficient for many applications. Confidently assigning structure and function to proteins based on low-

[13] Not shown in the first DIAMOND publication, refer to (Benjamin Buchfink, Reuter, and Drost 2021).

identity alignments becomes more difficult (Rost 1999), so the impact of losing this information often seems acceptable[14]. Even the sensitive mode of DIAMOND v0 still ran at a 650 to 2,500-fold speedup relative to BLAST, and made many computations possible that would not have been viable using BLAST, especially for researchers with limited resources. RAPSearch2, which we considered the state-of-the-art BLAST alternative at the time, was also surpassed by DIAMOND v0 in speed and sensitivity by large margins.

I note that the benchmark design of this publication was still a bit rudimentary, and raw alignment output of the three tools may not always be perfectly comparable. I refer to (Benjamin Buchfink, Reuter, and Drost 2021) for a more elaborate benchmark.

3.2 The DTRA challenge

In 2013, I was part of the winning team of the U.S. Defence Threat Reduction Agency's $1 Million Algorithm Challenge[15], together with Daniel Huson and Chao Xie (with each team member sharing 1/3 of the award). Participation in this challenge was a part of precursory work for DIAMOND v0. DTRA is a combat support agency of the U.S. Department of Defence charged with defending against nuclear, chemical, and biological weapons, including weaponized pathogens. It sought a software solution to detect pathogens in clinical samples based on raw DNA sequence. Ten test datasets with reads derived from PacBio, Ion Torrent, Roche-454 and Illumina sequencing technology (containing 100,000 to six million reads) were provided and had to be characterised by the competitors according to their organism composition and gene content. The challenge was twofold: mapping the DNA reads against a big database comprising all genomes of known microorganisms rapidly for real-time diagnosis (max. one hour per sample), and characterising the organisms contained in the sample down to strain level based on these raw DNA alignments.

At the time, nucleotide-level sequence alignment using BLASTN was already too slow for this task. Fast mappers such as Bowtie2 (Langmead and Salzberg 2012) or BWA (H. Li and Durbin 2009; Heng Li and Durbin 2010) were tailored towards read alignment in an intra-species setting with a single reference genome, not

[14] As evidence for this statement, I refer to the studies that have used and cited DIAMOND v0.
[15] Official press release: https://www.prweb.com/releases/dtra/algorithmchallenge/prweb11150852.htm

mapping metagenomic reads to large redundant databases or handling long, error-prone reads in this context. My main contribution to the challenge was the development of the SASS DNA aligner suitable for this application, mapping each test dataset to the microbial part of GenBank (Benson et al. 2011) within minutes. It was based on a hash-table index (Zaharia et al. 2011) instead of the slower, compressed FM-index (Ferragina and Manzini 2000) to accelerate the seed lookup step and efficiently use shorter, more sensitive seeds. For seed extension, it used the fast Myers bit vector algorithm (Myers 1999) as a heuristic filter prior to computing Smith Waterman extensions using SeqAn (Döring et al. 2008).

Alignments produced by the SASS aligner were fed into an analyser component that profiled the organisms and genes contained in the sample. Together, the software package was called MetaScope and described in a preprint (B. Buchfink, Huson, and Xie 2015) after having won the DTRA challenge. Due to limited time and human resources, development of the SASS DNA aligner was not followed up upon, and it has since been surpassed by other aligners such as minimap2 (Heng Li 2018).

3.3 DIAMOND version 2

3.3.1 Results

We benchmarked DIAMOND v2 against BLAST (v2.10.0), DIAMOND v0 (v0.7.2), MMseqs2 (release 11) (Steinegger and Söding 2017) and (at the request of a reviewer) QuickBLAST (v0.0.0). We considered MMseqs2 the state-of-the-art competitor tool at the time, which we still do as of the day of submitting this thesis. The MMseqs2 publication contains a benchmark against LAST (Kiełbasa et al. 2011), RAPSearch2 (Zhao, Tang, and Ye 2012), UBLAST (Edgar 2010), SWORD (Vaser, Pavlović, and Šikić 2016), SWIPE (Rognes 2011), DIAMOND v0 (Benjamin Buchfink, Xie, and Huson 2015) and BLAST.

When benchmarking protein aligners, comparing raw alignment output to a standard tool like BLAST has the inherent problem of disregarding the possibility of false positives, which are known to occur abundantly (Frith 2011). A false positive in this context refers to a pair of sequences with significant similarity that does not stem from actual (biological) homology. While in general we have no way to determine whether an alignment is a false positive or not, we can measure false positive rates

by aligning against random, reversed, shuffled or other sorts of decoy sequences (Frith 2011). In addition, we can be fairly certain about this fact when two sequences are of different origin according to expert curation.

To design a benchmark that is informed by a meaningful biological ground truth, we created query and target datasets annotated with SCOP (Fox, Brenner, and Chandonia 2014) domains. SCOP is an expert curation effort to hierarchically classify protein domain sequences into classes, folds, superfamilies, and families. We used SWIPE[16] (Rognes 2011) at an e-value threshold of 10^{-5} against the SCOPe ASTRAL40 set of domain sequences spanning 4,850 families to annotate the benchmark sequences with one or multiple SCOP domains. For the database, we used 37.5 million protein sequences from UniRef50 (September 2019 release) (Suzek et al. 2015), yielding 7.74 million annotated sequences. For the query set, we annotated the NCBI non-redundant database and selected a maximum of 1,000 sequences for each SCOP superfamily[17] for a combined dataset of 1.71 million sequences. We randomly shuffled all unannotated ranges in the sequences to allow an unequivocal evaluation[18]. We define a positive as an alignment between a query and a target sequence with at least one matching SCOP family annotation, and a negative as an alignment between sequences annotated with disjoint sets of SCOP folds (other cases are ignored).

Given this annotation of query and target datasets, we align the queries against the targets permitting unlimited hits at an e-value threshold of 1,000, such that the result list for each query will almost always contain a false positive. We define the AUC1 sensitivity value for a query as the number of targets matching the query's protein family, considering only targets in the result list until the first false positive, divided by the total number of targets in the database annotated with the query's protein family[19,20]. To obtain a composite AUC1 value for the whole dataset,

---

[16] SWIPE computes exhaustive Smith Waterman alignments against all database sequences and has optimal sensitivity on the pairwise level.

[17] This selection was made by computing a UPGMA clustering for all sequences annotated with the same superfamily, merging clusters until reaching a quantity of 1,000.

[18] Otherwise, alignments between unannotated ranges cannot be clearly identified as positives or negatives. Even when ignoring such cases, alignments that partially span an annotated and an unannotated range would still be ambiguous, while ignoring such cases too would likely lose too much information.

[19] If the query has multiple domain annotations, the AUC1 value is computed for each domain and averaged.

we average the AUC1 value over all queries. In addition to considering this single-number benchmark metric, we also produced ROC curves for our benchmark runs plotting the errors (false positives) per query against the coverage of the query's protein family depending on the e-value threshold (likewise an average over all queries).

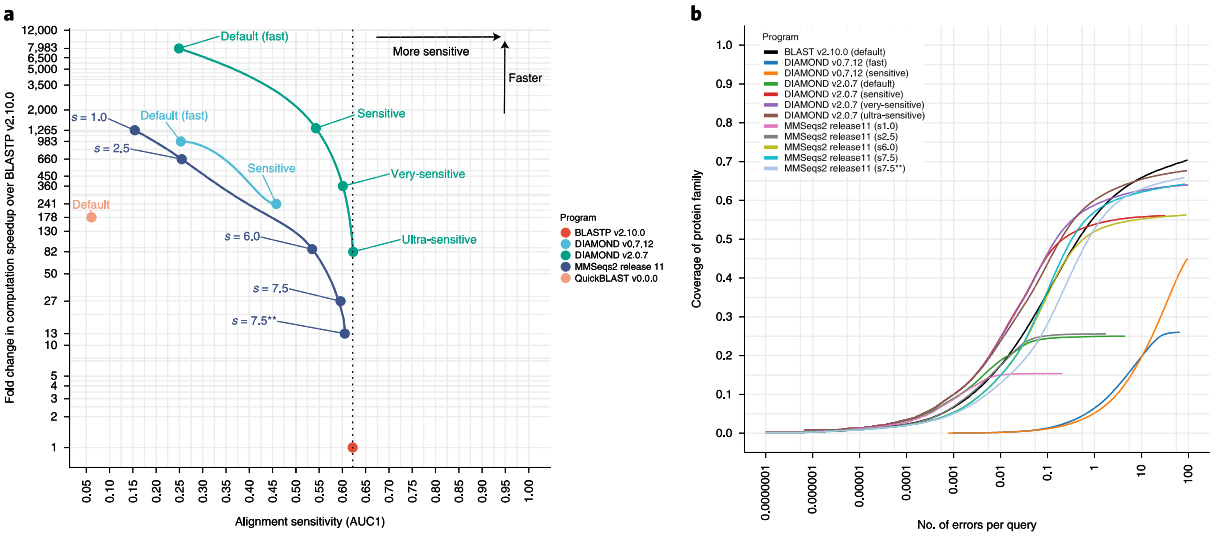


*Figure 6. Benchmark of DIAMOND v2, DIAMOND v0, MMseqs2, BLAST and QuickBLAST in various sensitivity modes as labeled (s=7.5** using the parameter --max-seqs 100000). (a) The recall of the query's protein family until the first false positive (AUC1) against the speedup vs BLAST, (b) ROC curves showing the number of errors per query against the recall of the query's protein family. From* (Benjamin Buchfink, Reuter, and Drost 2021) *(Figure created by H.-G. Drost).*

The benchmark runs were conducted aligning the query set of 1.71 million sequences against the target database of 7.74 million sequences on a 24-core workstation using various sensitivity modes of the tools[21]. Absolute run times ranged from 46 days for BLAST to 8 minutes for DIAMOND v2-fast. **(Fig. 6)** shows the sensitivity as measured by AUC1 against the speedup vs BLAST as well as the ROC curves. The fast and sensitive modes of DIAMOND v2 were 8,000-fold and 1,200-fold faster than BLAST respectively. The newly added very-sensitive mode was 360-fold faster than BLAST while being close to its sensitivity (AUC1 of 0.601 vs 0.622), and the ultra-sensitive mode was still 80-fold faster with equivalent sensitivity.

At comparable sensitivity levels, DIAMOND v2 was 12 to 15-fold faster than the competing MMseqs2 tool. At close to BLAST sensitivity, MMseqs2 was 27-fold

[20] This benchmark design is largely identical to the design used in the MMseqs2 publication (Steinegger and Söding 2017).

[21] The BLAST run was conducted on a compute cluster and its runtime on the benchmark machine estimated based on a sample of 10,000 queries.

faster (s = 7.5) compared to a 360-fold speedup of DIAMOND v2 (very-sensitive). In its most sensitive setting (labelled as s=7.5**), MMseqs2 did not reach the sensitivity of BLAST while running 13-fold faster.

The default and sensitive modes of DIAMOND v2 were 8-fold and 6-fold faster than the corresponding modes in DIAMOND v0. It must be noted here that the BLAST version we used for this benchmark (v2.10.0) is about four times faster than the version used for the benchmark of the DIAMOND v0 publication (v2.2.28), mostly due to better multi-threading efficiency. This means that the speedups we observed here for DIAMOND v0-fast (983-fold) and DIAMOND v0-sensitive (241-fold) are roughly consistent with the speedups reported by the DIAMOND v0 publication for longer sequences.

Based on the ROC curves for the same benchmark **(Fig. 6b)** we observe that both the very-sensitive and the ultra-sensitive mode of DIAMOND v2 perform better than BLAST for searches at error rates below 1, while being slightly surpassed by BLAST at error rates above 1. The better performance at lower error rates is explained by the more sophisticated tantan (Frith 2011) repeat masking employed by DIAMOND v2.

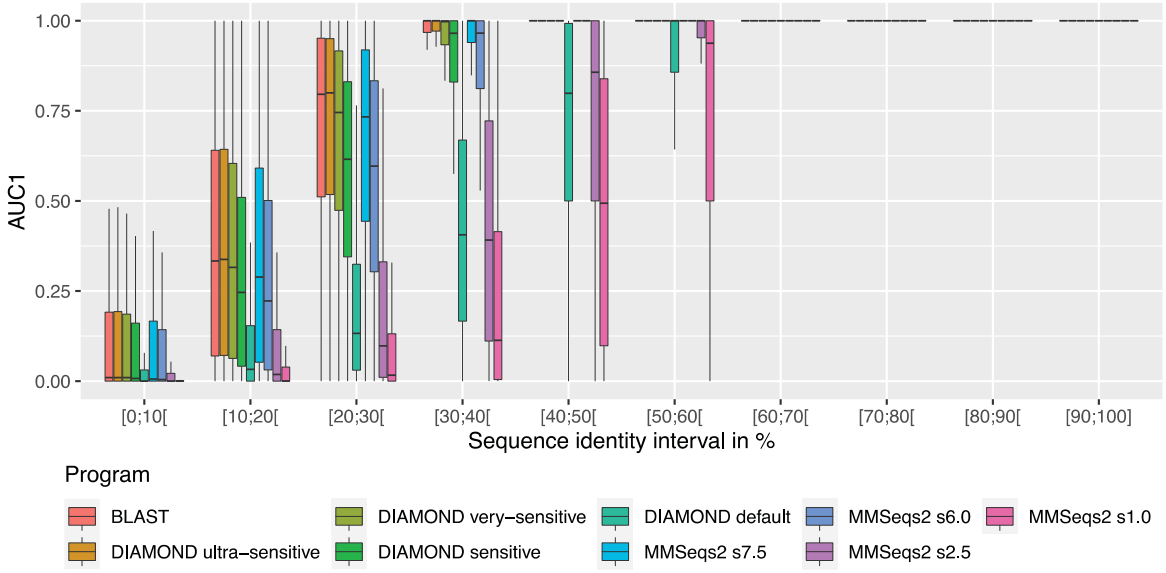


*Figure 7. AUC1 sensitivity (distribution over the queries) as reported for our main benchmark, resolved by sequence identity of the query-target association under our SCOPe annotation (middle = median, hinges=25%/75% quantiles, lower/upper whisker = smallest/largest observation greater/less than or equal to lower/upper hinge -/+ 1.5 * IQR). From* (Benjamin Buchfink, Reuter, and Drost 2021).

It is of further interest to explore the sensitivity depending on the sequence identity of the query-target alignment[22] **(Fig. 7)**. The default mode of DIAMOND v2 exhibits good sensitivity for alignments above 60% sequence identity, and the sensitive mode for alignments above 40% sequence identity. In the range below 40% sequence identity, no tool achieved full sensitivity anymore, and the very-sensitive or ultra-sensitive modes of DIAMOND v2 should be used for sensitivity comparable to BLAST.

3.3.2 Discussion

Building on the same algorithmic core ideas of double indexing and multiple spaced seeds as DIAMOND v0, we have optimised and scaled all stages of the computational pipeline to make it viable for the full spectrum of pairwise protein alignment down to 20% sequence identity. The exponential increase in false positive seed hits is most challenging when attempting to solve sensitive protein alignment, and double indexing combined with our cache-aware hamming distance filter stage proofed crucial to address this problem. The advantage of multiple spaced seeds was also demonstrated by our implementation, as we are still able to use reasonably long seeds of length 7 with up to 64 seed shapes for the ultra-sensitive mode.

Our pipeline consists of a chain of several heuristic filters that gradually reduce the search space and eliminate spurious seed hits. Obviously, none of these heuristics work perfectly and will also cause the loss of some information. While we did not quantify this effect for each computational stage individually, the overall results **(Fig. 6-7)** demonstrate sensitivity competitive to the standard tool BLAST, and likely close to the theoretical maximum that is achievable on the pairwise level, in a biologically informed benchmark.

Our result for the first time showed equivalent sensitivity to BLAST at a speedup of almost two orders of magnitude, making DIAMOND v2 ready to replace BLAST in evolutionary studies and other applications that critically depend on alignment sensitivity. DIAMOND v2 outperformed the state-of-the-art competitor tool MMseqs2 by an order of magnitude, and has to this date of submission of this thesis to the best of my knowledge not been surpassed by any other algorithm in published literature.

---

[22] We defined this as the sequence identity of a Needleman-Wunsch alignment between the pair of annotated ranges in the query and target.

3.4 Sequence clustering

Clustering protein sequences is a natural downstream application of protein sequence alignment. We recently added this feature to DIAMOND (Benjamin Buchfink et al. 2023) and discuss it in this paragraph. Approaches such as Markov clustering (Enright, Van Dongen, and Ouzounis 2002) infer a clustering based on all-vs-all alignment of the input dataset, followed by partitioning of the similarity graph. Since exhaustive all-vs-all alignment has quadratic computational cost in the input size and is often not practical for large datasets, the tools CD-HIT (W. Li and Godzik 2006) and UCLUST (Edgar 2010) with their fast heuristics have established themselves as standard tools for this application. These tools are based on computing a set of cluster representative sequences and requiring that each cluster member sequence satisfies a certain clustering criterion vs their representative sequence, i.e., the pairwise sequence identity being above a certain threshold. CD-HIT and UCLUST still struggle to scale to current dataset sizes with CD-HIT requiring about 24 days on a single server to cluster the NCBI non-redundant database at 90% sequence identity, to becoming unviable for computing deeper clusterings at lower sequence identity thresholds.

MMseqs2/Linclust (Steinegger and Söding 2018) marked an important advance of the clustering performance based on the MMseqs2 alignment algorithm. The tool provided near-linear runtime scaling for clustering for highly similar sequences and is also suitable for computing deep clusterings down to 20% sequence identity. Scaling such a computation to many billions of protein sequences already available today for prokaryotes and arising in the near future for eukaryotes as well as part of large sequencing efforts such as the Earth BioGenome Project (Lewin et al. 2018) still remains challenging. In response, we have implemented a clustering workflow into DIAMOND v2 (Benjamin Buchfink et al. 2023).

Algorithmically, clustering is based on all-vs-all alignment of the protein sequence dataset using cascaded clustering (Hauser, Steinegger, and Söding 2016) as a way to reduce the computational burden by chaining several rounds of comparison with increasing alignment sensitivity. This approach makes use of evolutionarily conserved information to limit the expensive search at full sensitivity to a highly reduced representative set.

We have benchmarked our implementation against MMseqs2 for clustering the NCBI non-redundant database containing 446 million protein sequences. Deep clustering with no limitation of the pairwise sequence identity ran for 7.65 hours on a 64-core server using the very-sensitive mode of DIAMOND, compared to 96.6 hours at corresponding sensitivity (s=7.5) using MMseqs2, resulting in a 13-fold speedup **(Fig. 8)** (*the source data of this figure is included in the appendix at the end of the thesis*).

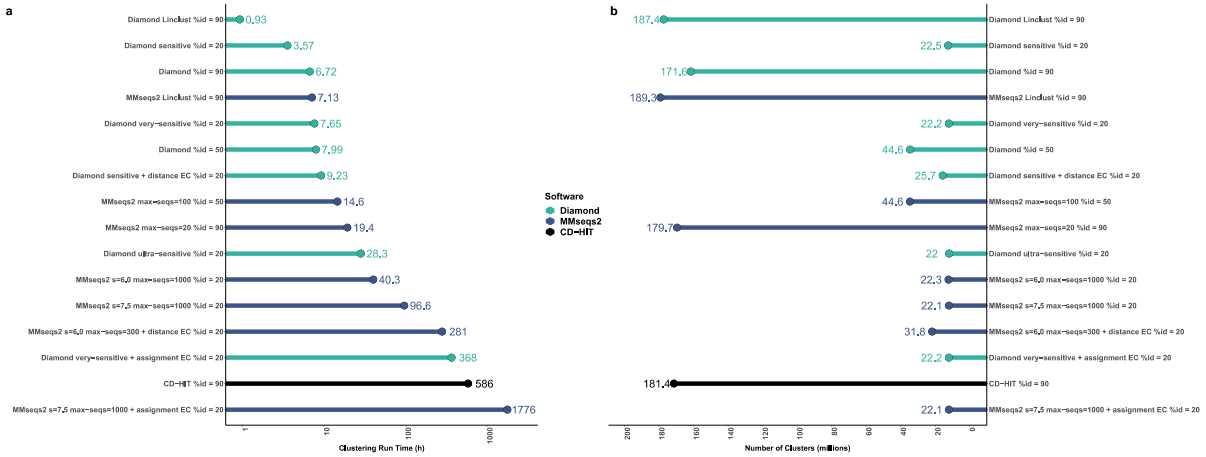


Figure 8. Benchmark for clustering the NCBI NR database (446 million sequences) using DIAMOND, MMseqs2 and CD-HIT at 90%, 50%, and 20% sequence identity thresholds. Distance EC refers to an additional round of error correction to fix cases where sequences do not satisfy the clustering criterion against their assigned representative. Assignment EC refers to error correction that assigns each sequence to its closest representative. (a) The run time for clustering on a 64-core (b) The corresponding number of clusters generated. From (Benjamin Buchfink et al. 2023) *(Figure created by H.-G. Drost) (source data in the appendix at the end of the thesis).*

To demonstrate the ability of our approach to scale to huge datasets and provide a future-proof solution, we collected about 23 billion protein sequences from public databases including JGI IMG (Chen et al. 2023), the Soil Reference Catalogue (Steinegger, Mirdita, and Söding 2019) and various others, corresponding to 19 billion deduplicated sequences. We clustered this dataset with 90% sequence coverage[23] and 30% sequence identity thresholds within 18 days running in parallel on up to 27 compute nodes, consuming 250,000 CPU hours in total. This resulted in 1.7 billion clusters, 544 million of which hold more than one sequence, and 335 million of which have at least three members. This compares to the AlphaFold2 Big Fantastic Database (Jumper et al. 2021) containing 61 million clusters with at least

[23] This means that the local alignment between the cluster representative sequence and the cluster member sequence has to cover at least 90% of the cluster member sequence.

three members, holding the status of the previously largest deeply-clustered protein database. Interpreting the cluster count as a measure of diversity (according to the clustering criterion above that was also used to create the BFD), this result represents an extension of diversity by a factor of 5.5.

The scale of our computation was still modest by High Performance Computing standards, showing that the DIAMOND v2 clustering implementation is ready to handle future problem sizes, including the approximately 27 billion protein sequences that will be generated by the Earth BioGenome Project.

3.5 Related methods

3.5.1 Profile alignment

Pairwise protein alignment does not have perfect sensitivity to detect any sequences of common ancestry, and profile methods have been developed to increase sensitivity. These use profiles in the form of position-specific scoring matrices (PSSMs) (Altschul et al. 1997) or hidden Markov models (Eddy 1998; Remmert et al. 2011) to characterise families of proteins, thereby achieving higher sensitivity than possible using a pairwise comparison.

PSSM-based approached as implemented in PSI-BLAST are closely connected to pairwise alignment, as they involve iterated rounds of searches with an adapted scoring scheme. Progress in pairwise alignment directly translates into improving these methods as well, as was demonstrated by the MMseqs2 tool. A similar feature for supporting PSSM-based profile searches is planned for integration into DIAMOND.

The hidden Markov Model based HMMER suite (Eddy 1998) has various operating modes. JackHMMER searches sequences against a sequence database and builds a profile for the query protein's family at runtime over multiple iterations. It may require several minutes to align a single query sequence against large databases even on a multi-core server (Mirdita et al. 2022), so it cannot easily replace pairwise alignment for performance reasons. The HHblits tool (Remmert et al. 2011) works in a similar way and is reportedly four-fold faster than HMMER.

HMMER's hmmscan searches sequences against a database of profiles such as Pfam (Bateman et al. 2000) and runs faster than JackHMMER, since such profile databases are built on the protein domain level and highly compressed (to

approximately 20,000 families in the case of Pfam). The construction of such profile databases is expensive however and involves pairwise alignment, clustering, and multiple sequence alignment. Pairwise alignment is therefore still an important prerequisite of such methods. Furthermore, the Pfam database does not cover all of the sequence space represented in big databases, for example only covering 53% of UniProtKB on the residue level (Mistry et al. 2021). Therefore, information may also be lost this way despite the higher sensitivity of the search method. Lastly, when annotating protein sequences on a domain level with profiles built from diverse families, the assignment of function for the entire protein may be highly non-trivial. This compares to a pairwise alignment spanning over the entire sequence with sufficiently high sequence identity, more easily informing a functional assignment. Such usability matters are also a reason why pairwise alignment is employed more frequently in practice than HMM profile methods according to citation metrics available from Google Scholar.

In conclusion, pairwise alignment and profile alignment should be regarded as complementary methods and will likely coexist for the foreseeable future.

3.5.2 Machine learning

Machine learning and especially protein language models have been employed to infer homology of proteins with promising results (Bileschi et al. 2022; Kaminski et al. 2022; Schütze et al. 2022), showing higher sensitivity than alignment-based methods. (Schütze et al. 2022) showed strong results for protein domain level benchmarks, but also reduced accuracy on multi-domain proteins, concluding that a hybrid method of alignment and machine learning worked best. The authors also noted that alignment using MMseqs2 ran about 25-fold faster than the protein language model based approach. Based on the numbers of that study, computing PLM embeddings for 446 million sequences of the NR database would run for 1.2 years on a single server equipped with a high-end GPU. In addition, current language models such as ProtT5 (Elnaggar et al. 2021) also impose an upper limit of the sequence length of 1024.

In general, studies in this context so far evaluated their methods based on annotated sequences using curated databases such as ECOD (Cheng et al. 2014), CATH (Sillitoe et al. 2020) or Pfam (Bateman et al. 2000). These sequences are well represented in current databases, but more systematic exploration of how these

machine learning methods generalise to true out-of-sample cases that were never seen during training is still needed. Contrary to classical alignment, a language model has no guarantee to detect a pairwise similarity that is sufficiently high, leaving it open whether current models can correctly capture all of the still unknown sequence space. At this time, more research and development are needed until machine learning methods are ready to be as universally deployed as tools like BLAST or DIAMOND.

3.5.3 Structure prediction

Recent advances in structure prediction (Jumper et al. 2021; Lin et al. 2023) offer an alternative to sequence-based comparisons that can be more sensitive to remotely related sequences. Methods like AlphaFold2 are based on the construction of multiple sequence alignments for the query to find coevolving loci in the protein family. The first step in this computation is the alignment of the query sequence against large clustered sequence databases containing several billion sequences, such as the Big Fantastic Database (Jumper et al. 2021) and MGnify (Mitchell et al. 2019). For better sensitivity, AlphaFold2 uses HMM-based alignment, but it has been shown that similar prediction accuracy is possible using PSSM-based alignment instead (Mirdita et al. 2022). Alignment and clustering also play an important role in creating these databases in the first place.

Computing the prediction itself is expensive (Mirdita et al. 2022) and requires supercomputing resources to be deployed on a large scale (Lin et al. 2023). Also, such predictions cannot be expected to be accurate for just any sequence without sufficient underlying data to support it. For example, for the ESMFold study (Lin et al. 2023) the authors folded about 617 million protein sequences, obtaining 225 million high-confidence predictions. Even in the light of these methods being available, the analysis and alignment of the primary protein structure will remain essential for the foreseeable future.

3.6 Applications

In this subchapter I will discuss some tools, pipelines and biological studies that found my work useful (with no claim of completeness). The citation counts were obtained from Google Scholar in April 2023.

3.6.1 Tools

- **antiSMASH** (Medema et al. 2011; K. Blin, Medema, and Kazempour 2013; Weber et al. 2015; Kai Blin et al. 2017, 2019, n.d.) (~8,800 citations) is a tool to identify biosynthetic gene clusters in bacteria, with a variant for the same functionality in plants called plantiSMASH (Kautsar et al. 2017). Since version 4.0 (Kai Blin et al. 2017), it has replaced BLAST by DIAMOND to search the input data against its database of several hundred thousand known biosynthetic gene clusters.
- **OrthoFinder** (Emms and Kelly 2015, 2019) (~4,600 citations) uses DIAMOND since its 2019 version as its default tool to detect orthologues genes in a set of input genomes, with comparative genomics applications such as inference of phylogenetic gene and species trees and gene annotation.
- **HUMAnN2** (Franzosa et al. 2018) (~1,100 citations) uses DIAMOND to map metagenomic reads to the UniRef (Suzek et al. 2015) and KEGG (Ogata et al. 1999) databases for functional profiling of microbial communities.
- **eggNOG-Mapper** (Huerta-Cepas et al. 2017; Cantalapiedra et al. 2021) (~2,300 citations) uses DIAMOND to map protein sequences to the eggNOG database (Hernández-Plaza et al. 2022) containing 17 million orthologous groups computed at 1,601 taxonomic levels, spanning 10,756 bacterial, 457 archaeal and 1,322 eukaryotic organisms. These orthology assignments are used as a basis for functional annotation that is more accurate than annotation based on homology.
- **panX** (Ding, Baumdicker, and Neher 2018) (216 citations) uses DIAMOND to infer orthogroups for a set of bacterial input genomes of the same species as the basis for building a phylogenetic tree and epidemiological analysis.
- **GenEra** (Barrera-Redondo et al. 2023) (3 citations) uses DIAMOND v2 to infer gene-family founder events by searching an input genome against the NR database in sensitive or ultra-sensitive mode and inspecting unlimited hits to discover the most distantly related sequences. It reportedly achieves comparable accuracy to BLAST-based methods while reducing the runtime from months to days.

3.6.2 Studies

- **“Structure and function of the global topsoil microbiome”** (Bahram et al. 2018) (Nature). The authors obtained soil samples from 1,450 sites worldwide

and sequenced them on the Illumina HiSeq 2500 platform (2×250 bp paired-end). They searched a total of 894 million reads against KEGG (Ogata et al. 1999), CAZy (Lombard et al. 2014) and eggNOG (Hernández-Plaza et al. 2022) using DIAMOND v0 in sensitive mode. They analysed bacterial and fungal genetic diversity and gene composition depending on habitat and environmental factors, and investigated bacterial-fungal interactions.

- **"Marine DNA Viral Macro- and Microdiversity from Pole to Pole"** (Gregory et al. 2019) (Cell). The authors analysed the Global Ocean Viromes 2.0 dataset derived from 3.95 terabases of sequencing data across 145 samples distributed throughout the world's oceans. DIAMOND was used for all-vs-all alignment of predicted protein sequences combined with MCL clustering (Enright, Van Dongen, and Ouzounis 2002).
- **"Compendium of 4,941 rumen metagenome-assembled genomes for rumen microbiome biology and enzyme discovery"** (Stewart et al. 2019) (Nature Biotechnology). The authors analysed 6.5 terabases of short and long-read sequence data from 283 cow rumen samples. 10.7 million protein sequences assembled and predicted from these samples were searched against the KEGG and UniRef databases using DIAMOND, and additionally against CAZy using dbCAN2 (H. Zhang et al. 2018) which also uses DIAMOND. The compendium of rumen-associated bacterial genomes is intended to advance the understanding of rumen microbiota.
- **"Extensive Unexplored Human Microbiome Diversity Revealed by Over 150,000 Genomes from Metagenomes Spanning Age, Geography, and Lifestyle"** (Pasolli et al. 2019) (Cell). The authors reconstructed 150,000 microbial genomes from 9,316 human metagenome samples from multiple populations, body sites, and host ages. All genomes were functionally annotated by aligning against the UniRef90 and UniRef50 databases using DIAMOND in translated search mode. About 3,800 species (or 77% of the total number) were classified as novel (no publicly available genome from isolate sequencing or previous metagenomic assemblies).
- **"A new genomic blueprint of the human gut microbiota"** (Almeida et al. 2019) (Nature). The authors analysed 13,133 sequencing runs of human gut metagenomes spanning 75 studies and reconstructed 92,143 metagenome-

assembled genomes by de-novo assembly. Taxonomic classification was performed by searching predicted genes on these assemblies against the UniProtKB database (UniProt Consortium 2022) using DIAMOND. The authors identified 1,952 uncultured candidate species, substantially increasing the known diversity of the gut microbiome.

- **“A unified catalog of 204,938 reference genomes from the human gut microbiome”** (Almeida et al. 2020) (Nature Biotechnology). The authors created a catalogue of ~200,000 prokaryotic genomes from the human gut. They used eggNOG-Mapper v2 to functionally annotate ~625 million CDS predicted on these assemblies, which internally uses DIAMOND.
- **“Ancient hepatitis B viruses from the Bronze Age to the Medieval period”** (Mühlemann et al. 2018) (Nature). The authors analysed a total of 114 billion Illumina HiSeq 2500 reads from ~300 ancient human samples with a sample range from 7.1 to 0.2 thousand years old. Reads that could not be mapped to the human genome were aligned against the NCBI viral protein reference database containing 274,038 viral proteins using DIAMOND. Reads that matched the Hepatitis B virus could be found in 25 samples spanning a range of approximately 4,000 years. Ancient HBV genomes reconstructed from these sampled were used to study the evolution of this virus through human history.
- **„Genomic features of bacterial adaptation to plants”** (Levy et al. 2017) (Nature Genetics). The authors analysed a set of 3,837 bacterial isolate genomes from plants. Phylogenetic analysis was performed based on coding sequences from these genomes using OrthoFinder and using DIAMOND for alignment. The authors study plant-associated genes to expand the genome-based understanding of plant-microbe interactions.
- **“A reference genome for pea provides insight into legume genome evolution”** (Kreplak et al. 2019) (Nature Genetics). The authors created a pea genome assembly (~ 4 gigabases in size) and investigated legume genome evolution. To determine orthogroups, genes from pea and 21 other eudicot species were processed using OrthoFinder with the DIAMOND alignment option.
- **“Extending and improving metagenomic taxonomic profiling with uncharacterized species using MetaPhlAn 4”** (Blanco-Míguez et al. 2023) (Nature Biotechnology). The authors built a catalogue of 729,000 metagenome-

assembled bacterial and archaeal genomes (spanning both host-associated and environmental samples) and annotated all their coding sequences (a total of 1.5 billion CDS[24]) against the UniRef90 database using DIAMOND. They identify unique marker genes for 27,000 species-level genome bins, 5,000 of which were taxonomically unidentified at the species level.

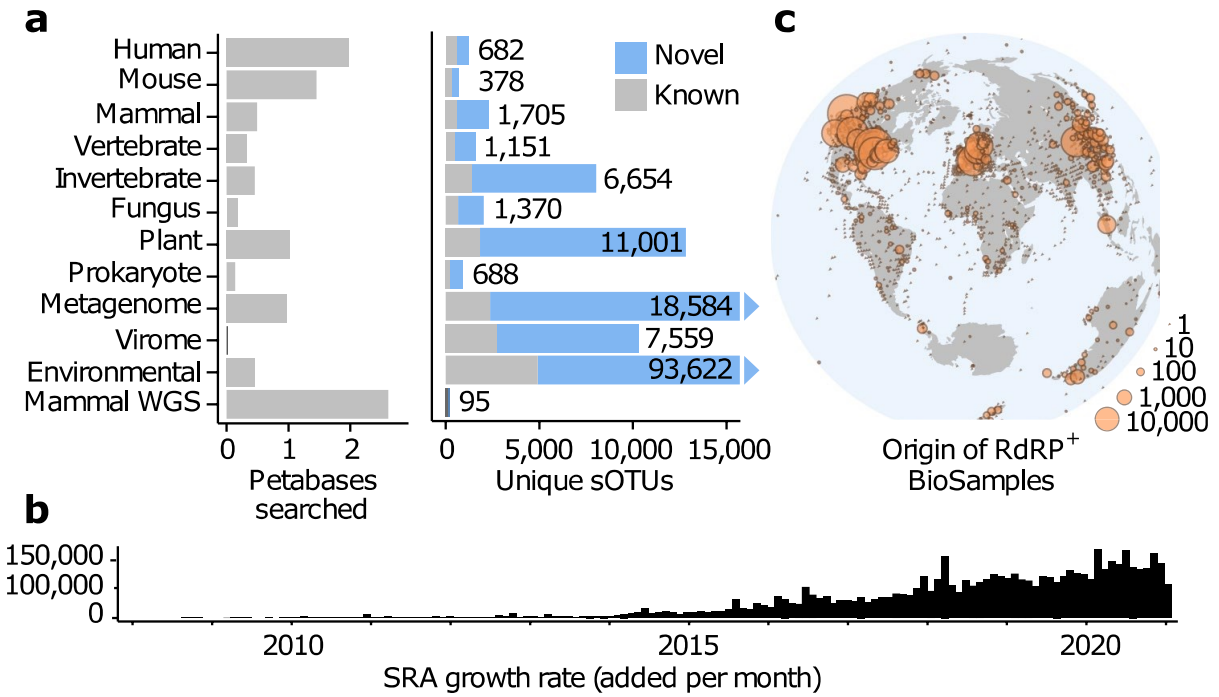


Figure 9. SRA search for viral RdRP. (a) Number of bases searched and number of unique sOTUs found by taxonomic group (b) Release date of the runs included in the search (c) Sample locations for 635,656 RdRP-containing contigs. From (Edgar et al. 2022) *(Figure created by co-authors).*

- **"Petabase-scale sequence alignment catalyses viral discovery"** (Edgar et al. 2022) (Nature). The authors searched 5.7 million samples from the NCBI Sequence Read Archive with a total size of 10.2 petabases for the RNA-dependent RNA polymerase gene. The RdRP is a hallmark gene of RNA viruses commonly used to infer species-level bins. The search was performed on the AWS Elastic Compute Cloud (EC2) using as many as 35,000 cores in parallel over 11 days at a total cost of $24,000 USD. Each SRA library was aligned against a representative database of 14,653 RdRP protein sequences in translated search mode using DIAMOND v2. Reads mapping against the RdRP database were subjected to micro-assembly with subsequent quality control, filtering, classification and clustering to obtain a final set of 132,957 novel sOTUs

[24] According to private correspondence with the first author.

(species-like operational taxonomic units) **(Fig. 9)**. Compared to a set 15,016 known sOTUs, this result represents an increase of known RNA virus species by a factor of 9.8.

My contribution to this project was the development of an optimised DIAMOND v2 feature tailored towards searching very small databases like the RdRP database. To speed up this computation, the database is pre-processed, and its seeds stored in a probabilistic hash set (similar to a Bloom filter). This hash set is used when building the query seed table to filter for seeds not occurring in the database, saving time for construction of the full table. Compared to the unoptimised version of DIAMOND v2, we have measured an overall speedup of 1.47-fold and a reduction of memory use by 64% resulting from this optimisation.

3.7 Conclusion

In this work I have shown a considerable advancement in solving the pairwise alignment problem of protein sequences efficiently. As an inherently hard problem in high dimensional space inaccessible to easy solutions, it can be a limiting factor for many research endeavours within the life sciences. While elementary solutions of this problem have been known for a long time, for practical purposes the computational efficiency can quickly be a determining factor in whether or not the implementation of a research idea is possible. Most scientists around the world are dependent on the ever-limited public funds, constraining their freedom to explore, be creative, and realise their dreams. The work presented here is a modest contribution to lifting these constraints and empower scientists to freely pursue their ideas, whether they possess a supercomputer or merely a personal laptop.

Solving the protein alignment problem is as much a challenge in software engineering as in clever algorithm design. While the concept of multiple spaced seeds has been theoretically described in 2004 (M. Li et al. 2004), it took some nine years for an efficient and practically successful implementation of this concept to materialise. The path from an idea to a software tool that is of practical use to the scientist is long, and careful attention must be paid to each step of the way. A sound understanding of the underlying hardware of modern computer architectures including special instruction sets is an essential part of the equation when developing for high-performance computing applications. This has always been an important

aspect in the development of DIAMOND, which would not have achieved the same performance without architecture-aware algorithm design and careful low-level code optimisation.

Producing software that is useful to the end user has many other aspects, such as compatibility, memory consumption, documentation, support, usability, features, deployment, and testing. These are most easily neglected in software development in an academic context, as academic reward structures are not designed to value such contributions. Consequently, much effort has been spent in the field of bioinformatics on projects with limited practical use. Adhering to these software engineering principles has been a major factor in setting apart DIAMOND from other attempts and producing an outcome that has more than just theoretical and academic value.

Information technology gave rise to an amazing transformation of our world during the last decades, and computers have been getting faster and smarter at a breath-taking pace. In comparison, progress within the field of bioinformatics sometimes appears to happen remarkably slowly, and the discipline tends to be marginalised in academic circles. The conditions under which research software is developed are far away from professional, industry-grade standards. Consequently, scientists are often left to dealing with substandard or archaic software tools. Important breakthroughs such as AlphaFold2 (Jumper et al. 2021) or ESMFold (Lin et al. 2023) were anticipated by industry researches. Only addressing systemic issues in academic research and the full recognition and respect as a science in itself will allow computer science as a part of life sciences research to thrive.

The work presented here has passed the test of time and proven itself useful to the life sciences community. It was successfully used to analyse the largest quantities of sequence data that exist today, from billions of protein coding sequences assembled from metagenomes to petabytes of raw sequencing reads from the SRA. As scientists work to unlock the secrets that these sequences encode, data science will continue to play an indispensable part in this process.

4. References

Almeida, Alexandre, Alex L. Mitchell, Miguel Boland, Samuel C. Forster, Gregory B. Gloor, Aleksandra Tarkowska, Trevor D. Lawley, and Robert D. Finn. 2019. “A New Genomic Blueprint of the Human Gut Microbiota.” *Nature* 568 (7753): 499–504.

Almeida, Alexandre, Stephen Nayfach, Miguel Boland, Francesco Strozzi, Martin Beracochea, Zhou Jason Shi, Katherine S. Pollard, et al. 2020. “A Unified Catalog of 204,938 Reference Genomes from the Human Gut Microbiome.” *Nature Biotechnology*, July. https://doi.org/10.1038/s41587-020-0603-3.

Altschul, S. F., W. Gish, W. Miller, E. W. Myers, and D. J. Lipman. 1990. “Basic Local Alignment Search Tool.” *Journal of Molecular Biology* 215 (3): 403–10.

Altschul, S. F., T. L. Madden, A. A. Schäffer, J. Zhang, Z. Zhang, W. Miller, and D. J. Lipman. 1997. “Gapped BLAST and PSI-BLAST: A New Generation of Protein Database Search Programs.” *Nucleic Acids Research* 25 (17): 3389–3402.

Bahram, Mohammad, Falk Hildebrand, Sofia K. Forslund, Jennifer L. Anderson, Nadejda A. Soudzilovskaia, Peter M. Bodegom, Johan Bengtsson-Palme, et al. 2018. “Structure and Function of the Global Topsoil Microbiome.” *Nature* 560 (7717): 233–37.

Barrera-Redondo, Josué, Jaruwatana Sodai Lotharukpong, Hajk-Georg Drost, and Susana M. Coelho. 2023. “Uncovering Gene-Family Founder Events during Major Evolutionary Transitions in Animals, Plants and Fungi Using GenEra.” *Genome Biology* 24 (1): 54.

Bateman, A., E. Birney, R. Durbin, S. R. Eddy, K. L. Howe, and E. L. Sonnhammer. 2000. “The Pfam Protein Families Database.” *Nucleic Acids Research* 28 (1): 263–66.

Benson, Dennis A., Ilene Karsch-Mizrachi, David J. Lipman, James Ostell, and Eric W. Sayers. 2011. “GenBank.” *Nucleic Acids Research* 39 (Database issue): D32-7.

Bileschi, Maxwell L., David Belanger, Drew H. Bryant, Theo Sanderson, Brandon Carter, D. Sculley, Alex Bateman, Mark A. DePristo, and Lucy J. Colwell. 2022. “Using Deep Learning to Annotate the Protein Universe.” *Nature Biotechnology*, February, 1–6.

Blanco-Míguez, Aitor, Francesco Beghini, Fabio Cumbo, Lauren J. McIver, Kelsey N. Thompson, Moreno Zolfo, Paolo Manghi, et al. 2023. “Extending and Improving Metagenomic Taxonomic Profiling with Uncharacterized Species Using MetaPhlAn 4.” *Nature Biotechnology*, February, 1–12.

Blin, K., M. H. Medema, and D. Kazempour. 2013. “AntiSMASH 2.0—a Versatile Platform for Genome Mining of Secondary Metabolite Producers.” *Nucleic Acids*. https://academic.oup.com/nar/article-abstract/41/W1/W204/1106076.

Blin, Kai, Simon Shaw, Alexander M. Kloosterman, Zach Charlop-Powers, Gilles P. van Wezel, Marnix h. Medema, and Tilmann Weber. n.d. “AntiSMASH 6.0: Improving Cluster Detection and Comparison Capabilities.” *Nucleic Acids Research*. Accessed May 12, 2021. https://doi.org/10.1093/nar/gkab335.

Blin, Kai, Simon Shaw, Katharina Steinke, Rasmus Villebro, Nadine Ziemert, Sang Yup Lee, Marnix H. Medema, and Tilmann Weber. 2019. “AntiSMASH 5.0: Updates to the Secondary Metabolite Genome Mining Pipeline.” *Nucleic Acids Research* 47 (W1): W81–87.

Blin, Kai, Thomas Wolf, Marc G. Chevrette, Xiaowen Lu, Christopher J. Schwalen, Satria A. Kautsar, Hernando G. Suarez Duran, et al. 2017. “AntiSMASH 4.0—Improvements in Chemistry Prediction and Gene Cluster Boundary Identification.” *Nucleic Acids Research* 45 (W1): W36–41.

Buchfink, B., D. H. Huson, and C. Xie. 2015. “MetaScope-Fast and Accurate Identification of Microbes in Metagenomic Sequencing Data.” *ArXiv Preprint ArXiv:1511.08753*. http://arxiv.org/abs/1511.08753.

Buchfink, Benjamin, Haim Ashkenazy, Klaus Reuter, John A. Kennedy, and Hajk-Georg Drost. 2023. “Sensitive Clustering of Protein Sequences at Tree-of-Life Scale Using DIAMOND DeepClust.” *BioRxiv*. https://doi.org/10.1101/2023.01.24.525373.

Buchfink, Benjamin, Klaus Reuter, and Hajk-Georg Drost. 2021. “Sensitive Protein Alignments at Tree-of-Life Scale Using DIAMOND.” *Nature Methods* 18 (4): 366–68.

Buchfink, Benjamin, Chao Xie, and Daniel H. Huson. 2015. “Fast and Sensitive Protein Alignment Using DIAMOND.” *Nature Methods* 12 (1): 59–60.

Cantalapiedra, Carlos P., Ana Hernández-Plaza, Ivica Letunic, Peer Bork, and Jaime Huerta-Cepas. 2021. “EggNOG-Mapper v2: Functional Annotation, Orthology

Assignments, and Domain Prediction at the Metagenomic Scale." *Molecular Biology and Evolution*, October. https://doi.org/10.1093/molbev/msab293.

Chen, I-Min A., Ken Chu, Krishnaveni Palaniappan, Anna Ratner, Jinghua Huang, Marcel Huntemann, Patrick Hajek, et al. 2023. "The IMG/M Data Management and Analysis System v.7: Content Updates and New Features." *Nucleic Acids Research* 51 (D1): D723–32.

Cheng, Hua, R. Dustin Schaeffer, Yuxing Liao, Lisa N. Kinch, Jimin Pei, Shuoyong Shi, Bong-Hyun Kim, and Nick V. Grishin. 2014. "ECOD: An Evolutionary Classification of Protein Domains." *PLoS Computational Biology* 10 (12): e1003926.

Chiu, Charles Y., and Steven A. Miller. 2019. "Clinical Metagenomics." *Nature Reviews. Genetics* 20 (6): 341–55.

Ding, Wei, Franz Baumdicker, and Richard A. Neher. 2018. "PanX: Pan-Genome Analysis and Exploration." *Nucleic Acids Research* 46 (1): e5.

Domazet-Loso, Tomislav, and Diethard Tautz. 2010. "Phylostratigraphic Tracking of Cancer Genes Suggests a Link to the Emergence of Multicellularity in Metazoa." *BMC Biology* 8 (May): 66.

Domazet-Lošo, Tomislav, and Diethard Tautz. 2010. "A Phylogenetically Based Transcriptome Age Index Mirrors Ontogenetic Divergence Patterns." *Nature* 468 (7325): 815–18.

Döring, Andreas, David Weese, Tobias Rausch, and Knut Reinert. 2008. "SeqAn an Efficient, Generic C++ Library for Sequence Analysis." *BMC Bioinformatics* 9 (January): 11.

Eddy, S. R. 1998. "Profile Hidden Markov Models." *Bioinformatics* 14 (9): 755–63.

Edgar, Robert C. 2004. "Local Homology Recognition and Distance Measures in Linear Time Using Compressed Amino Acid Alphabets." *Nucleic Acids Research* 32 (1): 380–85.

———. 2010. "Search and Clustering Orders of Magnitude Faster than BLAST." *Bioinformatics* 26 (19): 2460–61.

Edgar, Robert C., Jeff Taylor, Victor Lin, Tomer Altman, Pierre Barbera, Dmitry Meleshko, Dan Lohr, et al. 2022. "Petabase-Scale Sequence Alignment Catalyses Viral Discovery." *Nature* 602 (7895): 142–47.

Elnaggar, Ahmed, Michael Heinzinger, Christian Dallago, Ghalia Rehawi, Wang Yu, Llion Jones, Tom Gibbs, et al. 2021. “ProtTrans: Towards Cracking the Language of Lifes Code Through Self-Supervised Deep Learning and High Performance Computing.” *IEEE Transactions on Pattern Analysis and Machine Intelligence* PP (July). https://doi.org/10.1109/TPAMI.2021.3095381.

Emms, David M., and Steven Kelly. 2015. “OrthoFinder: Solving Fundamental Biases in Whole Genome Comparisons Dramatically Improves Orthogroup Inference Accuracy.” *Genome Biology* 16 (August): 157.

———. 2019. “OrthoFinder: Phylogenetic Orthology Inference for Comparative Genomics.” *Genome Biology* 20 (1): 238.

Enright, A. J., S. Van Dongen, and C. A. Ouzounis. 2002. “An Efficient Algorithm for Large-Scale Detection of Protein Families.” *Nucleic Acids Research* 30 (7): 1575–84.

Exposito-Alonso, Moises, Hajk-Georg Drost, Hernán A. Burbano, and Detlef Weigel. 2020. “The Earth BioGenome Project: Opportunities and Challenges for Plant Genomics and Conservation.” *The Plant Journal: For Cell and Molecular Biology* 102 (2): 222–29.

Ferragina, Paolo, and Giovanni Manzini. 2000. “Opportunistic Data Structures with Applications.” *Proceedings 41st Annual Symposium on Foundations of Computer Science*. https://ieeexplore.ieee.org/abstract/document/892127/.

Fox, Naomi K., Steven E. Brenner, and John-Marc Chandonia. 2014. “SCOPe: Structural Classification of Proteins--Extended, Integrating SCOP and ASTRAL Data and Classification of New Structures.” *Nucleic Acids Research* 42 (Database issue): D304-9.

Franzosa, Eric A., Lauren J. McIver, Gholamali Rahnavard, Luke R. Thompson, Melanie Schirmer, George Weingart, Karen Schwarzberg Lipson, et al. 2018. “Species-Level Functional Profiling of Metagenomes and Metatranscriptomes.” *Nature Methods* 15 (11): 962–68.

Frith, Martin C. 2011. “A New Repeat-Masking Method Enables Specific Detection of Homologous Sequences.” *Nucleic Acids Research* 39 (4): e23.

Gotoh, O. 1982. “An Improved Algorithm for Matching Biological Sequences.” *Journal of Molecular Biology* 162 (3): 705–8.

Gregory, Ann C., Ahmed A. Zayed, Nádia Conceição-Neto, Ben Temperton, Ben Bolduc, Adriana Alberti, Mathieu Ardyna, et al. 2019. “Marine DNA Viral Macro- and Microdiversity from Pole to Pole.” *Cell* 177 (5): 1109-1123.e14.

Hauser, Maria, Martin Steinegger, and Johannes Söding. 2016. “MMseqs Software Suite for Fast and Deep Clustering and Searching of Large Protein Sequence Sets.” *Bioinformatics* 32 (9): 1323–30.

Hauswedell, Hannes, Jochen Singer, and Knut Reinert. 2014. “Lambda: The Local Aligner for Massive Biological Data.” *Bioinformatics* 30 (17): i349-55.

Henikoff, S., and J. G. Henikoff. 1992. “Amino Acid Substitution Matrices from Protein Blocks.” *Proceedings of the National Academy of Sciences of the United States of America* 89 (22): 10915–19.

Hernández-Plaza, Ana, Damian Szklarczyk, Jorge Botas, Carlos P. Cantalapiedra, Joaquín Giner-Lamia, Daniel R. Mende, Rebecca Kirsch, et al. 2022. “EggNOG 6.0: Enabling Comparative Genomics across 12 535 Organisms.” *Nucleic Acids Research*, November. https://doi.org/10.1093/nar/gkac1022.

Huerta-Cepas, Jaime, Kristoffer Forslund, Luis Pedro Coelho, Damian Szklarczyk, Lars Juhl Jensen, Christian von Mering, and Peer Bork. 2017. “Fast Genome-Wide Functional Annotation through Orthology Assignment by EggNOG-Mapper.” *Molecular Biology and Evolution* 34 (8): 2115–22.

“Human Microbiome Project.” 2013. June 28, 2013. https://commonfund.nih.gov/hmp/.

Huson, Daniel H., and Chao Xie. 2014. “A Poor Man’s BLASTX--High-Throughput Metagenomic Protein Database Search Using PAUDA.” *Bioinformatics* 30 (1): 38–39.

Ilie, Lucian, and Silvana Ilie. 2007. “Multiple Spaced Seeds for Homology Search.” *Bioinformatics* 23 (22): 2969–77.

Ilie, Lucian, Silvana Ilie, Shima Khoshraftar, and Anahita Mansouri Bigvand. 2011. “Seeds for Effective Oligonucleotide Design.” *BMC Genomics* 12 (June): 280.

Jansson, J. 2011. “Towards Tera Terra: Terabase Sequencing of Terrestrial Metagenomics.” LBNL-5191E. Lawrence Berkeley National Laboratory. https://digital.library.unt.edu/ark:/67531/metadc841133/m2/1/high_res_d/1051845.pdf.

Jumper, John, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, et al. 2021. “Highly Accurate Protein Structure Prediction with AlphaFold.” *Nature*, July, 1–11.

Kaminski, Kamil, Jan Ludwiczak, Vikram Alva, and Stanislaw Dunin-Horkawicz. 2022. “PLM-BLAST – Distant Homology Detection Based on Direct Comparison of Sequence Representations from Protein Language Models.” *BioRxiv*. https://doi.org/10.1101/2022.11.24.517862.

Karlin, S., and S. F. Altschul. 1990. “Methods for Assessing the Statistical Significance of Molecular Sequence Features by Using General Scoring Schemes.” *Proceedings of the National Academy of Sciences of the United States of America* 87 (6): 2264–68.

Kautsar, Satria A., Hernando G. Suarez Duran, Kai Blin, Anne Osbourn, and Marnix H. Medema. 2017. “PlantiSMASH: Automated Identification, Annotation and Expression Analysis of Plant Biosynthetic Gene Clusters.” *Nucleic Acids Research* 45 (W1): W55–63.

Kaznadzey, Anna, Natalia Alexandrova, Vladimir Novichkov, and Denis Kaznadzey. 2013. “PSimScan: Algorithm and Utility for Fast Protein Similarity Search.” *PloS One* 8 (3): e58505.

Kennedy, Jonathan, Burkhardt Flemer, Stephen A. Jackson, David P. H. Lejon, John P. Morrissey, Fergal O’Gara, and Alan D. W. Dobson. 2010. “Marine Metagenomics: New Tools for the Study and Exploitation of Marine Microbial Metabolism.” *Marine Drugs* 8 (3): 608–28.

Kent, W. James. 2002. “BLAT--the BLAST-like Alignment Tool.” *Genome Research* 12 (4): 656–64.

Kiełbasa, Szymon M., Raymond Wan, Kengo Sato, Paul Horton, and Martin C. Frith. 2011. “Adaptive Seeds Tame Genomic Sequence Comparison.” *Genome Research* 21 (3): 487–93.

Koskinen, J. Patrik, and Liisa Holm. 2012. “SANS: High-Throughput Retrieval of Protein Sequences Allowing 50% Mismatches.” *Bioinformatics* 28 (18): i438–43.

Kreplak, Jonathan, Mohammed-Amin Madoui, Petr Cápal, Petr Novák, Karine Labadie, Grégoire Aubert, Philipp E. Bayer, et al. 2019. “A Reference Genome

for Pea Provides Insight into Legume Genome Evolution." *Nature Genetics* 51 (9): 1411–22.

Lam, Felix H., Burcu Turanlı-Yıldız, Dany Liu, Michael G. Resch, Gerald R. Fink, and Gregory Stephanopoulos. 2021. "Engineered Yeast Tolerance Enables Efficient Production from Toxified Lignocellulosic Feedstocks." *Science Advances* 7 (26). https://doi.org/10.1126/sciadv.abf7613.

Langmead, Ben, and Steven L. Salzberg. 2012. "Fast Gapped-Read Alignment with Bowtie 2." *Nature Methods* 9 (4): 357–59.

Levy, Asaf, Isai Salas Gonzalez, Maximilian Mittelviefhaus, Scott Clingenpeel, Sur Herrera Paredes, Jiamin Miao, Kunru Wang, et al. 2017. "Genomic Features of Bacterial Adaptation to Plants." *Nature Genetics* 50 (1): 138–50.

Lewin, Harris A., Gene E. Robinson, W. John Kress, William J. Baker, Jonathan Coddington, Keith A. Crandall, Richard Durbin, et al. 2018. "Earth BioGenome Project: Sequencing Life for the Future of Life." *Proceedings of the National Academy of Sciences of the United States of America* 115 (17): 4325–33.

Li, H., and R. Durbin. 2009. "Fast and Accurate Short Read Alignment with Burrows–Wheeler Transform." *Bioinformatics* . https://academic.oup.com/bioinformatics/article-abstract/25/14/1754/225615.

Li, Heng. 2018. "Minimap2: Pairwise Alignment for Nucleotide Sequences." *Bioinformatics* 34 (18): 3094–3100.

Li, Heng, and Richard Durbin. 2010. "Fast and Accurate Long-Read Alignment with Burrows-Wheeler Transform." *Bioinformatics* 26 (5): 589–95.

Li, Ming, Bin Ma, Derek Kisman, and John Tromp. 2004. "Patternhunter II: Highly Sensitive and Fast Homology Search." *Journal of Bioinformatics and Computational Biology* 2 (3): 417–39.

Li, Weizhong, and Adam Godzik. 2006. "Cd-Hit: A Fast Program for Clustering and Comparing Large Sets of Protein or Nucleotide Sequences." *Bioinformatics* 22 (13): 1658–59.

Lin, Zeming, Halil Akin, Roshan Rao, Brian Hie, Zhongkai Zhu, Wenting Lu, Nikita Smetanin, et al. 2023. "Evolutionary-Scale Prediction of Atomic-Level Protein Structure with a Language Model." *Science* 379 (6637): 1123–30.

Locey, Kenneth J., and Jay T. Lennon. 2016. “Scaling Laws Predict Global Microbial Diversity.” *Proceedings of the National Academy of Sciences of the United States of America* 113 (21): 5970–75.

Lombard, Vincent, Hemalatha Golaconda Ramulu, Elodie Drula, Pedro M. Coutinho, and Bernard Henrissat. 2014. “The Carbohydrate-Active Enzymes Database (CAZy) in 2013.” *Nucleic Acids Research* 42 (Database issue): D490-5.

Ma, Bin, John Tromp, and Ming Li. 2002. “PatternHunter: Faster and More Sensitive Homology Search.” *Bioinformatics* 18 (3): 440–45.

Mackelprang, Rachel, Mark P. Waldrop, Kristen M. DeAngelis, Maude M. David, Krystle L. Chavarria, Steven J. Blazewicz, Edward M. Rubin, and Janet K. Jansson. 2011. “Metagenomic Analysis of a Permafrost Microbial Community Reveals a Rapid Response to Thaw.” *Nature* 480 (7377): 368–71.

Mayer, Emeran A., Kirsten Tillisch, and Arpana Gupta. 2015. “Gut/Brain Axis and the Microbiota.” *The Journal of Clinical Investigation* 125 (3): 926–38.

Medema, Marnix H., Kai Blin, Peter Cimermancic, Victor de Jager, Piotr Zakrzewski, Michael A. Fischbach, Tilmann Weber, Eriko Takano, and Rainer Breitling. 2011. “AntiSMASH: Rapid Identification, Annotation and Analysis of Secondary Metabolite Biosynthesis Gene Clusters in Bacterial and Fungal Genome Sequences.” *Nucleic Acids Research* 39 (Web Server issue): W339-46.

Medlar, Alan, and Liisa Holm. 2018. “TOPAZ: Asymmetric Suffix Array Neighbourhood Search for Massive Protein Databases.” *BMC Bioinformatics* 19 (1): 278.

Mirdita, Milot, Konstantin Schütze, Yoshitaka Moriwaki, Lim Heo, Sergey Ovchinnikov, and Martin Steinegger. 2022. “ColabFold: Making Protein Folding Accessible to All.” *Nature Methods*, May, 1–4.

Mistry, Jaina, Sara Chuguransky, Lowri Williams, Matloob Qureshi, Gustavo A. Salazar, Erik L. L. Sonnhammer, Silvio C. E. Tosatto, et al. 2021. “Pfam: The Protein Families Database in 2021.” *Nucleic Acids Research* 49 (D1): D412–19.

Mitchell, Alex L., Alexandre Almeida, Martin Beracochea, Miguel Boland, Josephine Burgin, Guy Cochrane, Michael R. Crusoe, et al. 2019. “MGnify: The

Microbiome Analysis Resource in 2020." *Nucleic Acids Research*, November. https://doi.org/10.1093/nar/gkz1035.

Mühlemann, Barbara, Terry C. Jones, Peter de Barros Damgaard, Morten E. Allentoft, Irina Shevnina, Andrey Logvin, Emma Usmanova, et al. 2018. "Ancient Hepatitis B Viruses from the Bronze Age to the Medieval Period." *Nature* 557 (7705): 418–23.

Murphy, L. R., A. Wallqvist, and R. M. Levy. 2000. "Simplified Amino Acid Alphabets for Protein Fold Recognition and Implications for Folding." *Protein Engineering* 13 (3): 149–52.

Myers, Gene. 1999. "A Fast Bit-Vector Algorithm for Approximate String Matching Based on Dynamic Programming." *Journal of the ACM* 46 (3): 395–415.

Nguyen, Van Hoa, and Dominique Lavenier. 2009. "PLAST: Parallel Local Alignment Search Tool for Database Comparison." *BMC Bioinformatics* 10 (October): 329.

Ogata, H., S. Goto, K. Sato, W. Fujibuchi, H. Bono, and M. Kanehisa. 1999. "KEGG: Kyoto Encyclopedia of Genes and Genomes." *Nucleic Acids Research* 27 (1): 29–34.

Pasolli, Edoardo, Francesco Asnicar, Serena Manara, Moreno Zolfo, Nicolai Karcher, Federica Armanini, Francesco Beghini, et al. 2019. "Extensive Unexplored Human Microbiome Diversity Revealed by Over 150,000 Genomes from Metagenomes Spanning Age, Geography, and Lifestyle." *Cell* 176 (3): 649-662.e20.

Pearson, W. R., and D. J. Lipman. 1988. "Improved Tools for Biological Sequence Comparison." *Proceedings of the National Academy of Sciences of the United States of America* 85 (8): 2444–48.

Quint, Marcel, Hajk-Georg Drost, Alexander Gabel, Kristian Karsten Ullrich, Markus Bönn, and Ivo Grosse. 2012. "A Transcriptomic Hourglass in Plant Embryogenesis." *Nature* 490 (7418): 98–101.

Ramezaniaghdam, Maryam, Nadia D. Nahdi, and Ralf Reski. 2022. "Recombinant Spider Silk: Promises and Bottlenecks." *Frontiers in Bioengineering and Biotechnology* 10 (March): 835637.

Raynaud, Xavier, and Naoise Nunan. 2014. "Spatial Ecology of Bacteria at the Microscale in Soil." *PloS One* 9 (1): e87217.

Remmert, Michael, Andreas Biegert, Andreas Hauser, and Johannes Söding. 2011. “HHblits: Lightning-Fast Iterative Protein Sequence Searching by HMM-HMM Alignment.” *Nature Methods* 9 (2): 173–75.

Rognes, Torbjørn. 2011. “Faster Smith-Waterman Database Searches with Inter-Sequence SIMD Parallelisation.” *BMC Bioinformatics* 12 (June): 221.

Rosenboom, Jan-Georg, Robert Langer, and Giovanni Traverso. 2022. “Bioplastics for a Circular Economy.” *Nature Reviews. Materials* 7 (2): 117–37.

Rost, B. 1999. “Twilight Zone of Protein Sequence Alignments.” *Protein Engineering* 12 (2): 85–94.

Ruperti, Fabian, Nikolaos Papadopoulos, Jacob M. Musser, Milot Mirdita, Martin Steinegger, and Detlev Arendt. 2023. “Cross-Phyla Protein Annotation by Structural Prediction and Alignment.” *Genome Biology* 24 (1): 113.

Sabale, Shrinivas N., Padmaja P. Suryawanshi, and P. U. Krishnaraj. 2020. “Soil Metagenomics: Concepts and Applications.” In *Metagenomics - Basics, Methods and Applications*, edited by Wael N. Hozzein. London: IntechOpen.

Schütze, Konstantin, Michael Heinzinger, Martin Steinegger, and Burkhard Rost. 2022. “Nearest Neighbor Search on Embeddings Rapidly Identifies Distant Protein Relations.” *BioRxiv*. https://doi.org/10.1101/2022.09.04.506527.

Sillitoe, Ian, Nicola Bordin, Natalie Dawson, Vaishali P. Waman, Paul Ashford, Harry M. Scholes, Camilla S. M. Pang, et al. 2020. “CATH: Increased Structural Coverage of Functional Space.” *Nucleic Acids Research*, November. https://doi.org/10.1093/nar/gkaa1079.

Somervuo, Panu, and Liisa Holm. 2015. “SANSparallel: Interactive Homology Search against Uniprot.” *Nucleic Acids Research* 43 (W1): W24-9.

Steinegger, Martin, Milot Mirdita, and Johannes Söding. 2019. “Protein-Level Assembly Increases Protein Sequence Recovery from Metagenomic Samples Manyfold.” *Nature Methods* 16 (7): 603–6.

Steinegger, Martin, and Johannes Söding. 2017. “MMseqs2 Enables Sensitive Protein Sequence Searching for the Analysis of Massive Data Sets.” *Nature Biotechnology* 35 (11): 1026–28.

———. 2018. “Clustering Huge Protein Sequence Sets in Linear Time.” *Nature Communications* 9 (1): 2542.

Stewart, Robert D., Marc D. Auffret, Amanda Warr, Alan W. Walker, Rainer Roehe, and Mick Watson. 2019. “Compendium of 4,941 Rumen Metagenome-Assembled Genomes for Rumen Microbiome Biology and Enzyme Discovery.” *Nature Biotechnology* 37 (8): 953–61.

Sunagawa, Shinichi, Luis Pedro Coelho, Samuel Chaffron, Jens Roat Kultima, Karine Labadie, Guillem Salazar, Bardya Djahanschiri, et al. 2015. “Ocean Plankton. Structure and Function of the Global Ocean Microbiome.” *Science* 348 (6237): 1261359.

Suzek, Baris E., Yuqi Wang, Hongzhan Huang, Peter B. McGarvey, Cathy H. Wu, and UniProt Consortium. 2015. “UniRef Clusters: A Comprehensive and Scalable Alternative for Improving Sequence Similarity Searches.” *Bioinformatics* 31 (6): 926–32.

Suzuki, Shuji, Masanori Kakuta, Takashi Ishida, and Yutaka Akiyama. 2014. “GHOSTX: An Improved Sequence Homology Search Algorithm Using a Query Suffix Array and a Database Suffix Array.” *PloS One* 9 (8): e103833.

Trindade, Marla, Leonardo Joaquim van Zyl, José Navarro-Fernández, and Ahmed Abd Elrazak. 2015. “Targeted Metagenomics as a Tool to Tap into Marine Natural Product Diversity for the Discovery and Production of Drug Candidates.” *Frontiers in Microbiology* 6 (August): 890.

UniProt Consortium. 2022. “UniProt: The Universal Protein Knowledgebase in 2023.” *Nucleic Acids Research*, November. https://doi.org/10.1093/nar/gkac1052.

Vaser, Robert, Dario Pavlović, and Mile Šikić. 2016. “SWORD-a Highly Efficient Protein Database Search.” *Bioinformatics* 32 (17): i680–84.

Vijay, Amrita, and Ana M. Valdes. 2022. “Role of the Gut Microbiome in Chronic Diseases: A Narrative Review.” *European Journal of Clinical Nutrition* 76 (4): 489–501.

Voigt, Christopher A. 2020. “Synthetic Biology 2020-2030: Six Commercially-Available Products That Are Changing Our World.” *Nature Communications* 11 (1): 6379.

Weber, T., K. Blin, S. Duddela, and D. Krug. 2015. “AntiSMASH 3.0—a Comprehensive Resource for the Genome Mining of Biosynthetic Gene Clusters.” *Nucleic Acids*. https://academic.oup.com/nar/article-abstract/43/W1/W237/2467910.

Wilson, Micheal C., Tetsushi Mori, Christian Rückert, Agustinus R. Uria, Maximilian J. Helf, Kentaro Takada, Christine Gernert, et al. 2014. “An Environmental Bacterial Taxon with a Large and Distinct Metabolic Repertoire.” *Nature* 506 (7486): 58–62.

Wolfe, M. 1989. “More Iteration Space Tiling.” In *Proceedings of the 1989 ACM/IEEE Conference on Supercomputing*, 655–64. Supercomputing ’89. New York, NY, USA: Association for Computing Machinery.

Ye, Yuzhen, Jeong-Hyeon Choi, and Haixu Tang. 2011. “RAPSearch: A Fast Protein Similarity Search Tool for Short Reads.” *BMC Bioinformatics* 12 (May): 159.

Zaharia, Matei, William J. Bolosky, Kristal Curtis, Armando Fox, David Patterson, Scott Shenker, Ion Stoica, Richard M. Karp, and Taylor Sittler. 2011. “Faster and More Accurate Sequence Alignment with SNAP.” *ArXiv [Cs.DS]*. arXiv. http://arxiv.org/abs/1111.5572.

Zhang, Han, Tanner Yohe, Le Huang, Sarah Entwistle, Peizhi Wu, Zhenglu Yang, Peter K. Busk, Ying Xu, and Yanbin Yin. 2018. “DbCAN2: A Meta Server for Automated Carbohydrate-Active Enzyme Annotation.” *Nucleic Acids Research* 46 (W1): W95–101.

Zhang, Luoyan, Yi Tan, Shoujin Fan, Xuejie Zhang, and Zhen Zhang. 2019. “Phylostratigraphic Analysis of Gene Co-Expression Network Reveals the Evolution of Functional Modules for Ovarian Cancer.” *Scientific Reports* 9 (1): 2623.

Zhao, Yongan, Haixu Tang, and Yuzhen Ye. 2012. “RAPSearch2: A Fast and Memory-Efficient Protein Similarity Search Tool for next-Generation Sequencing Data.” *Bioinformatics* 28 (1): 125–26.

5. Appendix

Dataset	Reads	Length	Program	Speedup	%Queries (10^{-3})	%Matches (10^{-3})	%Queries (10^{-5})	%Matches (10^{-5})
Illumina (HMP)	5M	101	D-fast	19880	97.69%	79.52%	99.22%	90.25%
			D-sens	2516	99.00%	95.38%	99.60%	97.21%
			R-fast	603	96.86%	68.59%	98.43%	81.35%
			R-sens	43	99.36%	92.74%	99.88%	96.74%
Illumina (Permafrost)	6M	114	D-fast	22505	91.04%	68.65%	94.72%	80.16%
			D-sens	2120	98.54%	92.11%	99.34%	94.16%
			R-fast	582	82.59%	55.96%	87.74%	67.85%
			R-sens	45	95.85%	86.96%	98.71%	92.35%
Ion Torrent	3.3M	211	D-fast	17940	95.31%	74.60%	97.29%	80.62%
			D-sens	1262	98.19%	89.50%	99.21%	92.15%
			R-fast	772	92.50%	58.67%	94.73%	64.17%
			R-sens	45	98.32%	76.16%	99.20%	79.50%
454 Titanium	2.8M	556	D-fast	10196	91.25%	54.48%	92.67%	58.36%
			D-sens	1027	97.09%	78.60%	98.01%	81.45%
			R-fast	557	85.95%	33.48%	87.89%	36.02%
			R-sens	37	95.77%	55.38%	96.87%	57.95%
Sanger	1M	1025	D-fast	4284	95.18%	70.82%	96.13%	72.84%
			D-sens	651	98.33%	88.18%	98.92%	89.71%
			R-fast	375	91.43%	40.29%	92.82%	41.49%
			R-sens	28	96.65%	58.13%	97.59%	59.44%
Assembly ORFs	30K	788	D-fast	3305	92.49%	53.67%	93.17%	56.20%
			D-sens	894	98.29%	83.80%	98.70%	86.12%
			R-fast	575	84.95%	26.55%	86.25%	27.87%
			R-sens	34	95.78%	51.61%	96.48%	53.38%

Source data for figure 5 of the thesis/figure 1 from "Fast and sensitive protein alignment using DIAMOND" (2015). Comparison of four different programs and settings, D-fast (DIAMOND, fast setting), D-sens (DIAMOND, sensitive setting), R-fast (RAPSearch2, fast setting) and R-sens (RAPSearch2, sensitive setting), against BLASTX. For six different datasets, each with the specified number of reads ("# Reads") and average read length ("Length"), we report the speedup of each program over BLASTX ("Speedup"), the percentage of queries for which the program reports at least one alignment ("% Queries") and the percentage of matches recovered by the program ("% Matches"). The two latter quantities are reported both for an e-value threshold of 10^{-3} and 10^{-5}. Programs were set to report alignments for up to 250 target sequences per read. Times are wall-clock times on a server using 48 cores and exclude one-time program startup overhead, which was less than a minute for BLASTX and five minutes for DIAMOND-fast. Supplementary Table 1 from (Benjamin Buchfink, Xie, and Huson 2015).

The source data of the figures for the DIAMOND DeepClust paper are at the end of the thesis after the manuscript of that paper.

Material from: 'Buchfink, B., Xie, C. & Huson, D. Fast and sensitive protein alignment using DIAMOND. Nat Methods 12, 59–60 (2015), Springer Nature Limited'

Fast and sensitive protein alignment using DIAMOND

Benjamin Buchfink[1], Chao Xie[2,3] & Daniel H Huson[1,2]

The alignment of sequencing reads against a protein reference database is a major computational bottleneck in metagenomics and data-intensive evolutionary projects. Although recent tools offer improved performance over the gold standard BLASTX, they exhibit only a modest speedup or low sensitivity. We introduce DIAMOND, an open-source algorithm based on double indexing that is 20,000 times faster than BLASTX on short reads and has a similar degree of sensitivity.

© 2015 Nature America, Inc. All rights reserved.

In metagenomics studies, millions of sequence reads are analyzed to determine the functional or taxonomic content of microbial samples from the environment[1]. An important computational step is to determine which genes are present, usually by aligning translated DNA sequences against a reference database of protein sequences such as the NCBI nonredundant (NCBI-nr) database[2] or KEGG[3]. BLASTX[4] has long been considered the gold standard tool for this owing to its high sensitivity. However, BLASTX is much too slow for routine application in a high-throughput context.

A number of faster approaches have been proposed, such as BLAT[5] and USEARCH[6]. Notably, RAPSearch2 (ref. 7) improves speed by a factor of up to 100 over BLASTX while maintaining a similar level of sensitivity. However, as the size and number of samples continue to grow, even faster methods are required. PAUDA[8] provides a 10,000-fold increase in speed over BLASTX but has very low sensitivity on the level of individual alignments, reporting only 2–3% of all BLASTX alignments.

Here we present DIAMOND (double index alignment of next-generation sequencing data), an open-source program that is ideally suited for replacing BLASTX in a high-throughput setting (http://ab.inf.uni-tuebingen.de/software/diamond and **Supplementary Software**). When targeting significant alignments against the NCBI-nr database with an expected value below 10^{-3}, DIAMOND aligns short sequence reads at approximately 20,000 times the speed of BLASTX and has a similar level of sensitivity. Like BLASTX, DIAMOND is an 'all mapper' that attempts to determine exhaustively all significant alignments for a given query.

Most sequence comparison programs, including BLASTX, follow the seed-and-extend paradigm. In this two-phase approach, users search first for matches of seeds (short stretches of the query sequence) in the reference database, and this is followed by an 'extend' phase that aims to compute a full alignment.

Sequence comparison programs typically precompute an index that holds all seed locations in the reference sequences. A file of queries is then linearly scanned, and the seeds of a given query are matched to seeds in the reference sequences by random-access lookups in the index. In contrast, DIAMOND uses double indexing, an approach that determines the list of all seeds and their locations in both the query and reference sequences (Online Methods and **Supplementary Software**). The two lists are sorted lexicographically and traversed together in a linear manner to determine all matching seeds and their corresponding locations. Double indexing takes advantage of the cache hierarchy by increasing data locality, thus reducing the demands on main memory bandwidth.

Most sequence comparison programs, including BLASTX and RAPSearch2, use single consecutive seeds, which need to be short (length 3–6 amino acids) to ensure sensitivity. To increase speed without losing sensitivity, DIAMOND uses spaced seeds—that is, longer seeds in which only a subset of positions are used[9,10]. The number and exact layout of those positions are called the weight and shape of the spaced seed, respectively. To achieve high sensitivity, DIAMOND uses a set of four carefully chosen shapes[11] of length 15–24 and weight 12 by default. The most sensitive version of DIAMOND uses 16 shapes of weight 9. In addition, DIAMOND uses a reduced amino acid alphabet of size 11 to enhance sensitivity[12]. A simple exact match criterion determines which seeds are passed on to the extension phase, in which a Smith-Waterman alignment[13] is computed.

In a recent metagenomic study of 12 permafrost samples[14], a BLASTX comparison of 176 million high-quality DNA reads against the KEGG reference database[3] was reported to require 800,000 CPU hours at a supercomputing center[15]. When we used DIAMOND with its default settings, the analysis of all 246 million reads took 2.3 h on a single workstation, producing a total of 568.9 million alignments on 43 million reads.

To systematically compare the performance of DIAMOND (version 0.4.7) with BLASTX (version 2.2.28+) and RAPSearch2 (version 2.18), we downloaded publicly available metagenome data sets produced with Illumina (permafrost[14] and Human Microbiome Project[16] data), Ion Torrent (ERP004234), 454 Titanium (SRR1298978 and SRR1298979) and Sanger[17] sequencing technologies as well as open reading frames (ORFs) predicted from a microbial assembly[18]. We used all three programs to align all data sets against NCBI-nr (version May 2013) on a single workstation using 48 cores. We ran both DIAMOND and RAPSearch2

[1]Department of Computer Science and Center for Bioinformatics, University of Tübingen, Tübingen, Germany. [2]Singapore Centre on Environmental Life Sciences Engineering, School of Biological Sciences, Nanyang Technological University, Singapore. [3]Life Sciences Institute, National University of Singapore, Singapore. Correspondence should be addressed to B.B. or D.H.H..

RECEIVED 29 APRIL; ACCEPTED 20 OCTOBER; PUBLISHED ONLINE 17 NOVEMBER 2014; DOI:10.1038/NMETH.3176

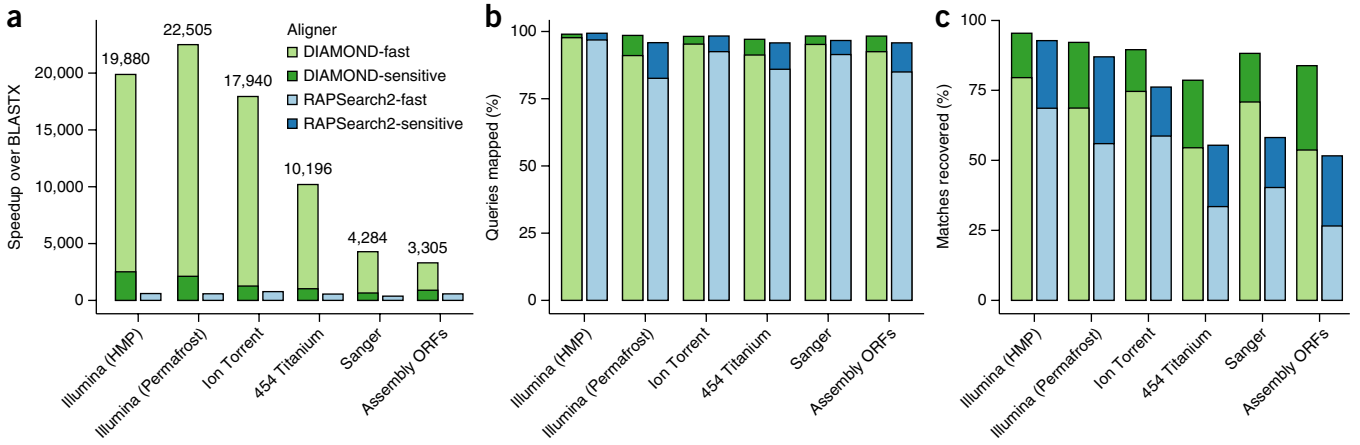


Figure 1 | Comparison of DIAMOND and RAPSearch2 against BLASTX for four sequencing technologies and for ORFs predicted from a bacterial assembly. (**a**) Fold speedup of each program over BLASTX. (**b**) Percentage (relative to BLASTX) of queries for which each program reports at least one alignment. (**c**) Percentage (relative to BLASTX) of matches recovered by each program. Only alignments with an expected value of ≤0.001 are considered. Programs were set to report alignments for up to 250 target sequences per read. Times are wall-clock times on a server using 48 cores and exclude one-time program startup overhead, which was <1 min for BLASTX and 5 min for DIAMOND-fast. HMP, Human Microbiome Project.

using a fast setting (–fast) and a sensitive setting (–sensitive) (**Fig. 1** and **Supplementary Table 1**).

DIAMOND-fast uses four seed shapes and runs around 20,000 times faster than BLASTX. DIAMOND-sensitive uses 16 seed shapes and runs about 2,000 times faster than BLASTX, aligning 99% of reads that have a BLASTX alignment and obtaining over 92% of all BLASTX alignments, on Illumina reads. DIAMOND-fast was 40 times faster than RAPSearch2-fast, with greater sensitivity, and was up to 500 times faster than RAPSearch2-sensitive, with similar sensitivity. DIAMOND-fast and DIAMOND-sensitive consistently outperformed RAPSearch2-fast and RAPSearch2-sensitive, respectively, on all data sets. The peak memory usage of DIAMOND-default was 100 GB, but the program can be configured to require less than 32 GB of memory, at the expense of a 30% reduction in speed.

Whereas the functional analysis of metagenome data sets has in the past been restricted by the requirement for supercomputing services, researchers can now use DIAMOND to perform functional analysis routinely on all their data sets. An analysis that takes 1 month with BLASTX takes only a few minutes with DIAMOND. Functional analysis of all 35 billion Illumina reads generated by the Human Microbiome Project, the largest published metagenome data set to date, would take about 2 weeks on a single server with DIAMOND.

METHODS

Methods and any associated references are available in the online version of the paper.

Note: Any Supplementary Information and Source Data files are available in the online version of the paper.

ACKNOWLEDGMENTS

This research was partially supported by the National Research Foundation and Ministry of Education Singapore under its Research Centre of Excellence Programme, and by the A*STAR Computational Resource Centre through the use of its high-performance computing facilities.

AUTHOR CONTRIBUTIONS

B.B. designed and implemented the algorithm. C.X. performed the experimental study. C.X. and D.H.H. initiated and guided the project. D.H.H. and B.B. wrote the manuscript.

COMPETING FINANCIAL INTERESTS

The authors declare no competing financial interests.

Reprints and permissions information is available online at http://www.nature.com/reprints/index.html.

1. Handelsman, J., Rondon, M., Brady, S., Clardy, J. & Goodman, R. *Chem. Biol.* **5**, R245–R249 (1998).
2. Benson, D.A., Karsch-Mizrachi, I., Lipman, D., Ostell, J. & Wheeler, D. *Nucleic Acids Res.* **33**, D34–D38 (2005).
3. Kanehisa, M. & Goto, S. *Nucleic Acids Res.* **28**, 27–30 (2000).
4. Altschul, S.F., Gish, W., Miller, W., Myers, E.W. & Lipman, D.J. *J. Mol. Biol.* **215**, 403–410 (1990).
5. Kent, W.J. *Genome Res.* **12**, 656–664 (2002).
6. Edgar, R.C. *Bioinformatics* **26**, 2460–2461 (2010).
7. Zhao, Y., Tang, H. & Ye, Y. *Bioinformatics* **28**, 125–126 (2012).
8. Huson, D.H. & Xie, C. *Bioinformatics* **30**, 38–39 (2014).
9. Burkhardt, S. & Kärkkäinen, J. *Fundamenta Informaticae* **23**, 1001–1018 (2003).
10. Ma, B., Tromp, J. & Li, M. *Bioinformatics* **18**, 440–445 (2002).
11. Ilie, L., Ilie, S., Khoshraftar, S. & Bigvand, A.M. *BMC Genomics* **12**, 280 (2011).
12. Murphy, L.R., Wallqvist, A. & Levy, R.M. *Protein Eng.* **13**, 149–152 (2000).
13. Smith, T.F. & Waterman, M.S. *J. Mol. Biol.* **147**, 195–197 (1981).
14. Mackelprang, R. *et al. Nature* **480**, 368–371 (2011).
15. Jansson, J. *Microbe* **6**, 309–315 (2011).
16. Turnbaugh, P.J. *et al. Nature* **449**, 804–810 (2007).
17. Venter, J.C. *et al. Science* **304**, 66–74 (2004).
18. Wilson, M.C. *et al. Nature* **506**, 58–62 (2014).

© 2015 Nature America, Inc. All rights reserved.

ONLINE METHODS

Overview of DIAMOND. DIAMOND is a high-throughput alignment program that compares a file of DNA sequencing reads against a file of protein reference sequences, such as NCBI-nr[19] or KEGG[3]. It is implemented in C++ and is designed to run on multicore servers. The software can be obtained at http://ab.inf.uni-tuebingen.de/software/diamond.

DIAMOND is 4 orders of magnitude faster than BLASTX[4] at comparing short DNA reads against the NCBI-nr database and maintains a comparable level of sensitivity on alignments with an e-value $<10^{-3}$. The program is explicitly designed to make use of modern computer architectures that have large memory capacity and many cores. It follows the seed-and-extend approach. Additional algorithmic ingredients are the use of a reduced alphabet, spaced seeds and double indexing.

Seed and extend. The program is based on the traditional seed-and-extend paradigm for sequence comparison, in which exact occurrences of seeds (that is, short words of a given fixed length) contained in query sequences are located in the reference sequences, and these seed matches are then extended, if possible, to full alignments between the queries and references. The seed length used by an alignment program has a substantial impact on performance; shorter seeds increase sensitivity, whereas longer seeds increase speed.

Reduced alphabet. To increase speed without losing sensitivity, one approach is to use a reduced alphabet when comparing seeds. Using this, RAPSearch2 (ref. 7) is 40–100 times faster than BLASTX with minimal loss of sensitivity. For DIAMOND, we investigated the use of published reductions to four, eight and ten letters[12]. By analyzing a large number of BLASTX alignments, we developed a new reduction to an alphabet of size 11 that achieves slightly better sensitivity (brackets indicate one letter): [KREDQN] [C] [G] [H] [ILV] [M] [F] [Y] [W] [P] [STA].

Spaced seeds. A second improvement of the seed step is to use spaced seeds—that is, longer seeds in which only a subset of positions are used. The number and exact layout of those positions are called the weight and shape of the spaced seed, respectively. Theoretical analysis shows that a single spaced seed can perform better than a contiguous seed of the same weight, if its shape is suitably chosen[10]. Moreover, sensitivity can be increased further by using additional seed shapes, each resulting only in a sublinear increase in running time[10]. By default, DIAMOND uses a set of four shapes[11] of length 15–24 and weight 12 (**Supplementary Fig. 1**).

Seed index. The main bottleneck in aligning a large number of reads against a large reference database is not CPU performance but rather memory latency and bandwidth. The limiting factor is the amount of time required to load seed locations from main memory for comparison. Moving data from main memory into the cache takes hundreds of CPU clock cycles. Thus, a fast algorithm should take the cache hierarchy of computers into account so as to maximize data locality and minimize the number of main memory accesses. This can be done by decomposing the problem into small subproblems that fit into the cache.

In most seed-and-extend programs, an index structure (such as hash table or FM index) is built on the reference sequences to facilitate the location of seeds in the references. Queries are processed in the order that they occur in the input file. For a given query, all seeds are determined and the index is then used to look up all matching locations in the reference sequences. The reference locations are loaded from main memory into the cache. When another read is later encountered that contains some of the same seeds, the corresponding index and sequence data will usually have been evicted from the cache, so this data will have to be loaded from main memory again. Hence, using a single index in this way does not make good use of the cache. The problem is compounded when using a full-text index, such as a suffix array or a compressed FM index, as these require multiple individual memory accesses per index lookup.

Double indexing. DIAMOND uses a double-indexing approach in which both the queries and the references are indexed. A DIAMOND index is a list of seed-location pairs that is lexicographically ordered on a compressed representation of the seed. By traversing the two index lists linearly in parallel, all matching seeds between the queries and the references are identified, allowing the local alignment computation at the corresponding seed locations. The index memory access pattern of this approach is linear and can be efficiently handled by the hardware prefetcher, which will fill the cache with the indexing information before it is needed.

The double-indexed approach also improves data locality with respect to accessing the sequences. To see this, let S_q and S_r denote the set of seeds contained in the set of queries and references, respectively. For a given seed s, let m_s and n_s be the number of occurrences of the seed in the queries and the references. Using the standard index approach, the number of memory access operations is approximately $K = \Sigma_{s \in Sq \cap Sr} m_s n_s$, as for each occurrence of a reference seed in the queries, all corresponding reference locations will be loaded into memory. Using the double-indexing approach, the number of memory access operations is approximately $K' = \Sigma_{s \in Sq \cap Sr}(m_s + n_s)$, assuming that the combined size of query and reference locations for one seed is small enough to fit into the cache. This number is much smaller than K unless the sum is dominated by singleton seeds. To demonstrate this effect on real data, we plot the ratio of memory accesses for the two approaches depending on the length of the query sequence in letters as observed on our benchmark data (**Supplementary Fig. 2**).

The double-indexing algorithm used by DIAMOND is based on the well-known database sort-merge join algorithm, applied to the two seed sets of the queries and references. The main computation, the compilation and sorting of the two lists of seeds, can be efficiently addressed in parallel using a radix clustering step[20] in combination with a fast sorting algorithm. The total amount of time required to sort all the seeds in the given set of queries is smaller than what is required to access all seeds in a hash-table approach. Further, sorting all seeds in the reference sequences 'on the fly' takes much less time than loading a precomputed index. For example, the complete seed index for the current version of the NCBI-nr database is about 100 GB in size. This takes 100 s to generate in memory and about six times as long to read from disk at a typical read rate of 150 MB per second.

The alignment program mrsFast uses sorted lists as an index structure[21]. The authors of that tool spent a lot of effort on making their algorithm cache oblivious. Much of the challenge discussed there stems from the fact that short nonoverlapping seeds were used, which causes the set of all occurrences of a seed to exceed the cache capacity. Owing to the more elaborate seed strategy used by DIAMOND, in our program the amount

© 2015 Nature America, Inc. All rights reserved.

of data associated with a given seed will always be small enough to fit into the cache.

Memory efficiency. A drawback of using multiple spaced seeds is that this uses a lot of memory, which is a main reason why this approach has not been widely used, despite its proven advantages. The naïve implementation of a multiple spaced seed index builds a hash-table index for each of the seed shapes. With one hash table index for the NCBI-nr database being about 100 GB in size, four seed shapes would consume 400 GB, and 16 shapes would consume 1.6 TB of memory.

DIAMOND constructs and processes its indexes for one shape at a time, freeing up the memory used by one shape before moving on to the next. Thus, DIAMOND can perform alignment tasks with its sensitive 16-shape configuration using only as much memory as one shape index requires, which is an additional advantage of our approach. Moreover, using the radix cluster technique, the seed space is decomposed into 1,024 disjoint partitions. By building and processing indexes for only a subset of these partitions at the same time, the memory usage will be limited to the size of the subset index.

Seed extension. For each seed match found, DIAMOND determines whether it can be extended to an ungapped alignment of ten or more amino acids. If this is the case, then the seed match triggers the extend phase of the algorithm, which involves computing a Smith-Waterman alignment[13]. DIAMOND uses its own streaming SIMD extension (SSE)-accelerated Smith-Waterman implementation that extends previous algorithms[22] to allow the computation of banded and anchored alignments. By default, the program uses the BLOSUM62 matrix[23], a gap score of 11 and an extension score of 1, however, other BLOSUM matrices and scoring parameters can be used. The program determines the bit score and expected value of the computed alignment as in BLASTX. By default, alignments with a bit score <50 are not reported.

Because DIAMOND proceeds seed by seed rather than read by read, a key issue is how to avoid computing the same local alignment between a read and a reference more than once at different times during the search phase. To address this, DIAMOND allows a seed match to trigger an extension only if it is the left-most seed match in the corresponding ungapped alignment.

Experimental study. We downloaded the NCBI-nr database, which consists of 25.9 million sequences and 8.9 billion letters, in May 2013 to use as reference database.

We downloaded ten files of Illumina reads from the Human Microbiome Project website (http://www.hmpdacc.org/) covering samples from a range of different human-associated microbiomes (SRA SRS011134, SRS013687, SRS013951, SRS015578, SRS042628, SRS011239, SRS013800, SRS015369, SRS016753 and SRS053335). We extracted 500,000 random reads from each file so as to obtain a total set of 5 million reads, of average length 101. We refer to this as the Illumina (HMP) data set. We downloaded 12 Illumina data sets associated with permafrost cores[14] from the US Department of Energy Joint Genome Institute website. We extracted 500,000 random reads from each file so as to obtain a total set of 6 million reads, of average length 114, to use as our benchmark set of read sequences. We refer to this as our Illumina (permafrost) data set.

We downloaded a single data set of Ion Torrent reads from a study entitled 'Metagenome from artisanal cheeses from Tucuman' from NCBI (ERP004234). We downloaded two data sets of 454 Titanium reads from a study entitled 'Microbial community gene content and expression in the Central North Pacific Gyre, Station ALOHA, HOT186' from NCBI, with accession numbers SRR1298978 and SRR1298979. We downloaded a single data set of Sanger reads from the Sargasso Sea project[17]. We download a set of contigs from a microbial assembly[18] and used MetaGeneMark[24] to predict a total of 30,000 open reading frames (ORFs).

These data sets were used to compare the performance of DIAMOND (version 0.4.7), RAPSearch2 (version 2.18) and BLASTX (version 2.2.28+). All three programs were run on the same computer using 48 cores of a 64 core AMD Opteron server with 512 GB of main memory, running Ubuntu 12.04. The following parameter settings were used for the programs. BLASTX: blastx _num_threads 48 -evalue 0.1 -max_target_seqs 250 -comp_based_stats 0; DIAMOND-fast: diamond blastx -t 48 -k 250 -min-score 40; DIAMOND-sensitive: diamond blastx -t 48 -k 250 -sensitive -min-score 40; RAPSearch2-fast: rapsearch -a T -z 48 -v 250 -e -1 -b 1; RAPSearch2-default: rapsearch -z 48 -v 250 -e -1 -b 1.

To measure run times, BLASTX and RAPSearch2 were each run on three random subsets of each data set, and the run times were then extrapolated to the full data sets and averaged. DIAMOND was run on the full data sets (**Fig. 1** and **Supplementary Table 1**). For **Figure 1**, we considered only alignments with a BLASTX e-value below 10^{-3}. For **Supplementary Table 1** we also considered only the subset of all alignments with a BLASTX e-value below 10^{-5}. Alignments of poorer quality are usually not considered in metagenome analysis.

The reported run times are the total wall-clock time required on 48 cores, minus the program's overhead time. The overhead time of a program is measured by the wall clock time that the program requires to process an input file that contains only one read. This is a constant contribution to the run time that we ignore in our analysis because DIAMOND is intended to run on data sets that are much larger than our test data sets. For DIAMOND-fast, this is 5 min, and for DIAMOND-sensitive, this is 20 min. The reason we tested data sets that are small by practical standards was to allow us to run a full BLASTX analysis of each data set in a reasonable amount of time.

Sensitivity on the level of queries (percentage of queries mapped) is based on the number of reads for which BLASTX and the test method both find at least one common alignment against some reference sequence, divided by the total number of reads for which BLASTX finds at least one alignment. Sensitivity on the level of matches (percentage of matches recovered) is based on the number of alignments found both by BLASTX and the test method, divided by the total number of alignments found by BLASTX.

Principal coordinates (PCoA) analysis. To illustrate that minor differences between the output of BLASTX and DIAMOND do not affect the results of higher-level analyses, we used 12 random subsamples, each containing 200,000 reads, from 12 published permafrost samples[14]. Each subsample was aligned against NCBI-nr using both BLASTX and DIAMOND-fast,

© 2015 Nature America, Inc. All rights reserved.

doi:10.1038/nmeth.3176

keeping alignments with a minimum bit score of 50. Reads were mapped to KEGG Orthology (KO) numbers[3], for each read using the best alignment for which a KO number is known. We then computed Bray-Curtis distances from the resulting profiles and used PCoA to generate diagrams (**Supplementary Fig. 3**).

Memory usage and compatibility. The memory management of DIAMOND is designed to allow for an adaptable memory footprint that does not depend on the total size of the input. DIAMOND breaks down the input query and reference data into fixed size blocks of B sequence letters to be compared against each other at a time. With a seed index entry being 8 bytes long and the index being processed in C chunks, the total memory usage is then given by $2(B + 8B/C + const)$, where const represents a constant amount of overhead memory. With a default value of $C = 4$, the memory usage of the program is bounded by $6B + const$ irrespective of the total size of the database and queries. The block size B can be arbitrarily chosen by the user to fit the target machine. The command line options for B and C are -b and -c, respectively.

To explore the effect of the block size parameter on the performance, we aligned a query set of 35 million Illumina reads from permafrost against the NCBI-nr database containing 9 billion letters using different values of B (**Supplementary Table 2**). This operation requires a high memory server for maximum performance but can be efficiently handled by a machine with 16 GB of memory at about half the speed. This amount of RAM is readily available at a price of $160 on a standard desktop computer.

To ensure the compatibility of the program, we conducted tests on various systems using the NCBI-nr database, downloaded in September 2014, and a query set of 61 million Illumina reads from the Human Microbiome Project (**Supplementary Table 3**).

Code availability. DIAMOND v0.4.7 source code is available in **Supplementary Software** and at http://ab.inf.uni-tuebingen.de/software/diamond.

19. Wheeler, D.L. *et al. Nucleic Acids Res.* **36**, D13–D21 (2008).
20. Boncz, P., Manegold, S. & Kersten, M.L. *Proc. VLDB Conf.* **99**, 54–65 (1999).
21. Hach, F. *et al. Nat. Methods* **7**, 576–577 (2010).
22. Rognes, T. *BMC Bioinformatics* **12**, 221 (2011).
23. Henikoff, J.G. & Henikoff, S. *Methods Enzymol.* **266**, 88–105 (1996).
24. Zhu, W., Lomsadze, A. & Borodovsky, M. *Nucleic Acids Res.* **38**, e132 (2010).

© 2015 Nature America, Inc. All rights reserved.

arXiv:1511.08753v1 [q-bio.GN] 25 Nov 2015

MetaScope - Fast and accurate identification of microbes in metagenomic sequencing data

Benjamin Buchfink[1], Daniel H. Huson[1,2] & Chao Xie[2,3]

[1]*Department of Computer Science and Center for Bioinformatics, University of Tübingen, Sand 14, 72076 Tübingen, Germany*
[2]*Life Sciences Institute, National University of Singapore, #05-02 28, Medical Drive Singapore 117456 Singapore*
[3]*Human Longevity Incorp., Singapore*

MetaScope is a fast and accurate tool for analyzing (host-associated) metagenome datasets. Sequence alignment of reads against the host genome (if requested) and against microbial Genbank is performed using a new DNA aligner called SASS. The output of SASS is processed so as to assign all microbial reads to taxa and genes, using a new weighted version of the LCA algorithm. MetaScope is the winner of the 2013 DTRA software challenge entitled "Identify Organisms from a Stream of DNA Sequences".

Metagenomics is the study of microbes using DNA sequencing[1]. One major area of application is the human microbiome[2] with the aim of understanding the interplay between human-associated microbes and disease. Other areas of metagenomic research include water[3], waste-water treatment[4], soil[5] and ancient pathogens[6]. Another envisioned area of application metagenomics is in bio-threat detection, for example, when a group of individuals becomes infected by an unknown agent and the goal is to quickly and reliably determine the identity of the pathogens involved.

In 2013 the Defense Threat Reduction Agency (DTRA) sponsored an algorithms competition entitled "Identify Organisms from a Stream of DNA Sequences" with a one million dollar prize. Proposed solutions "must generate equivalent identification and characterization performance regardless of the sequencing technology used" and "... must achieve this in a timeline that is substantively shorter than possible with currently available techniques." The challenge provided nine test datasets for analysis and proposed results were scored based on the correctness of organisms identified (organisms score), reads assigned (reads score) and genes identified (genes score). This paper describes the winning entry.

Such analysis requires the comparison of a large number of sequencing reads (typically millions of reads) against a large reference database (typically many billions of nucleotides or amino acids). Hence, tools that address this type of problem must be very fast. Because current reference databases only represent a small fraction of the sequence diversity that exists in the environment[7], such tools must also be very sensitive.

MetaScope performs very fast and very accurate analysis of metagenome datasets, includ-

ing the removal of host reads, if required. MetaScope employs a new fast and sensitive DNA aligner called SASS. The aligner is first used to compare a given set of metagenomic sequencing reads against a host genome, if available, so as to discard reads that probably come from the host genome. The remaining reads are then compared against microbial Genbank[8] using SASS. A second program called Analyzer processes the output of SASS and maps the reads to taxa and genes using a novel variant of the LCA algorithm[9]. The output is written in XML and can, for example, be loaded into the metagenome analysis program MEGAN[10] for further processing.

Like BLAST[11], SASS uses a seed-and-extend approach to alignment. To achieve both high speed and high accuracy, SASS uses spaced seeds[12,13], a hash-table for seed lookup and is implemented as a parallel algorithm in C++[14]. A crucial heuristic decision is when to extend a given seed match so as to compute a full alignment. SASS uses Myers' bit vector algorithm[15] and a gain-based termination criterion to decide this. In the context of host-genome filtering, the score obtained in this way is used as a proxy for the full local alignment score and the extension phase is not used.

Removal of all reads that align to the host genome does not completely resolve the problem of false positive taxon assignments because many viruses and vector sequences in Genbank contain human sequences. Hence, in a preprocessing step, we use SASS to compare the viral and vector portion of Genbank against the human genome and then mask every region of the reference sequences that has a significant alignment to some host sequence.

The assignment of reads is often performed using the naive LCA algorithm[9] in which a read is placed on the lowest-common ancestor of all taxa in the NCBI taxonomy for which the read has a high-scoring alignment to a corresponding sequence in the reference database. As the naive LCA algorithm analyses each read in isolation, in the presence of many similar reference sequences from different species, the result is often very unspecific placement of reads. To overcome this, MetaScope uses a new weighted LCA algorithm that proceeds in two steps. First, the naive LCA algorithm is used to assign a weight to each reference sequence, reflecting the number of reads that are assigned to the corresponding species and have a significant alignment to that reference sequence. Then each read is placed on the taxon node that covers 75% (by default) of the total weight of all reference sequences that have a significant alignment with the read.

MetaScope predicts genes based on which annotated genes the alignments of a read overlap with. A read will often align to many different reference sequences and so the potential number of genes to report for a single read can be quite large, containing many false positives. To address this, all genes that are partially overlapped by some alignment of a read are ranked by the weight of the corresponding reference sequence and by the proportion of the gene sequence covered by any reads, and a small number of top ranked genes are reported.

The results obtained by MetaScope on the nine DTRA datasets are listed in Table 1. Slightly

different algorithmic parameters are used based on the different sequencing platforms, as described in the Materials section. The accuracy score ranges from $90.1 - 98.7\%$ and the run time ranges from 4 to 13 minutes per dataset.

We have also investigated the use of an intermediate assembly step (except for PacBio reads). In more detail, all reads that did not align to the host genome (human) were presented to the Newbler assembler[16] as input. The obtained contigs and all unassembled reads were aligned against microbial Genbank using SASS. The output of this was then processed as described above and all assembled reads inherited the taxon assignment of their containing contigs. The results produced by this approach scored equally high as those reported in Table 1, but not better, so we did not pursue this further.

We plan to make Metascope freely available from `http://www.metascope.net`.

Methods

Overview MetaScope is able to accurately analyze millions of sequencing reads in minutes. The MetaScope pipeline takes a file of sequencing reads in FastQ or FastA format as input and produces a report file in XML format as output. The input file represents an host-associated metagenome sample and the aim of MetaScope is to determine the taxonomic and functional content of the non-host portion of the sample. For each organism detected in the input file, the report file contains an estimation of its relative amount, the list of all reads assigned to the organism and the list of all genes identified for the organism.

The MetaScope pipeline (see Figure 1) is invoked using the command `metascope` *platform reads work output*. The four arguments specify the sequencing platform (one of Illumina, 454, IonTorrent or PacBio), an input file containing all reads in FastQ format, a sample-specific working directory where intermediate files are to be placed, and the name of the output file. In addition, any part of the pipeline can be run individually with more control over programming parameters.

First, the program SASS is used to compare all reads in the input file against the host genome. All detected alignments between reads and the host genome are written to a file called `host.m8`. Second, a script called `triage` uses the alignments in `host.m8` to count the host reads and to write all non-host reads to a file called `non-host.fq`. Third, SASS compares the set of all non-host reads against Genbank. All found alignments are written to a file called `metagenome.m8`. The number of reads and host reads, and the file of all metagenome alignments are provided as input to the MetaScope `analyzer`, which produces the final MetaScope report `output.xml` in XML format.

(1) Name	(2) Sequencing platform	(3) Number of reads	(4) Average length	(5) Seq. acc.	(6) Total score	(7) Org. score	(8) Reads score	(9) Genes score	(10) Time (mins)
Testing1	PacBio	92 948	1883	83	90.074	100	85	85	7:48
Testing2	PacBio	98 323	1837	83	98.747	100	98	98	8:24
Testing3	Ion-Torrent	379 028	160	98	91.949	85	93	96	6:28
Testing4	Roche-454	399 671	363	99	91.595	75	99	99	6:47
Testing5	Illumina	5 550 655	150	100	91.538	93	99	82	6:14
Testing6	Illumina	6 038 557	150	100	95.357	100	100	86	7:27
Testing7	Ion-Torrent	323 028	159	98	92.258	83	99	94	4:20
Testing8	Roche-454	351 799	263	99	96.843	100	100	90	4:49
Testing10	Illumina	6 164 558	151	100	97.803	100	100	93	12:10

Table 1: Nine DTRA challenge human-associated DNA sequencing datasets (1–5), percentage scores achieved by MetaScope (6–9), and time required (10). Sequencing accuracy (5) is the average percent identity of read alignments to the human reference genome. The total score (6) is the average of the three component scores (6–8).

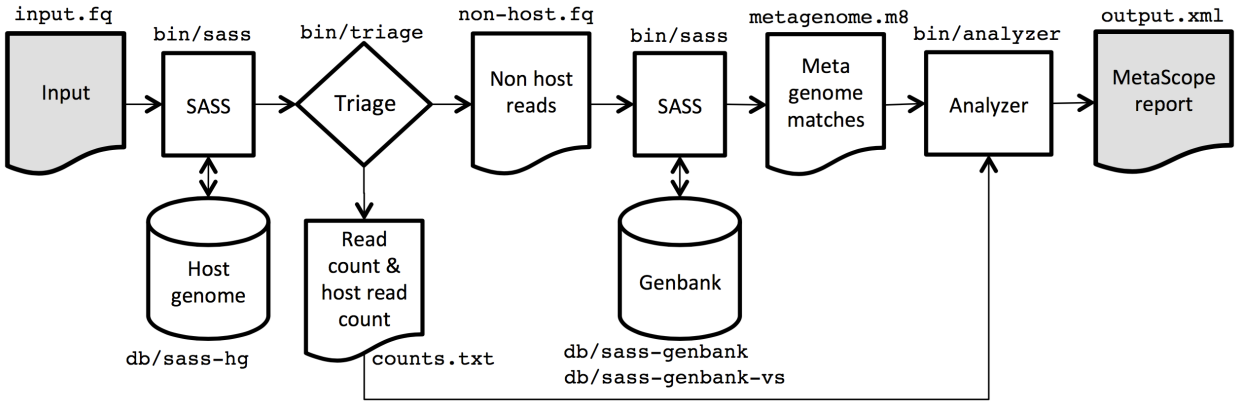


Figure 1: Overview of the MetaScope pipeline. Data moves from left to right. The input reads are compared against the host genome using SASS, then all non-host reads are compared against Genbank using SASS, and finally, the result of the second comparison is processed by the MetaScope Analyzer program to generate the final MetaScope XML report.

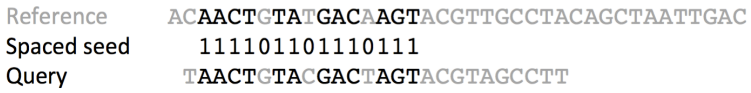


Figure 2: Illustration of the application of a spaced seed to match letters between a reference and a query sequence. Ones and zeros indicate positions to use and ignore, respectively

Sequence alignment The main computational bottleneck in metagenome analysis is the comparison of the reads against a host database, in the case of a host-associated sample, and then against a comprehensive collection of bacterial and viral sequences, such as Genbank[8]. To address this problem in an efficient manner, MetaScope introduces a new sequence alignment tool called SASS (an acronym for "sequence alignment using spaced seeds"). SASS is written in C++ and uses SeqAn[14] and Boost[17].

Designed to target significant alignments with a bit score of at least 50, SASS aligns DNA sequencing reads at about 50-100 times the speed of discontiguous MegaBLAST[13]. Like BLAST, SASS attempts to exhaustively determine all significant alignments, which is crucial for accurate taxonomic analysis.

Fast pairwise alignment programs usually follow the seed-and-extend paradigm[11]. In this two-phase approach, first one searches for exact matches of small parts of the query sequence in the reference database, such seed matches are evaluated and those deemed promising are then followed up in an extend phase that aims at computing a full alignment.

Existing approaches typically employ an index data structure for the reference database in order to quickly compute all seed matches between the query sequence and the reference. For example, Bowtie2[18] and BWA[19] use a compressed FM-index, which is very memory efficient, but at the expense of a slower access time. In contrast, SASS uses a hash table, which requires more memory, but is much faster. The high speed of the index allows SASS to achieve high speed and good sensitivity even when aligning low quality reads such as produced by PacBio and Ion Torrent sequencers.

Most aligners employ a simple seed shape that consists of a short word of consecutive positions. The choice of seed length is based on a trade-off between sensitivity and speed. A hash table index permits the use of spaced seeds. These are longer seeds in which only a subset of positions are used (see , as Figure 2). The number and exact layout of the utilized positions are called the weight and shape of the spaced seed, respectively. Spaced seeds are known to perform better than simple seeds in terms of the speed/sensitivity trade-off[12,13]. By default, SASS uses a single spaced seed, `111110111011110110111111`[20].

To sustain the high throughput achieved in the seeding phase, we attempt to avoid unnecessary Smith-Waterman computations in the extension phase. To this end, we evaluate seed matches

using a modified version of Myers' bit vector algorithm for approximate string matching[15], which computes the edit distance between two short patterns encoded in machine words using fast bit-parallel operations. Starting from the location of a seed match, an alignment is extended in both directions by block-wise invocation of Myers' algorithm in conjunction with a termination criterion based on the score gain. Tentative scores are calculated that approximate the full BLAST score. A full banded Smith-Waterman alignment[21] is only performed on the 100 (by default) best tentative alignments for a read, thus producing accurate standardized BLAST alignment scores for them.

In the case of a host-associated sample, the first step is to identify all reads that come from the host organism. To address this, SASS is used to compare all reads against the host genome. For human, we used assembly release CRCh37 downloaded from NCBI in June 2013. The output of SASS is written to a file called `host.m8`. To reduce the running time of this calculation, here we compute only the approximated score for any alignment and do not perform a full Smith-Waterman calculation.

Based on the host alignments detected in the previous step, a simple Perl script called `triage` is then used to determine all reads that do not have a significant alignment to the host genome and only these reads are considered in the downstream analysis. Here, an alignment is considered significant if it has an expected score of less than 10^{-10}. These reads are placed in a file called `non-host.fq`. An additional file, `counts.txt` is generated that contain the total number of reads and the number of reads that have a significant alignment to the host genome.

Then SASS is used to compare all non-host reads reads (contained in `non-host.fq`) against a large portion of Genbank (consisting of all bacterial, viral, phage and synthetic sequences), downloaded from NCBI in June 2013. The resulting alignments are placed in a file called `metagenome.m8`.

SASS uses two different indices for Genbank, depending on the quality and quantity of the sequencing reads. For high quality and high quantity input samples SASS uses an index that is optimized for speed (using longer seeds) whereas for samples of lesser sequencing quality and smaller size, SASS uses an index that is optimized for sensitivity (using shorter seeds).

The files `counts.txt` and `metagenome.m8` form the basis of MetaScope's taxonomy and gene content analysis, as described in the following sections.

Taxonomic analysis The number of reads and host reads, and the file of all metagenome alignments obtained using SASS are provided as input to the MetaScope `analyzer`, a Perl script that produces the final MetaScope report `output.xml` in XML format.

The analyzer uses three criteria to decide which alignments are deemed significant and all

non-significant alignments are ignored in all subsequent analysis steps. The first criterion is a minimum alignment bit score (option `minscore`, default is 50). Second, for each read we only consider alignments that have maximal bit score, or that are within $x\%$ of the top score, where x is set by a user option called `top`.

The third criterion, which is only applied to Illumina reads, aims at ensuring that a significant alignment covers a large proportion of the corresponding read. Because the quality of an Illumina read tends to degrade toward the end of the read, we calculate the proportion of read covered as alignment length divided by "covered prefix length", where the latter is the length of the prefix of the read up to the last base that is covered by the alignment. In more detail, an alignment must fulfill $1 - P_s/P_e <$ `minover` to be deemed significant, where P_s and P_e are the alignment start and end position on the read and `minover` is a user-specified parameter (default is 0.8).

Weighted LCA The assignment of reads to taxa based on a set of alignments to a reference database is often performed using the naive LCA algorithm[9] in which a read is placed on the lowest-common ancestor of all taxa in the NCBI taxonomy for which the read has a high-scoring alignment to a corresponding sequence in the reference database. This approach is quite conservative and does not work well when there are multiple closely related references in the database, as these will move the assignment to higher level on the phylogenetic tree.

To overcome this, MetaScope uses a new weighted LCA algorithm that proceeds in two rounds. In the first round, the naive LCA is applied to all reads. During this process, each reference sequence is assigned a weight that is the number of reads that have a significant alignment to that reference sequence and for which the naive LCA assigns the read to the same species that the reference sequence has. Reference sequences that are not assigned a weight in this way are assigned weight 1.

In the second round, each read is then assigned to the lowest taxonomic node that lies above a fixed proportion (user parameter `lca` default value 0.8) of the sum of weights of reference sequences to which the read has a significant alignment. The lowest rank that we consider here is that of species. Reads that are assigned to a sub-species or strain are moved up to the species level.

To address the problem of over-aggressive taxa assignment, for each assigned taxa node, we calculate and report the average alignment identity between the reads assigned to this node and the reference sequences. If the average identity is below 90% for a species level taxa node, a `low_identity` tag is reported in the XML output to indicate that a species-level assignment might be too aggressive.

Strain level assignment Our implementation of the weighted LCA assigns reads down to the level of species, but not further. If the user requests strain-level analysis (option `strain`) then the `analyzer` proceeds as follows. For each read that is assigned to a species node, we consider all alignments whose bit score are within x percent of the best score for the read, where x is determined by a user parameter `strain_top` (default value 10%). If a significant proportion (controlled by a user parameter `strain_lca`, default 80%) of the best alignments agree on a strain and these alignments have high sequence identity (controlled by parameter `strain_iden`, default 95%), then the read is tentatively assigned to that strain. A strain is reported, if a significant proportion (controlled by parameter `strain_report`, default 80%) of the reads previously assigned to the species are tentatively assigned to the strain.

Gene prediction To decide which genes to report for a given read, MetaScope produces two separate lists of all genes that are partially covered by an alignment of the read. The first list is ranked by descending weight of reference sequence (as described above) and the second is ranked by descending coverage of genes (that is, by the number of bases of the gene covered by any significant alignment of any read). By default, Metascope reports the top five genes (user parameter `maxgene`) from each of the two ranked lists.

Supporting data generation

Data source All supporting data for MetaScope were downloaded from NCBI. The URLs for the data source are as follows:

- Human genome data: `ftp://ftp.ncbi.nlm.nih.gov/genomes/H_sapiens/`
- GenBank data in ASN.1 format: `ftp://ftp.ncbi.nih.gov/ncbi-asn1/`
- GenBank data in Genbank format: `ftp://ftp.ncbi.nih.gov/genbank/`
- Taxonomy data: `ftp://ftp.ncbi.nlm.nih.gov/pub/taxonomy/`

Masking Human-like reference sequences Because virus and synthetic constructs often contain human sequences, we decided to mask all human-like regions in those sequences in our working version of the GenBank microbial database. To do this, we built a SASS index for the human genome reference in sensitive mode (SASS option `--index-mode 2`). All virus and synthetic construct sequences were then shredded into 100bp fragments with 50bp overlap. We used SASS to align the shredded sequences against the human reference. If a shredded sequence aligned to the human genome with an alignment of 50 bases or more, and at least 80% identity, then the source

region of the shredded sequence covered by the alignment was masked by replacing all nucleotides by N's.

Data preprocessing DNA sequences were first extracted from GenBank ASN.1 data. Only sequences under GenBank BCT, VRL, PHG, and SYN sections were included. The four sections cover all GenBank sequences from bacteria, archaea, virus, phage, and synthetic constructs.

A mapping of GI numbers to NCBI taxon identifiers was extracted from the GenBank ASN.1 files. We also extracted taxonomic lineage information for each reference sequence from these files, rather than from the NCBI taxonomy dump file because only the ASN.1 files contain the correctly labeled strain description of reference sequences. The NCBI taxonomy file was used to complement the ASN.1-derived taxonomy data.

Information on protein coding regions, such as location, protein accession number, locus tag, description, was extracted from the GenBank flat files.

The set of scripts used to download process all reference data is distributed in the `aux` folder of the MetaScope package.

Parameters The output of different sequencing platforms varies in three main aspects, namely the number of reads produced, the average read length and the sequence quality. Individual Illumina datasets usually consist of millions of reads with a read length of hundreds of base pairs. Roche-454 datasets usually have less than one million reads, with a read length approaching 1000 bp. Ion Torrent datasets contain hundreds of thousands of reads, hundreds of base pairs long, with a lower level of quality than the afore mentioned datasets. Finally, PacBio datasets are usually smaller yet, with read lengths of thousands of base pairs, with a very high level of errors.

To address these differences, MetaScope uses slightly different parameter settings depending on which sequencing platform was used to generate the input (see Supplement Table 2). For datasets with higher error rates and smaller size, the pipeline uses SASS' sensitive Genbank index so as to improve the detection of significant alignments in the presence of sequencing errors. Moreover, in the taxonomic analysis of such data, the pipeline employs a relaxed LCA with a `top` setting of 10% so as to help avoid unreliable placement of reads for PacBio data, but 5% for other platforms. Because Illumina datasets usually contain millions of reads, here a even small sequence error rate can lead to a large number of wrongly assigned reads. Hence, for Illumina we use a `minover` setting of 0.9 to ensure that significant alignments cover at least 90% of the high quality end of a read.

The MetaScope parameters employed in the DTRA challenge differ slightly from the default settings recommended in Table 2 due to the specific nature of the DTRA testing datasets. Their

metagenomic reads appeared to have originated from organisms whose genome sequences are well represented in Genbank and thus they usually have a top-scoring alignment to the correct species (but also to many others). In this situation, we were able to set `top` to 0 for all sequencing technologies except for PacBio, were 0.1 was used to accommodate for the high rate of sequencing errors in PacBio data.

The default value for the `maxgene` parameter (that controls the number of genes reported per read) is 5, as this value works well on all DTRA challenge datasets. However, for the DTRA challenge Roche-454 datasets we used a value of 1 so has to achieve a particularly high gene score so as to compensate for low organisms scores on the Roche-454 test datasets.

Acknowledgements

This research is partially supported by the National Research Foundation and Ministry of Education Singapore under its Research Centre of Excellence Programme.

Author contributions

All authors contributed equally to the development and implementation of the described software.

Bibliography

1. Handelsman, J. Metagenomics: Application of genomics to uncultured microorganisms. *Microbiology and Molecular Biology Reviews* **68**, 669–685 (2004).
2. Turnbaugh, P. J. *et al.* The Human Microbiome Project. *Nature* **449**, 804–810 (2007).
3. Rusch, D. B. *et al.* The Sorcerer II Global Ocean Sampling expedition: northwest Atlantic through eastern tropical Pacific. *PLoS Biol* **5**, e77 (2007). URL `http://dx.doi.org/10.1371/journal.pbio.0050077`.
4. Albertsen, M., Hansen, L. B., Saunders, A. M., Nielsen, P. H. & Nielsen, K. L. A metagenome of a full-scale microbial community carrying out enhanced biological phosphorus removal. *ISME J* (2011). URL `http://dx.doi.org/10.1038/ismej.2011.176`.
5. Mackelprang, R. *et al.* Metagenomic analysis of a permafrost microbial community reveals a rapid response to thaw. *Nature* **480**, 368–371 (2011).
6. Schuenemann, V. J. *et al.* Targeted enrichment of ancient pathogens yielding the ppcp1 plasmid of yersinia pestis from victims of the black death. *Proceedings of the National Academy of Sciences* **108**, E746–E752 (2011). URL `http://www.pnas.org/content/108/38/E746.abstract`. `http://www.pnas.org/content/108/38/E746.full.pdf+html`.
7. Wu, D. *et al.* A phylogeny-driven genomic encyclopaedia of bacteria and archaea. *Nature* **462**, 1056–1060 (2009).
8. Wheeler, D. L. *et al.* Database resources of the National Center for Biotechnology Information. *Nucleic Acids Res* **36**, D13–D21 (2008). URL `http://dx.doi.org/10.1093/nar/gkm1000`.
9. Huson, D. H., Auch, A. F., Qi, J. & Schuster, S. C. MEGAN analysis of metagenomic data. *Genome Res* **17**, 377–386 (2007). URL `http://dx.doi.org/10.1101/gr.5969107`.

10. Huson, D. H., Mitra, S., Weber, N., Ruscheweyh, H.-J. & Schuster, S. C. Integrative analysis of environmental sequences using MEGAN 4. *Genome Research* **21**, 1552–1560 (2011).

11. Altschul, S. F., Gish, W., Miller, W., Myers, E. W. & Lipman, D. J. Basic local alignment search tool. *Journal of Molecular Biology* **215**, 403–410 (1990).

12. Burkhardt, S. & Kärkkäinen, J. Better filtering with gapped q-grams. *Fundamenta Informaticae* **XXIII**, 1001–1018 (2001).

13. Ma, B., Tromp, J. & Li, M. PatternHunter: faster and more sensitive homology search. *Bioinformatics* **18**, 440–445 (2002).

14. Döring, A., Weese, D., Rausch, T. & Reinert, K. SeqAn an efficient, generic C++ library for sequence analysis. *BMC Bioinformatics* **9**, 11–11 (2008).

15. Myers, E. A fast bit-vector algorithm for approximate string matching based on dynamic programming. *Journal of the ACM* **46**, 395–415 (1999).

16. Margulies, M. & *et al.* Genome sequencing in microfabricated high-density picolitre reactors. *Nature* **437**, 376–380 (2005).

17. Siek, J., Lee, L.-Q. & Lumsdaine, A. Boost random number library. http://www.boost.org/libs/graph/ (2000).

18. Langmead, B. & Salzberg, S. Fast gapped-read alignment with Bowtie 2. *Nat Meth* **9**, 357–359 (2012).

19. Li, H. & Durbin, R. Fast and accurate short read alignment with Burrows-Wheeler transform. *Bioinformatics (Oxford, England)* **25**, 1754–1760 (2009).

20. Ilie, L., Ilie, S., Khoshraftar, S. & Bigvand, A. M. Seeds for effective oligonucleotide design. *BMC Genomics* **12**, 280 (2011).

21. Smith, T. F. & Waterman, M. S. Identification of common molecular subsequences. *JMB* **147**, 195–197 (1981).

Platform	SASS Index	`top`	`minover`
Illumina	normal	0.05	0.9
Ion Torrent	sensitive	0.05	0
Roche 454	normal	0.05	0
PacBio	sensitive	0.1	0
Unspecified	normal	0.05	0

Table 2: Default platform-specific settings used by MetaScope.

BRIEF COMMUNICATION
https://doi.org/10.1038/s41592-021-01101-x

nature methods

Check for updates

OPEN

Sensitive protein alignments at tree-of-life scale using DIAMOND

Benjamin Buchfink[1], Klaus Reuter[2] and Hajk-Georg Drost[1]✉

We are at the beginning of a genomic revolution in which all known species are planned to be sequenced. Accessing such data for comparative analyses is crucial in this new age of data-driven biology. Here, we introduce an improved version of DIAMOND that greatly exceeds previous search performances and harnesses supercomputing to perform tree-of-life scale protein alignments in hours, while matching the sensitivity of the gold standard BLASTP.

Within the next decade, The Earth BioGenome Project[1,2] aims to sequence and assemble the reference genomes for ~1.5 million of the 10–15 million known eukaryotic species that inhabit our planet. Current sequence search algorithms and software tools would be impractical for analyzing data of this magnitude when aiming to retain sensitivity similar to the gold standard BLASTP[3]. In an experimental study we estimated that querying the National Center for Biotechnology information (NCBI) non-redundant (nr) database (~280 million sequences) against the UniRef50 database (~40 million sequences) using BLASTP would require more than 2 months even on a supercomputer equipped with 20,800 cores (Methods). However, the newly developed version of DIAMOND can now accomplish the same task in several hours, with an alignment sensitivity that matches BLAST. We overcome this computational bottleneck and enable sensitive large-scale protein searches on a tree-of-life scale by introducing improved algorithmic procedures and a customized high-performance computing framework, which incorporate optimized distributed computing, double indexing and multiple spaced seeding. This version of DIAMOND is available as Open Source Software under the GPL3 license (http://www.diamondsearch.org).

DIAMOND is a fast and sensitive protein aligner that was initially developed for metagenomics applications to achieve ultra-fast alignments at the cost of alignment sensitivity, compared with the gold standard, BLAST. Although DIAMOND is proven to be practical for many metagenomics studies that also often rely on *k*-mer information for annotation and classification[4], most functional and phylogenomic studies rely heavily on high alignment sensitivity to obtain useful insights about the functional conservation of proteins and their evolutionary divergence along phylogenetic lineages. For data-intensive studies in these fields, BLAST remains the tool of choice due to its paramount alignment sensitivity.

Here, we introduce a greatly improved version of DIAMOND that provides two sensitivity modes, --very-sensitive and --ultra-sensitive, which will enable data-intensive comparative genomics research such as tree-of-life scale tracing of protein evolution[5], gene age inference[6,7], and functional annotation of genes and gene families[8] to be carried out with the same accuracy as BLAST, but with an 80–360-fold computational speedup. In --ultra-sensitive mode, DIAMOND (v2.0.7) achieves this BLAST-like sensitivity milestone while reducing the computational run time of BLASTP-heavy studies from months to hours.

This version of DIAMOND is different from other protein aligners and its older versions in that it focuses on ultra-fast but sensitive protein searches that can scale with sequencing efforts; for example, to meet the demands of the large-scale Earth BioGenome Project and analogous bulk sequencing projects. Alternative tools such as BLASTP (ref. [3]), USearch (ref. [9]), LAST (ref. [10]) or MMSeqs2 (ref. [11]) are also optimized to run fast protein alignments, but still require longer computation times and, with the exception of BLAST, are less sensitive than DIAMOND when dealing with very large datasets. These tools already experience limitations when they try to handle searches at the scale of the NCBI nr database, which currently contains the largest collection of sequences, representing genomic information for ~12,000 eukaryotic species. Therefore we sought to build a protein search infrastructure that can accommodate the demands of sensitive homology searches on this exponentially growing dataset of sequenced species.

DIAMOND now achieves this goal by providing four different levels of alignment sensitivity and by optimizing two distinct computational paradigms. First, it leverages an ultra-fast integration of algorithmic steps optimized for the latest generation of computer architectures that are designed to function optimally when dealing with massive query and subject databases. Second, it harnesses high-performance computing (HPC) and cloud computing by providing a powerful distributed computing implementation customized for large-scale protein searches, incorporating our new DIAMOND search scheme (Methods). In summary, our method is based upon on-the-fly double indexing (in which both the reference database and the query are indexed for comparison) and hash join on the seed space spanned by up to 64 multiple spaced seeds (seeds that are extracted from the sequence according to a pattern of 'match' and 'don't care' positions) to greatly improve the specificity of seeding relative to a baseline strategy. Furthermore, double indexing focuses the comparison operations with respect to a seed and enables the operations to be streamed through the CPU in an efficient, cache-aware manner, avoiding the memory latency bottleneck of a classical single-indexed seed lookup approach. A chain of heuristic filter stages that makes heavy use of vector instructions is designed to gradually eliminate spurious hits, while passing on potentially significant alignments to a vectorized Smith–Waterman extension.

We demonstrate the search capabilities of DIAMOND (v2.0.7) by systematically comparing its performance against BLASTP (v2.10.0) and MMSeqs2 (release 11), and against an older version of DIAMOND (v0.7.12), all of which are currently the most promising alternatives for sensitive tree-of-life scale protein searches (Fig. 1). To create a benchmark dataset covering annotated protein sequences

[1]Computational Biology Group, Max Planck Institute for Developmental Biology, Tübingen, Germany. [2]Max Planck Computing and Data Facility, Garching, Germany. ✉e-mail:

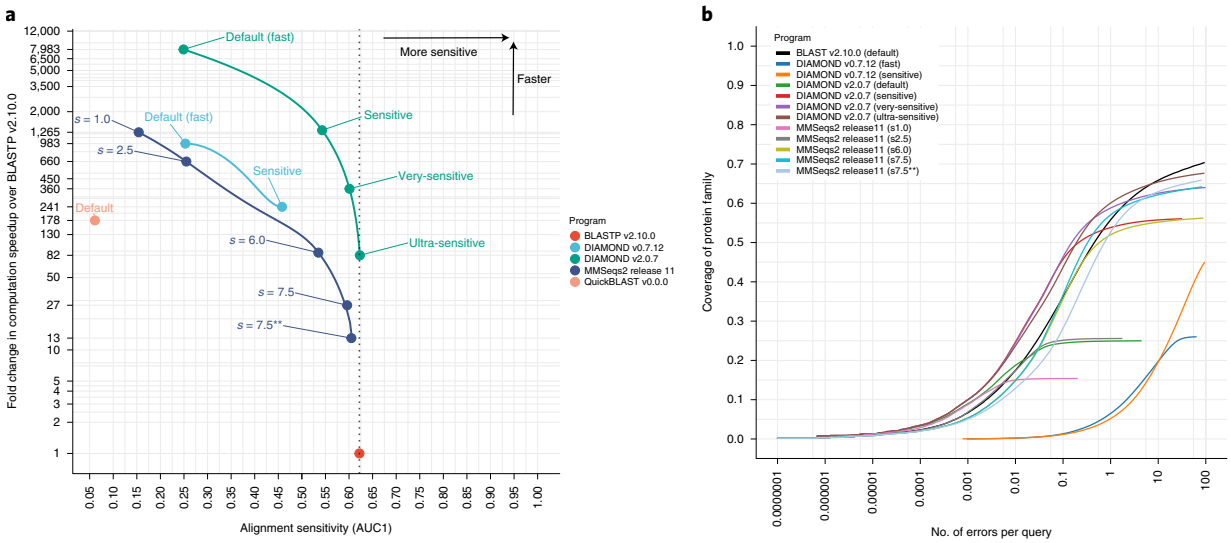


Fig. 1 | Benchmark of DIAMOND, MMSeqs2 and BLASTP using various sensitivity modes. Computational speedup and alignment sensitivity comparisons were carried out between the new version of DIAMOND, v2.0.7 (using default, --sensitive, --very-sensitive and --ultra-sensitive modes), the old version of DIAMOND, v0.7.12 (using default and --sensitive modes), MMSeqs2 release 11 (using modes $s=1.0$, $s=2.5$, $s=6.0$, $s=7.5$, $s=7.5^{**}$ with --max-seqs 100000), BLASTP v2.10.0 and QuickBLAST v0.0.0. **a**, Alignment sensitivity (AUC1) measured as the fraction of the query's protein family covered until the first false positive, averaged over all queries in the benchmark dataset. Dashed vertical line, alignment sensitivity level of BLASTP v2.10.0 (AUC1 = 0.622). **b**, ROC curves of the same benchmark showing the true average error rate per query versus the average coverage of the protein family, depending on the *e*-value threshold.

spanning the full diversity of the tree of life, we downloaded the NCBI nr database (25 October 2019) and annotated each protein sequence in accordance with their SCOPe (structural classification of proteins–extended) domains[12] (http://scop.berkeley.edu/) (Methods). Establishing a ground truth on the basis of SCOP domains has been considered the gold standard for benchmarking protein aligners[13]. As a result of the annotation, we obtained a query dataset of ~1.7 million protein sequences covering ~1,000 representative sequences for each SCOPe superfamily. Furthermore, we annotated the UniRef50 database[14] (accessed 14 September 2019) following the same procedure to serve as a reference database for the benchmark.

It is important to note that some previous performance benchmarks between older versions of DIAMOND and other aligners[15] used small benchmark datasets for the comparison with DIAMOND. As stated earlier, DIAMOND is optimized for searches using large query and reference databases. Therefore valuable benchmarking insights can only be achieved when comparing DIAMOND and other tools using large benchmark datasets, rather than focusing on small query or reference examples.

We ran DIAMOND (v2.0.7) in all four sensitivity modes using our SCOPe-annotated benchmark dataset as a query against the UniRef50 database, and compared its computational performance and level of sensitivity against analogous runs performed with BLASTP (v2.10.0), MMSeqs2 (release 11) and DIAMOND (v0.7.12). Figure 1 shows the benchmarking results of these alignments against the UniRef50 database. For each tool, we show the performance increase of the respective search algorithm over BLASTP against the average recall of a query's protein family until the first false positive (Fig. 1a), and the corresponding receiver operating characteristic (ROC) curve (Fig. 1b). We found that DIAMOND (v2.0.7) computed alignments 12–15-fold faster than MMSeqs2 (release 11) while maintaining similar sensitivity levels. When the new DIAMOND was compared with older versions of DIAMOND[16] (v0.7.12) we observed a 6–8-fold speedup, while the old DIAMOND was also far behind the other benchmarked tools in terms of sensitivity. When comparing DIAMOND (v2.0.7) to BLASTP (v2.10.0) we observed an ~8,000-fold speedup when using the least sensitive mode, and still an 80-fold speedup when running DIAMOND (v2.0.7) with a sensitivity level matching that of BLASTP (ultra-sensitive mode). Closer inspection of the trade-off between sensitivity and specificity on the basis of ROC curves (Fig. 1b) shows that DIAMOND (v2.0.7) in both the very-sensitive and ultra-sensitive modes maintained equal or marginally better sensitivity than BLAST at low error rates, while being only slightly surpassed by BLAST at error rates of >1 false positive per query (in which searches at error rates of >1 have only rare practical applications). We also conclude that the more sophisticated repeat masking used by DIAMOND (v2.0.7) (Methods) enables lower true error rates at similar sensitivity levels.

In addition, we compared older versions of BLASTP (v2.2.31; 2015) to the 2019 version of BLASTP (v2.10.0) and found that the 2019 version of BLASTP was fourfold faster than its 2015 version. Although this speedup is impressive, we are not able to envision a scenario in which this rate of increase will enable tree-of-life scale protein alignments when dealing with sequences from millions of eukaryotic species.

To demonstrate the capabilities of our tool when supported by an HPC infrastructure, we aligned all 281 million protein sequences from the NCBI nr database against the UniRef50 database, which consists of 39 million sequences, using DIAMOND (v2.0.7) in ultra-sensitive mode on the Cobra supercomputer of the Max Planck Society. This comprehensive comparison across all domains of life was computed in less than 18 hours using 520 compute nodes (Fig. 2 and Extended Data Fig. 1), compared with an estimated 2 months with BLAST.

For further evaluation, we report the alignment sensitivity resolved by sequence identity (Extended Data Fig. 2), the distribution of homologs across identity bins (Extended Data Fig. 3), and the results of two supplementary benchmarks based on short reads (Extended Data Figs. 4–7).

Here, we introduce a comprehensive sequence search framework based on an extensively improved version of DIAMOND (v2.0.7) that enables users to handle the accelerating growth of sequence

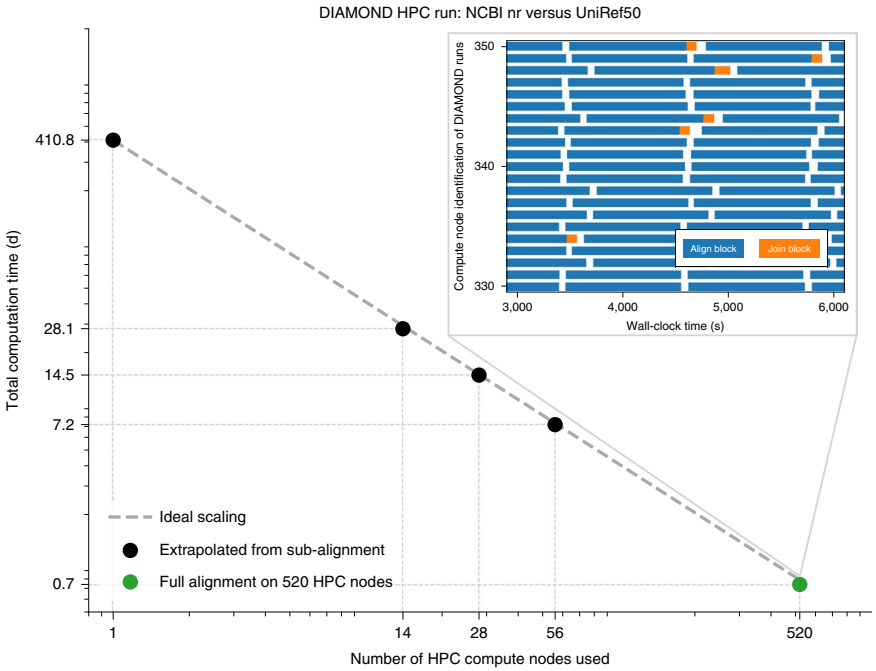


Fig. 2 | Strong scaling of DIAMOND on up to 520 nodes (20,800 cores) of the Cobra supercomputer of the Max Planck Society. The parallel scalability of DIAMOND in a distributed computing environment (supercomputer) is shown. Runs are performed in --ultra-sensitive mode, using the NCBI nr database (~281 million sequences) as the query database and the UniRef50 database (~39 million sequences) as the reference database. The full alignment was performed on 520 nodes (20,800 cores total), finishing in less than 18 hours of wall-clock time. On 1, 14, 28 and 56 nodes, only a subset of the query blocks could be processed, and the time for the full alignment was linearly extrapolated for each node count. An almost ideal scaling is observed, with a small super-linear effect caused by the caches of the parallel input-output system. To illustrate the massive parallelism, the inset shows a zoomed-in view of the sequence of tasks that DIAMOND (v2.0.0) has performed over time on 20 of the 520 nodes, indicated by blue rectangles (alignment process) and orange rectangles (join operations). White spaces encode the input-output activity on the supercomputer's shared parallel file system (Extended Data Fig. 1).

information for data-driven comparative and functional genomics studies. We designed this framework to meet the computational demands of future high-sensitivity sequence searches, to gain fundamental insights into protein evolution and molecular phylogenetics.

Online content

Any methods, additional references, Nature Research reporting summaries, source data, extended data, supplementary information, acknowledgements, peer review information; details of author contributions and competing interests; and statements of data and code availability are available at https://doi.org/10.1038/s41592-021-01101-x.

Received: 23 July 2020; Accepted: 22 February 2021;
Published online: 7 April 2021

References

1. Lewin, H. A. et al. Earth BioGenome Project: sequencing life for the future of life. *Proc. Natl Acad. Sci. USA* **115**, 4325–4333 (2018).
2. Exposito-Alonso, M., Drost, H.-G., Burbano, H. A. & Weigel, D. The Earth BioGenome project: opportunities and challenges for plant genomics and conservation. *Plant J.* **102**, 222–229 (2020).
3. Altschul, S. F. et al. Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Res.* **25**, 3389–3402 (1997).
4. Wood, D. E. & Salzberg, S. L. Kraken: ultrafast metagenomic sequence classification using exact alignments. *Genome Biol.* **15**, R46 (2014).
5. Caetano-Anollés, G. & Caetano-Anollés, D. An evolutionarily structured universe of protein architecture. *Genome Res.* **13**, 1563–1571 (2003).
6. Capra, J. A., Stolzer, M., Durand, D. & Pollard, K. S. How old is my gene? *Trends Genet.* **29**, 659–668 (2013).
7. Dunn, C. W., Luo, X. & Wu, Z. Phylogenetic analysis of gene expression. *Integr. Comp. Biol.* **53**, 847–856 (2013).
8. Glover, N. et al. Advances and applications in the quest for orthologs. *Mol. Biol. Evol.* **36**, 2157–2164 (2019).
9. Edgar, R. C. Search and clustering orders of magnitude faster than BLAST. *Bioinformatics* **26**, 2460–2461 (2010).
10. Kiełbasa, S. M., Wan, R., Sato, K., Horton, P. & Frith, M. C. Adaptive seeds tame genomic sequence comparison. *Genome Res.* **21**, 487–493 (2011).
11. Steinegger, M. & Söding, J. MMseqs2 enables sensitive protein sequence searching for the analysis of massive data sets. *Nat. Biotechnol.* **35**, 1026–1028 (2017).
12. Fox, N. K., Brenner, S. E. & Chandonia, J.-M. SCOPe: Structural Classification of Proteins–extended, integrating SCOP and ASTRAL data and classification of new structures. *Nucleic Acids Res.* **42**, D304–D309 (2014).
13. Brenner, S. E., Chothia, C. & Hubbard, T. J. Assessing sequence comparison methods with reliable structurally identified distant evolutionary relationships. *Proc. Natl Acad. Sci. USA* **95**, 6073–6078 (1998).
14. Suzek, B. E., Huang, H., McGarvey, P., Mazumder, R. & Wu, C. H. UniRef: comprehensive and non-redundant UniProt reference clusters. *Bioinformatics* **23**, 1282–1288 (2007).
15. Hauser, M., Steinegger, M. & Söding, J. MMseqs software suite for fast and deep clustering and searching of large protein sequence sets. *Bioinformatics* **32**, 1323–1330 (2016).
16. Buchfink, B., Xie, C. & Huson, D. H. Fast and sensitive protein alignment using DIAMOND. *Nat. Methods* **12**, 59–60 (2015).

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this license, visit http://creativecommons.org/licenses/by/4.0/.
© The Author(s) 2021

Methods

Algorithmic overview of DIAMOND. *Double indexing.* DIAMOND uses the double-indexing approach, in conjunction with multiple spaced seeds[17], to optimize the handling of large query and large reference databases. In the first step, tables of seed–location pairs are built for query and reference sequences. Next, matching seeds are computed using a hash join technique that conducts recursive radix clustering of both tables until a hash table for the query data fits into the cache, at which point the rest of the join is computed by hashing[18]. We found this approach to be faster than the sorting method used by older versions of DIAMOND[16], especially given that a full sorting of the reference table is avoided for smaller query datasets.

The double-indexing algorithm is designed to be cache aware, given that the data associated with one seed need to be loaded for comparison from memory only once, while the classical index-based linear seed lookup suffers from poor data locality. Additionally, our on-the-fly indexing method enables efficient use of multiple spaced seeds by processing the shapes one at a time and not requiring the index tables for all shapes to be present in memory simultaneously, while also avoiding expensive seed lookups through our cache-friendly hash join implementation.

DIAMOND (v2.0.7) uses two seed shapes of weight 10 for its fast mode, 16 shapes of weight 8 and 14 shapes of weight 7 for its sensitive and very-sensitive modes, respectively, and 64 shapes of weight 7 for its ultra-sensitive mode. The seed shapes were computed using SpEED[19]. Even when operating with 64 shapes, the run time generation of the indices, together with the join computation, take up less than 1% of the total run time of the program. When processing the NCBI nr database, the total size of these indices would be 123 billion letters × 9 bytes per entry × 64 shapes, which is ~64 TB if kept in memory or written to disk, while DIAMOND (v2.0.7) requires less than 16 GB of RAM when running in ultra-sensitive mode. This shows that DIAMOND does not require expensive computing infrastructures and can be operated with modest hardware resources if needed. Because of the runtime indexing, DIAMOND maintains disk-based database files that contain only the reference sequences, and can optionally also use BLAST databases (since v2.0.8).

Hamming distance filter. In the first stage of the sequence comparison process, a hamming distance computation between a query sequence and a subject sequence is performed at all seed hit locations in a 48-letter window encompassing the hit. We optimized this procedure using a chain of SSE (streaming single-instruction multiple-data (SIMD) extensions) pcmpeqb, pmovmskb and popcnt instructions to achieve a tenfold decrease in computation time compared with an ungapped alignment incorporating a scoring matrix, while reducing the number of hits by 1–2 orders of magnitude. A sensitivity-level-dependent cut-off for the hamming distance that can also be manually set by the user determines whether a hit is passed to the next filter stage.

We further extend our initial approach, introduced in the original version of DIAMOND[16], and maximize the filtering throughput by using a loop-tiling strategy to incorporate the cache hierarchy and address the fact that the data associated with a single seed may exceed the cache capacity in the new very-sensitive and ultra-sensitive modes of DIAMOND (v2.0.7). We also load the 48-letter windows at the query and subject locations into linear buffers prior to running the all-versus-all hamming distance computation, to make best use of the hardware prefetcher and to avoid any random memory access.

Ungapped extension. After the hamming distance stage, the next step in the pipeline computes ungapped extensions at the seed hit locations. This procedure is vectorized using AVX2 instructions, aligning one query against up to 32 subject sequences. After 32 subject sequences are loaded into AVX2 registers, a 32 × 32 byte matrix transposition is computed using a series of 160 unpack instructions, such that 32 letters of different subjects are interleaved into one SIMD register, and the match scores can be loaded along the query. A sensitivity-level-dependent *e*-value threshold determines the hits that will be passed to the next stage.

Leftmost seed filter. Due to its double-indexing algorithm, DIAMOND may find the same alignment multiple times independently during the search stage. These redundant hits need to be filtered out to avoid an excessive use of temporary disk space. DIAMOND accomplishes this task by inspecting the local ungapped alignment for seed hits to the left of the hit that is currently being processed, as well as seed hits by previously processed shapes. If such a hit is found, DIAMOND notices the repetition and the current hit is discarded. Given that this procedure entails checking against up to 64 different seed shapes, we further optimized this process by incorporating a precomputed lookup table that stores information on whether any of the processed shapes will hit a given bit-encoded match or mismatch pattern, thus enabling the same check to be performed in one pass over the local hit pattern.

Adaptive ranking. Given that the typical application of an aligner will require the reporting of a certain number of best alignments (hits) for each query (as set on the command line using the --max-target-seqs option), DIAMOND makes use of this parameter to control the computational effort spent on seed extension and avoid having to compute gapped extensions for all seed hits. To this end, after the seed search within target sequences has been concluded, we determine a tentative order of target hits with respect to a single query. In the present case, this ranking procedure uses the ungapped extension scores at seed hits to assign a linear order to the targets. DIAMOND sorts the target list by ungapped extension score (from best to worst) for each target, similar to the way in which MMSeqs2 uses its ungapped extension-derived prefilter scores. Although MMSeqs2 will then compute Smith–Waterman extensions for a fixed number of best targets (as set using the --max-seqs parameter), DIAMOND uses a dynamic criterion to halt evaluation of further targets. We refer to this dynamic approach as adaptive ranking, which improves the DIAMOND reporting accuracy compared with the static criterion used by MMSeqs2, while providing a less biased and more data-adapted filtering procedure. The ranked list is processed in chunks of 400 targets (configurable on the command line using ext-chunk-size), for which extensions are computed. If no extension in the current chunk yields a significant alignment under the user-specified reporting criteria, computation of further extensions for the query is aborted, otherwise the next chunk of targets will be processed.

Gapped extension filter. Given that computing full Smith–Waterman[20] extensions is expensive, we have developed a fast heuristic algorithm designed to estimate a gapped alignment score and discard hits that most probably do not meet the user-set reporting threshold. We use a query profile data structure in the same way as the vectorized Smith–Waterman algorithm introduced by Farrar[21], which is an array for each of the amino acid letters that stores the scores along the query against the given residue. We then use AVX2 instructions to sum up these scores along diagonals of the dynamic programming matrix, thus computing local ungapped extension scores on diagonals. This approach ignores gaps in the alignment and therefore eliminates intra-register data dependencies. With its minimal logic, our heuristic achieves a throughput ~fivefold faster than a Smith–Waterman computation using the vectorized SWIPE method[22]. Nevertheless, ungapped scores on the diagonals can be used to estimate a gapped extension score by thresholding and computing a one-dimensional dynamic program that disregards the location of the diagonal segments. Although this simplifying assumption leads to an overestimation of the true alignment score most of the time, the heuristic is still able to reduce the number of spurious hits by one order of magnitude in the most sensitive alignment mode. If required by the user, this filter step can be disabled using the option gapped-filter-evalue 0.

Chaining. Chaining is the computation of a dynamic program at the level of diagonal segments instead of at the base or residue level, and has been used successfully in DNA alignment tools such as minimap2 (ref. [23]). DIAMOND (v2.0.7) introduces the use of chaining on protein sequences. The result of the chaining computation is used to infer a scaffold for the optimal alignment and to determine the band geometry for a banded Smith–Waterman algorithm[20].

Chaining can be simplified on DNA sequences by considering only diagonal segments of exact matches. However, this is not possible for protein sequences, which makes this computation substantially more elaborate. DIAMOND solves this problem by sorting the diagonal segments obtained by the ungapped extension stage on the starting position in the subject, and constructs a graph in which nodes represent diagonal segments and edges denote diagonal shifts (gaps) by computing pairwise connections between the diagonal segments in one left-to-right pass. Such pairwise connections are then stored as graph edges, incorporating their inbound and outbound coordinates to prevent invalid chains and to allow zigzag connections in which the optimal path repeatedly shifts between the same two diagonal nodes. A red–black tree for the nodes ordered on the diagonal is used to quickly access the most proximal nodes and candidates for determining a connection. For each node, the best score of a local alignment ending in that node is stored, the maximum of which yields the final score estimate and end point for backtracing of the approximate optimal alignment.

Banded SWIPE. The final extensions are computed using a modified version of the vectorized SWIPE (ref. [22]) approach that accommodates banding. Due to their design, both the SWIPE and the 'striped' SIMD vectorization[21] algorithms do not easily allow banded alignment, resulting in the need for an $O(n^2)$ computation in proportion to the length of the query and subject sequences. We vectorize the alignment of a query against up to 32 subjects by overlaying the banded dynamic programming matrix columns of the subjects based on their query ranges (the query coordinate interval $[i_0,i_1]$ that corresponds to a slice of the given column with the subject's band). Given that the bands of the subjects are different, this cannot be fitted perfectly into the register, but reaches a register load efficiency of 80–90% for larger databases. All extensions are computed using 8-bit scores and are repeated when an overflow is detected, unless an alignment score of >255 is already known from previous stages.

Alignments are scored using the BLOSUM62 matrix by default. In addition, we also use a method of composition-based score adjustments[15] that is designed to increase the specificity of the scoring procedure. If required, DIAMOND (since v2.0.6) also supports applying the BLAST compositional matrix adjust scoring procedure[24] to compute BLAST-like alignment scores (options --comp-based-stats[3,4]).

As an alternative, DIAMOND (v2.0.7) also includes the option to compute full-matrix instead of banded Smith–Waterman extensions (command line option --ext full), which are also vectorized using the SWIPE algorithm.

Frameshift alignments. Reads produced by MinION technology[25] are known to be noisy and contain frequent indel errors, a problem that also translates to assemblies derived from such long reads. In consequence, genes cannot be detected reliably on such DNA sequences. DIAMOND addresses this issue by providing frameshift alignments in translated search (blastx) mode. The protein sequences corresponding to all three reading frames of a strand are aligned simultaneously against the target sequence, allowing shifts in the reading frame at any position in the alignment, while incurring a user-defined score penalty (set using -F on the command line). The raw MinION reads and contigs up to the length of full bacterial chromosomes are supported as input in translated search mode, enabling gene discovery and annotation in the absence of known gene boundaries.

Repeat masking. Differentiating between true evolutionary relationships and spurious similarities presents a big challenge in remote homology detection, particularly given the repetitive nature of sequence regions found in many genomes. When dealing with an increasing load of available genomes for tree-of-life scale sequence searches, the ability to differentiate between similarity relationships based on sequence repetitiveness and homology based on a biologically meaningful sequence structure (non-repetitive sequence under purifying selection) becomes crucial to reduce the number of false-positive hits and increase alignment specificity at scale. Masking of low-complexity regions (repeat masking) is the most commonly used strategy to eliminate false-positive hits and to retain only hits found in biologically meaningful homologs. It has been shown that despite using the SegMasker tool included in BLASTP[26], many more and stronger spurious similarities will arise than are expected on random sequences, as defined by an *e*-value threshold parameter[27]. DIAMOND reduces this false-positive bias by using more stringent and more sophisticated masking paradigms based on tantan. If required, the tantan masking can be replaced by the more conservative default BLASTP SEG masking and composition-based statistics using the option --comp-based-stats 3 (ref. [24]).

Distributed-memory parallelization. As part of DIAMOND, our comprehensive sequence search framework supports a distributed-memory parallelization to leverage the computing power of state-of-the-art HPC and cloud-computing resources for massive-scale protein alignments. To this end, both the query database and the reference database are segmented into data packages that we refer to as chunks. The Cartesian product of both query and reference sets defines a (typically large) set of work packages. In the first instance, files containing metadata on these work packages are created centrally before a parallel run is started on independent computing nodes and are subsequently processed in a distributed manner by multiple worker processes of DIAMOND. Usually, only one worker process runs per compute node, efficiently utilizing all of the locally available cores via threads. Unlike related work such as mpiBLAST[28], our implementation does not use any special interprocess communication libraries, such as the message passing interface (MPI) specific to HPC environments, instead it relies on input–output operations supported by any POSIX-compliant parallel file system that is mounted on all of the compute nodes involved. The advantage of this approach is that work packages are distributed in a self-organized way at run time to all participating worker processes using simple file-based stacks located in the parallel file system, with atomic push and pop operations. Once all database chunks for a specific query chunk have been processed, the final worker process involved in the query chunk takes on the role of performing the join operation to ultimately create the output stream. Note that the largest part of the temporary files stays local to a compute node, and only the lightweight work-stack files and the DIAMOND hits from the protein searches are written into the shared parallel file system. This strategy significantly reduces input–output overloads and enables massively parallel processing of DIAMOND runs. In addition to the lack of complex dependencies, such as on MPI, we highlight the particular advantages of our approach. First, there is no designated primary worker to induce a bottleneck due to synchronization, or to act as a potential single point of failure. Second, and by design, worker processes may join and leave at run time, which is less important on classical HPC systems that use batch systems to orchestrate potentially large numbers of processes, but is of striking advantage on elastic cloud-computing resources and on existing commodity resources such as networked laboratory desktop computers. Last, our transactional file-based work-distribution protocol enables fault tolerance, which means that if worker processes die unexpectedly, other processes in a subsequent run can take on and resume their work packages.

Benchmarks. *Main benchmark.* To create a benchmark database, we annotated the 14 September 2019 release of UniRef50 containing 37.5 million sequences with SCOP families. To categorize each protein sequence, we ran SWIPE[22] using an *e*-value cut-off of 10^{-5} against the SCOPe ASTRAL40 v2.07 dataset[12] of domain sequences consisting of 4,850 protein families, which resulted in a collection of 7.74 million annotated protein sequences. We used the hit with the highest bit score per SCOPe fold (a grouping of structurally similar superfamilies) to infer the protein family annotation while allowing multidomain associations.

Given that DIAMOND requires a large query dataset to reach its maximum efficiency, we used an analogous SWIPE approach and annotated the NCBI nr database from 25 October 2019 in accordance with SCOPe families. We used UPGMA clustering[29] on the sets of all protein sequences annotated with the same superfamily to cluster and reduce them to a maximum of 1,000 sequences, which we selected as representatives of that superfamily, resulting in a benchmark dataset of 1.71 million queries.

Both query and reference sequences were locally shuffled in 40-letter windows outside the annotated ranges. All benchmark datasets and annotations have been published[30].

Alignment for all tools was run on an AMD Ryzen Threadripper 2970WX 24-core workstation clocking at 3.0 GHz with 256 GB of RAM, except for the BLASTP (v2.10.0) run, which, due to its run time limitations on a desktop computer workstation, was performed on the Max Planck Society's Draco supercomputer at Garching, Germany, using 24 nodes (32 cores on two Intel Haswell E5-2698v3 chips per node). On the benchmark machine the performance of BLASTP (v2.10.0) was estimated using a random subset of 10,000 queries sampled from the initial benchmark dataset.

For each query, we determined the AUC1 value, defined as the number of alignments against sequences matching the query's protein family, divided by the total number of database sequences of that family (also called the coverage of the protein family). Only hits until the first alignment against a false positive were taken into account, which was defined as the alignment of query and subject sequences from different SCOPe folds. For multidomain proteins, the AUC1 value was averaged over the domains. The AUC1 values of the individual queries were again averaged over the query dataset to obtain the final sensitivity value (Fig. 1a). To ensure that a false positive is contained in the result list of every query, the tools were configured to report all alignments up to an *e*-value of 1,000 (Supplementary Information). Further information about the benchmark design can also be found in the Nature Research Reporting Summary.

Detailed assessment of sequence identities in true-positive alignments. We explored the sensitivity of all compared tools in more detail by resolving it at the level of amino acid sequence identity of true-positive alignments. For this purpose, we define the sequence identity of a query–subject association induced by annotation with the same SCOPe protein family as that obtained from the Needleman–Wunsch alignment between the pair of annotated ranges in the query and subject. Extended Data Figure 2 shows a breakdown of the AUC1 sensitivity for our main benchmark, computed as if the search space of positive cases were restricted to associations of the respective sequence identity ranges. Additionally, Extended Data Fig. 3 shows how a query sequence's family associations are distributed across the identity bins for our benchmark dataset.

Supplementary benchmarks. We report benchmark results for two additional datasets, consisting of sequencing reads from Illumina HiSeq 4000 paired end sequencing (2×150 base pairs) and Illumina HiSeq 2500 paired end sequencing (2×250 base pairs). The datasets were created based on data from a recent rumen metagenome study[31] (Supplementary Information, see Supplementary Benchmark 1) and an environmental study of the topsoil microbiome[32] (Supplementary Information, see Supplementary Benchmark 2). SCOPe-annotated datasets of 1.55 million and 1 million reads, respectively, were obtained as described in the Supplementary Information. The benchmark runs for the two query read datasets were carried out analogously to the run for our main benchmark, operating all tools in translated search mode against the same database of SCOPe-annotated UniRef50 sequences. We report performance, AUC1 values and ROC curves for both runs (Extended Data Figs. 4–7).

Experimental study. The ultimate ambition of DIAMOND v2.0.7 is to provide a comprehensive search framework for sensitive tree-of-life scale protein alignments in the Earth BioGenome Project era and beyond. Although BLAST-like sensitivity levels are the maximally achievable thresholds for pairwise alignments, the next focus of any aligner should be the computational scalability to process millions of sequenced species. With the new --ultra-sensitive mode introduced in DIAMOND v2.0.0 we achieve this critical BLAST-like sensitivity level while maintaining an 80-fold computational speedup, and we achieve an additional near-linear parallel speedup when using the custom DIAMOND HPC implementation. To simulate all facets of a tree-of-life scale protein search that is able to mimic future applications of large-scale comparative genomics projects, we performed DIAMOND --very-sensitive and --ultra-sensitive searches on 520 nodes of the Cobra supercomputer of the Max Planck Society (40 cores on two Intel Skylake 6148 chips, and 192 GB RAM per node), totaling 20,800 computing cores (41,600 threads), using the NCBI nr database (currently storing all sequenced proteins for ~12,000 eukaryotic species and all proteins from ~440,000 genomes of non-eukaryotic species) as the query database, and UniRef50 as the reference dataset. We randomly shuffled the sequences in both FASTA files to avoid a load imbalance due to a biased distribution of sequences in the original files. As a result, DIAMOND v2.0.0 produced 23.1 billion pairwise alignments in the --ultra-sensitive case and 23.0 billion pairwise alignments in the --very-sensitive case, starting from an initial query dataset that contained 281 million sequences

and a reference dataset that contained 39 million subject sequences. In --very-sensitive mode the run terminated in 5.42 hours, while in --ultra-sensitive mode it terminated in 17.77 hours. The latter run is shown in Fig. 2 and Extended Data Fig. 1, demonstrating the massive parallelism achieved on the HPC infrastructure, as shown by the processing of individual tasks over time. Due to the parallel nature of the align and join operations, the parallel speedup is virtually linear and is limited only by the throughput of the shared parallel file system of the supercomputer used. This demonstrates that DIAMOND v2.0.0 can harness its algorithmic improvements and its new HPC support to cover all sequenced species in the tree of life within hours rather than months, while matching the alignment sensitivity levels of BLAST. The uncompressed output generated by this run occupies ~1,100 GB of disk space and stores the 100 best protein hits for each sequence in the NCBI nr database.

We envision that in the future this type of DIAMOND output will be easily accessible to all life scientists via a web application in which users can filter and search for their protein homologs of interest within minutes across the tree of life on a precomputed dataset, instead of having to perform complex data analytics and months' or years' worth of BLAST searches to obtain sensitive protein alignments at this scale.

Reporting Summary. Further information on research design is available in the Nature Research Reporting Summary linked to this article.

Data availability

The sequence and annotation data that support the findings of this study are available in figshare (https://doi.org/10.6084/m9.figshare.c.5053112.v1). The SCOPe ASTRAL40 dataset can be downloaded at http://scop.berkeley.edu/downloads/scopeseq-2.07/astral-scopedom-seqres-gd-sel-gs-bib-40-2.07.fa. The UniRef50 database can be downloaded from ftp://ftp.uniprot.org/pub/databases/uniprot/uniref/uniref50/uniref50.fasta.gz and the NCBI nr database can be downloaded from ftp://ftp.ncbi.nlm.nih.gov/blast/db/FASTA/nr.gz. The sequencing reads of the supplementary benchmarks are part of the samples with European Nucleotide Archive (ENA) accessions SAMEA5383815, SAMEA5383897, SAMEA5383886, SAMEA5383828, SAMEA5383925, SAMEA5383848, SAMEA5383824, SAMEA5383873, SAMEA5384011, SAMEA5383807, SAMEA103892455, SAMEA103892562, SAMEA103892552, SAMEA103892441, SAMEA103892588, SAMEA103892582, SAMEA103892581, SAMEA103892571, SAMEA103892491, SAMEA103892619. Source data are provided with this paper.

Code availability

The source code of DIAMOND v2.0.7 is available at https://github.com/bbuchfink/diamond and in figshare (https://doi.org/10.6084/m9.figshare.14071334.v1).

References

17. Ma, B., Tromp, J. & Li, M. PatternHunter: faster and more sensitive homology search. *Bioinformatics* **18**, 440–445 (2002).
18. Kim, C. et al. Sort vs. Hash revisited: fast join implementation on modern multi-core CPUs. *Proc. VLDB Endow.* **2**, 1378–1389 (2009).
19. Ilie, L., Ilie, S. & Bigvand, A. M. SpEED: fast computation of sensitive spaced seeds. *Bioinformatics* **27**, 2433–2434 (2011).
20. Smith, T. F. & Waterman, M. S. Identification of common molecular subsequences. *J. Mol. Biol.* **147**, 195–197 (1981).
21. Farrar, M. Striped Smith-Waterman speeds database searches six times over other SIMD implementations. *Bioinformatics* **23**, 156–161 (2007).
22. Rognes, T. Faster Smith-Waterman database searches with inter-sequence SIMD parallelisation. *BMC Bioinformatics* **12**, 221 (2011).
23. Li, H. Minimap2: pairwise alignment for nucleotide sequences. *Bioinformatics* **34**, 3094–3100 (2018).
24. Altschul, S. F. et al. Protein database searches using compositionally adjusted substitution matrices. *FEBS J.* **272**, 5101–5109 (2005).
25. Jain, M., Olsen, H. E., Paten, B. & Akeson, M. The Oxford Nanopore MinION: delivery of nanopore sequencing to the genomics community. *Genome Biol.* **17**, 239 (2016).
26. Wootton, J. C. & Federhen, S. Statistics of local complexity in amino acid sequences and sequence databases. *Comput. Chem.* **17**, 149–163 (1993).
27. Frith, M. C. A new repeat-masking method enables specific detection of homologous sequences. *Nucleic Acids Res.* **39**, e23 (2011).
28. Darling, A. E., Carey, L. & Feng, W. C. The design, implementation, and evaluation of mpiBLAST. In *ClusterWorld Conference & Expo and the 4th International Conference on Linux Clusters: The HPC Revolution 2003* https://public.lanl.gov/radiant/pubs/bio/cwce03.pdf (2003).
29. Sokal, R. R. & Michener, C. D. *A Statistical Method for Evaluating Systematic Relationships* (University of Kansas, 1958).
30. Buchfink, B. DIAMOND v2.0.7 benchmark. *figshare* https://doi.org/10.6084/m9.figshare.c.5053112.v1 (2021).
31. Stewart, R. D. et al. Compendium of 4,941 rumen metagenome-assembled genomes for rumen microbiome biology and enzyme discovery. *Nat. Biotechnol.* **37**, 953–961 (2019).
32. Bahram, M. et al. Structure and function of the global topsoil microbiome. *Nature* **560**, 233–237 (2018).

Acknowledgements

We thank D. Weigel for careful reading of the manuscript and for valuable feedback and support, the Max Planck Computing and Data Facility, especially R. Dohmen and M. Rampp, for access to and support of the HPC infrastructure, and A. Tomescu for sharing insights regarding the bit pattern matching algorithm used in the leftmost seed filter. This work was supported by the Max Planck Society.

Author contributions

H.-G.D. and B.B. designed this study, B.B. designed and implemented the algorithms, K.R. implemented the distributed-memory parallelization, B.B. and H.-G.D. analyzed and interpreted the results, and B.B. and H.-G.D. wrote the manuscript with contributions from K.R. All authors have read and approved the final version of the manuscript.

Competing interests

The authors declare no competing interests.

Additional information

Extended data are available for this paper at https://doi.org/10.1038/s41592-021-01101-x.

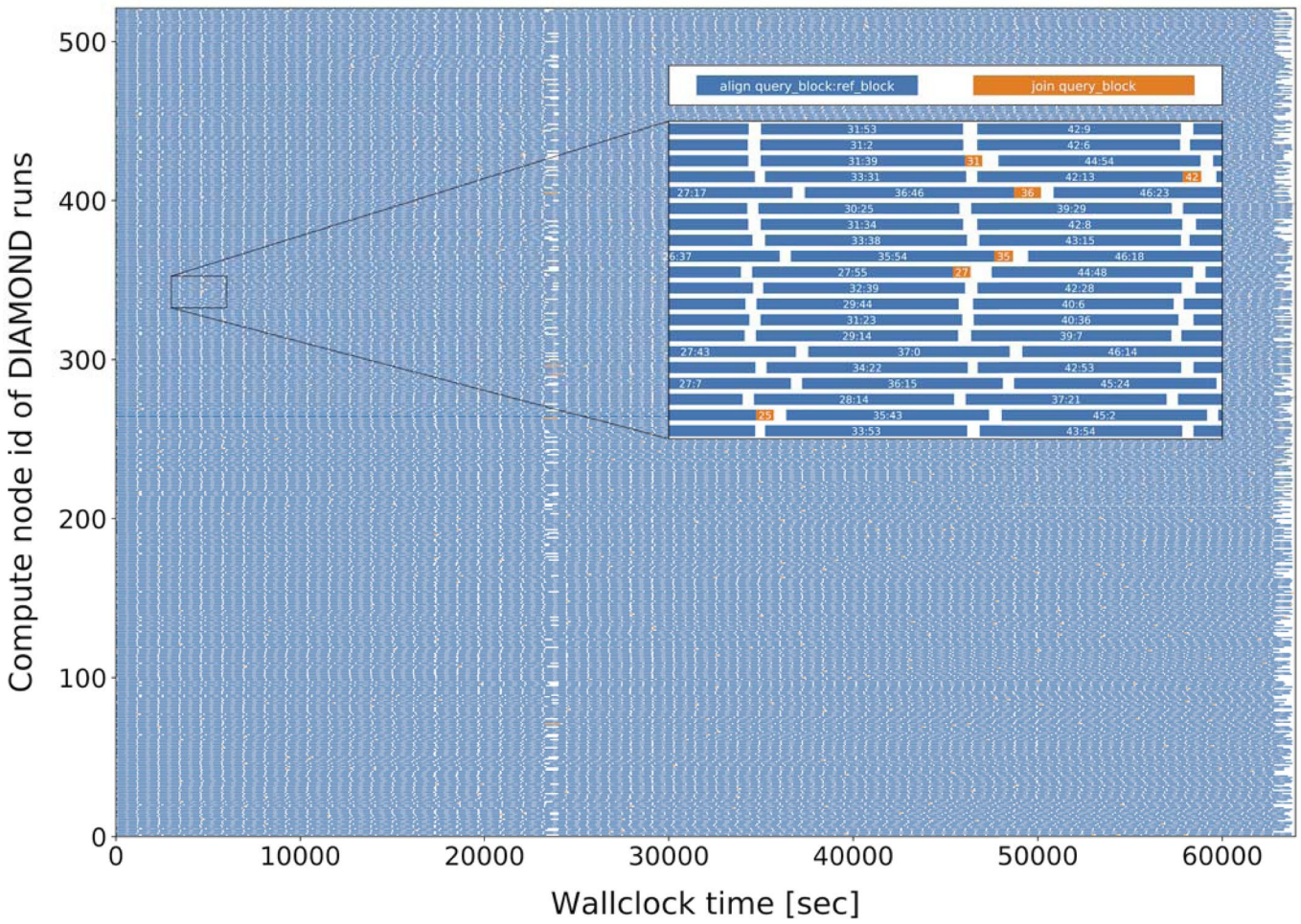
Supplementary information The online version contains supplementary material available at https://doi.org/10.1038/s41592-021-01101-x.

Correspondence and requests for materials should be addressed to H.-G.D.

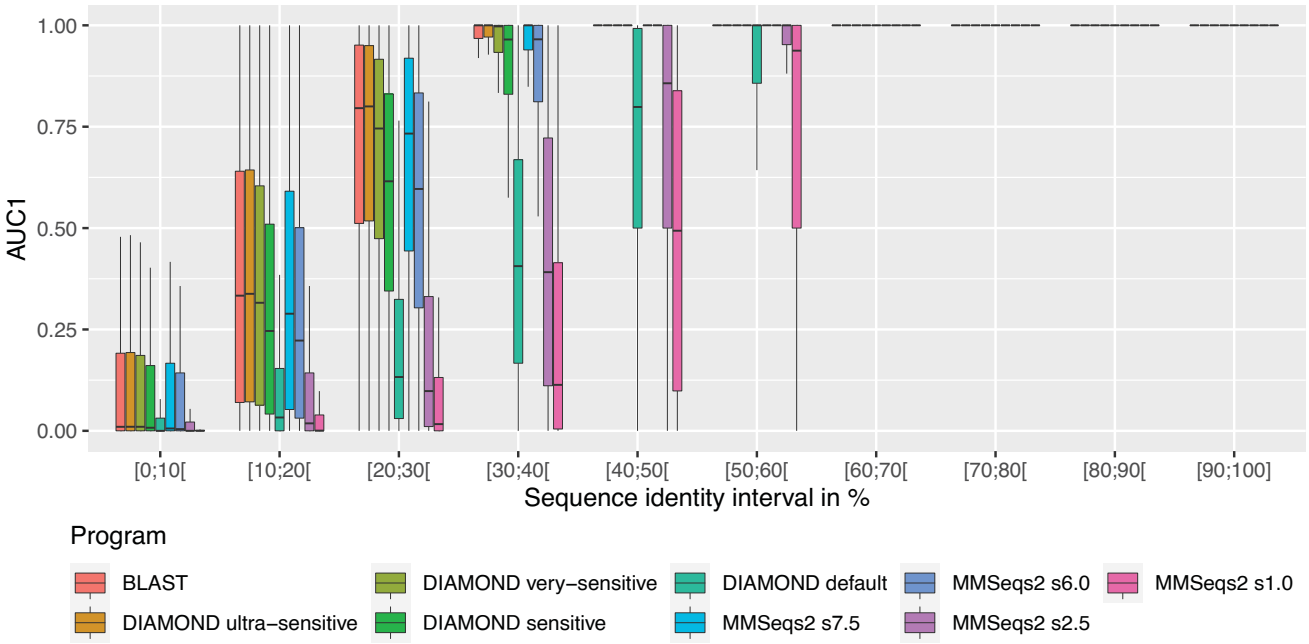
Reprints and permissions information is available at www.nature.com/reprints.

Peer reviewer information *Nature Methods* thanks Weizhong Li, Istvan Albert and Curtis Huttenhower for their contributions to the peer review of this work. Arunima Singh was the primary editor on this article, and managed its editorial process and peer review in collaboration with the rest of the editorial team.

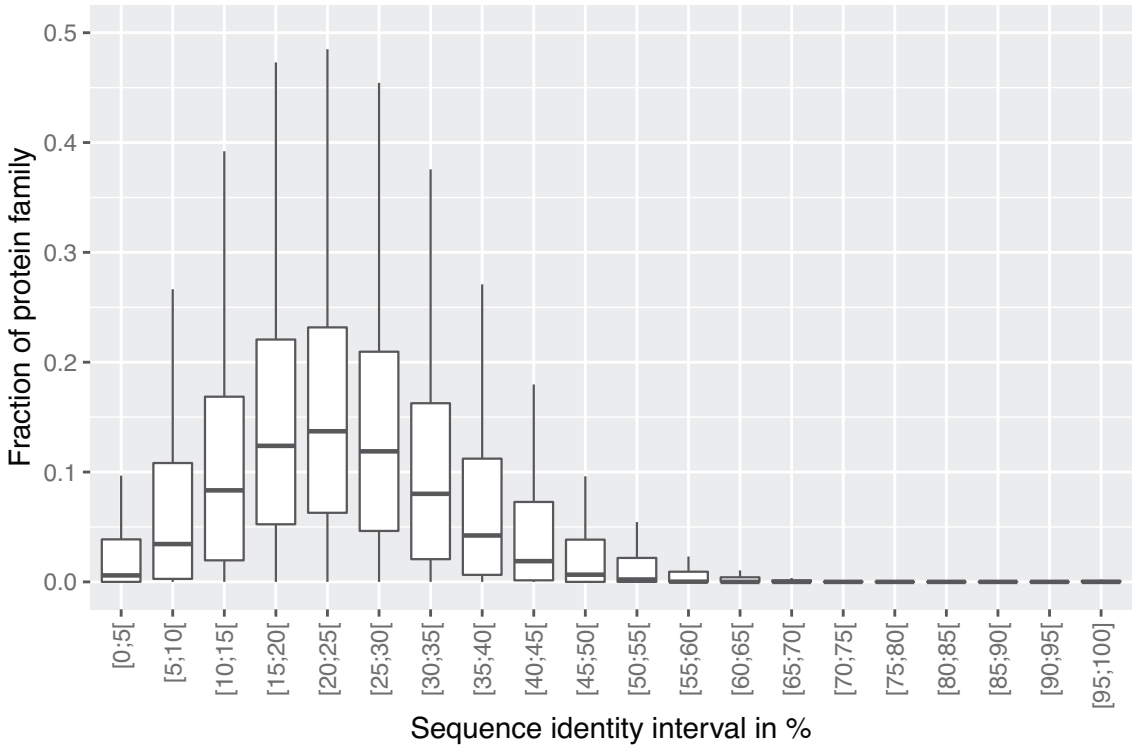
Material from 'Buchfink, B., Reuter, K. & Drost, HG. Sensitive protein alignments at tree-of-life scale using DIAMOND. Nat Methods 18, 366–368 (2021), Springer Nature Limited '



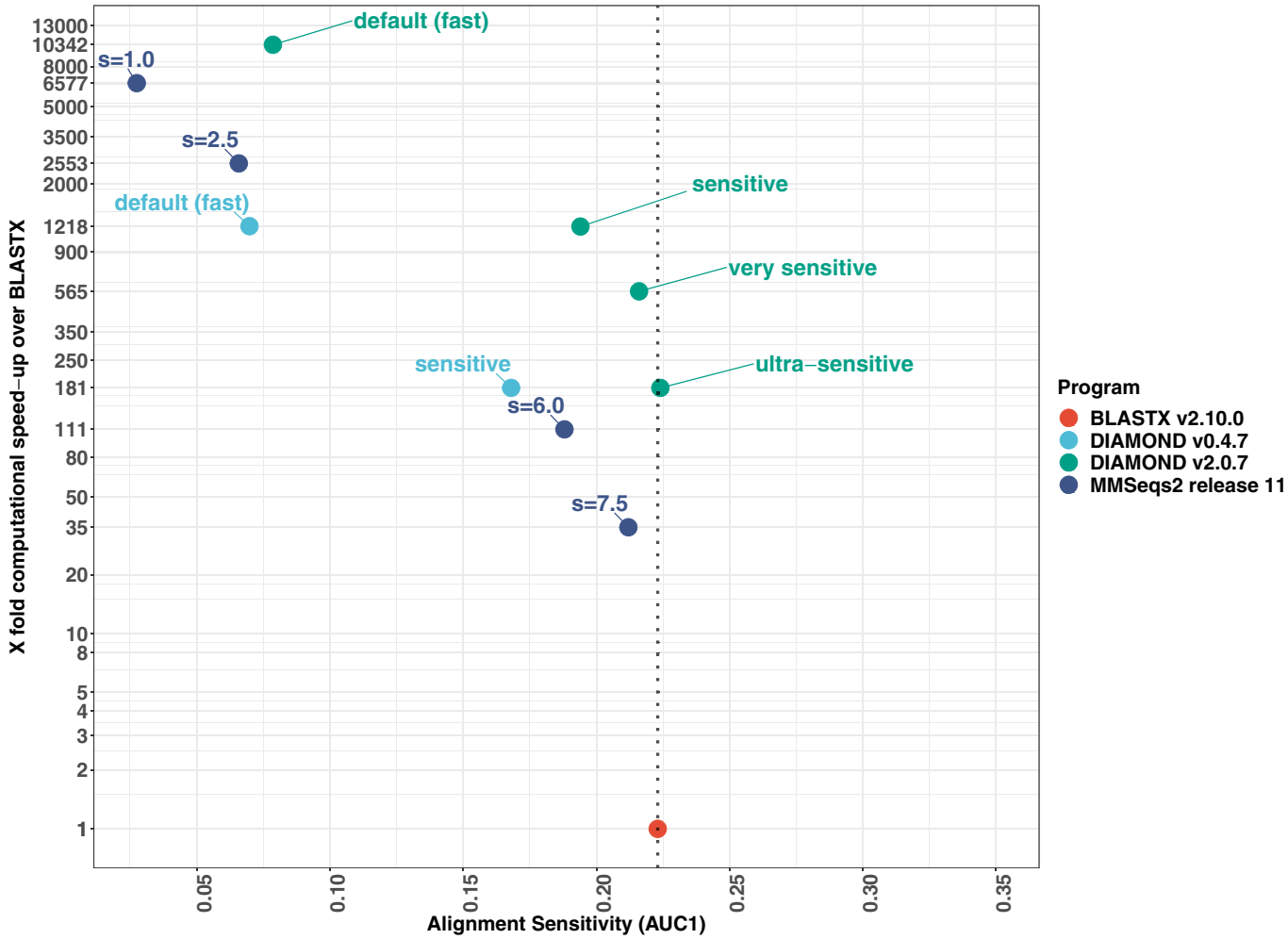
Extended Data Fig. 1 | Detailed visual trace of a massively parallel DIAMOND run on 520 nodes (41,600 threads) of the Cobra supercomputer of the Max Planck Society. To illustrate the scalability of DIAMOND (v2.0.0) in a distributed computing environment (supercomputer), line-by-line worker tasks are shown individually for each worker node (the detailed version of Fig. 2). The sequence of tasks DIAMOND (v2.0.0) has performed over time is indicated by blue rectangles and orange rectangles, in which blue rectangles denote the alignment process and orange rectangles represent join operations. The numbers within the rectangle indicate the indices of the individual query chunks and of their respective reference chunks. White spaces encode the IO activity on the supercomputer's shared parallel file system. The run shown here was performed in–ultra-sensitive mode and used the full NCBI non-redundant database as the query database, and the UniRef50 database as the reference database, finishing in below 18 hours of wallclock time. The result demonstrates DIAMOND's efficient distribution of massively parallel work packages at scale, showing that all workers finish around the same time without creating a significant load imbalance (Experimental Study).



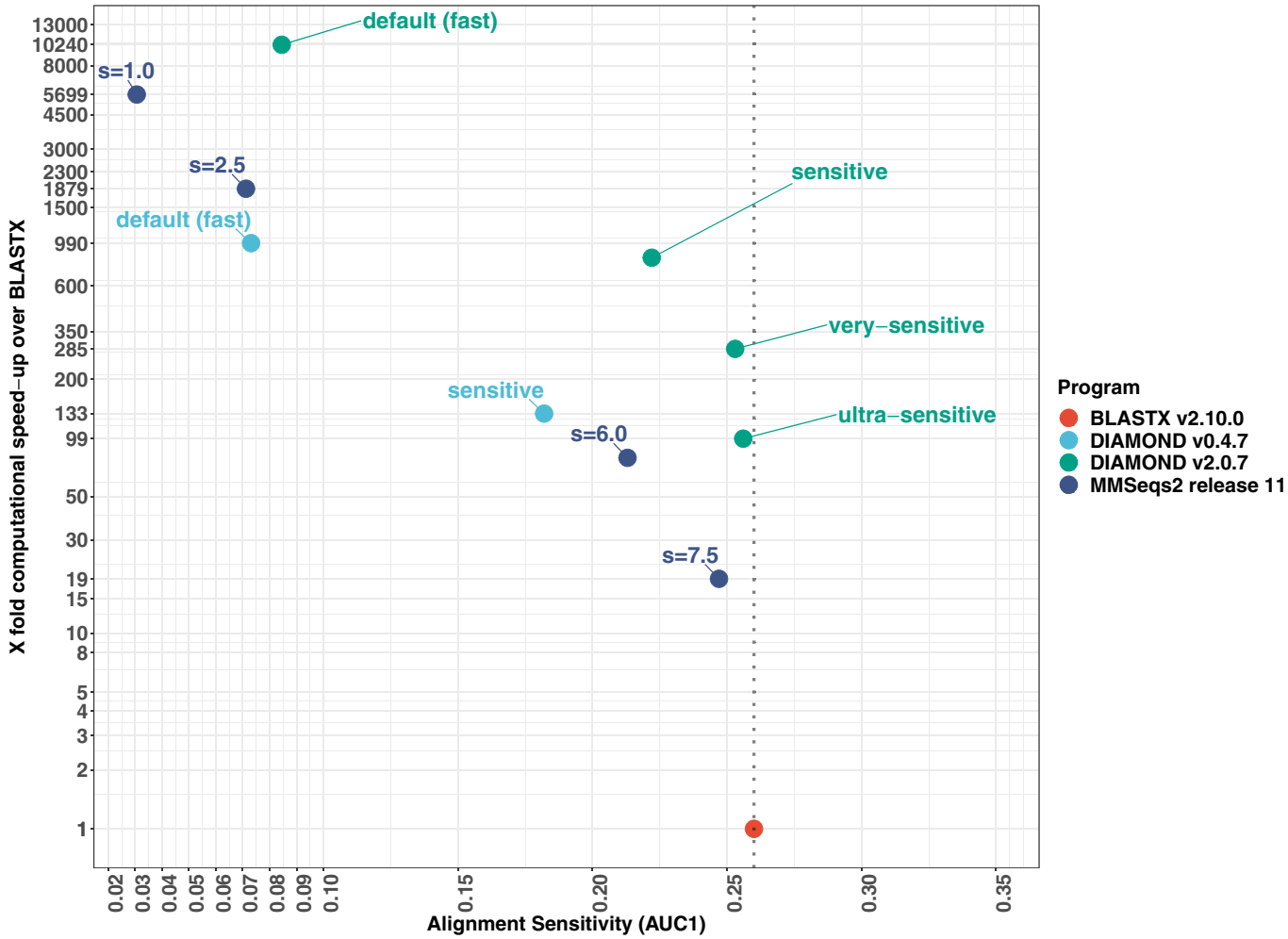
Extended Data Fig. 2 | Assessment of true positive alignments. AUC1 sensitivity as reported for our main benchmark, resolved by sequence identity of the query-subject association under our SCOPe annotation (middle = median, hinges=25%/75% quantiles, lower/upper whisker = smallest/largest observation greater/less than or equal to lower/upper hinge -/+ 1.5 * IQR).



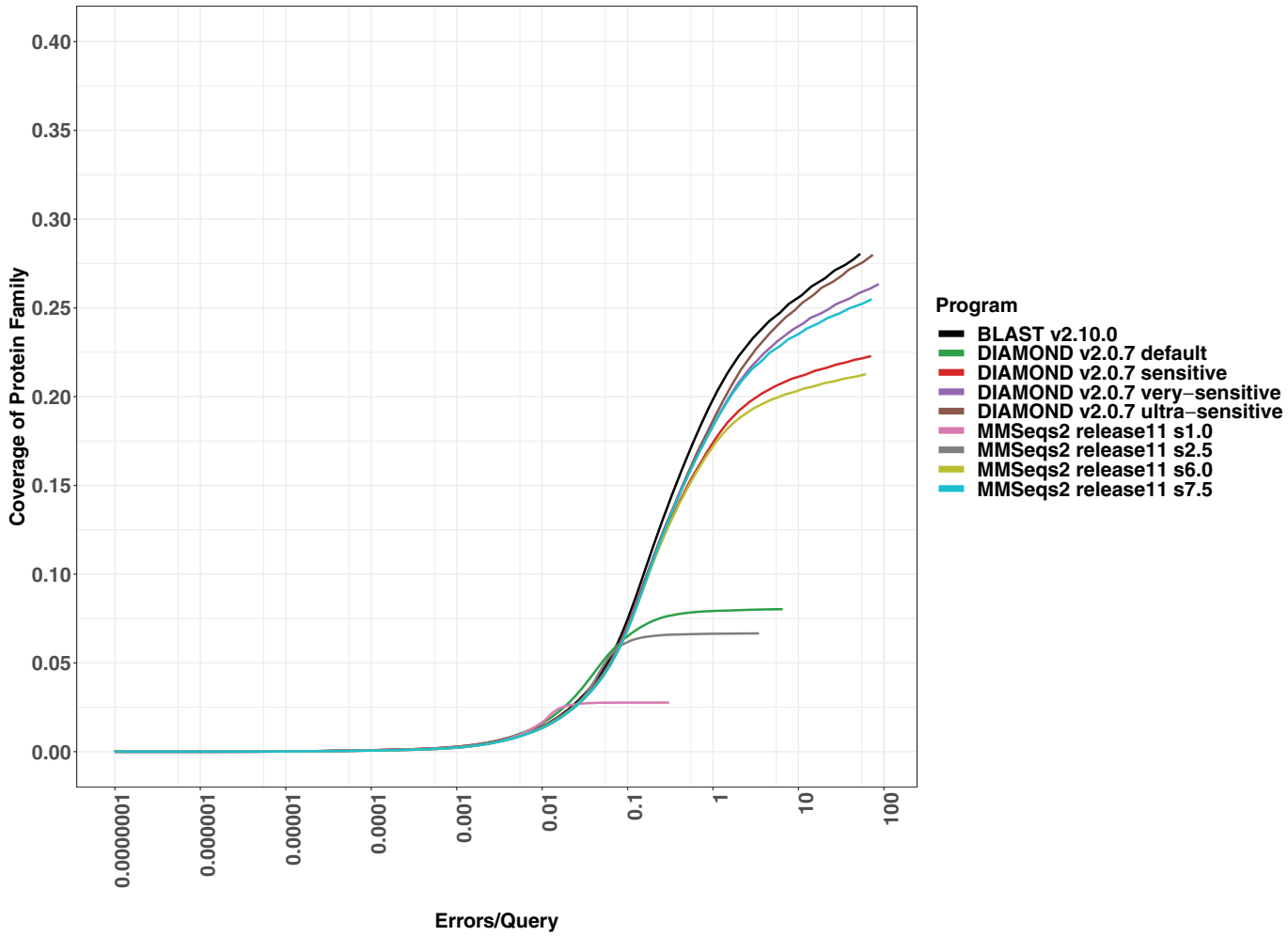
Extended Data Fig. 3 | Assessment of protein family associations. The distribution (shown in 5% bins) of a query's protein family member associations with respect to the sequence identity of the corresponding Needleman Wunsch alignments between the annotated ranges (middle = median, hinges=25%/75% quantiles, lower/upper whisker = smallest/largest observation greater/less than or equal to lower/upper hinge −/+ 1.5 * IQR).



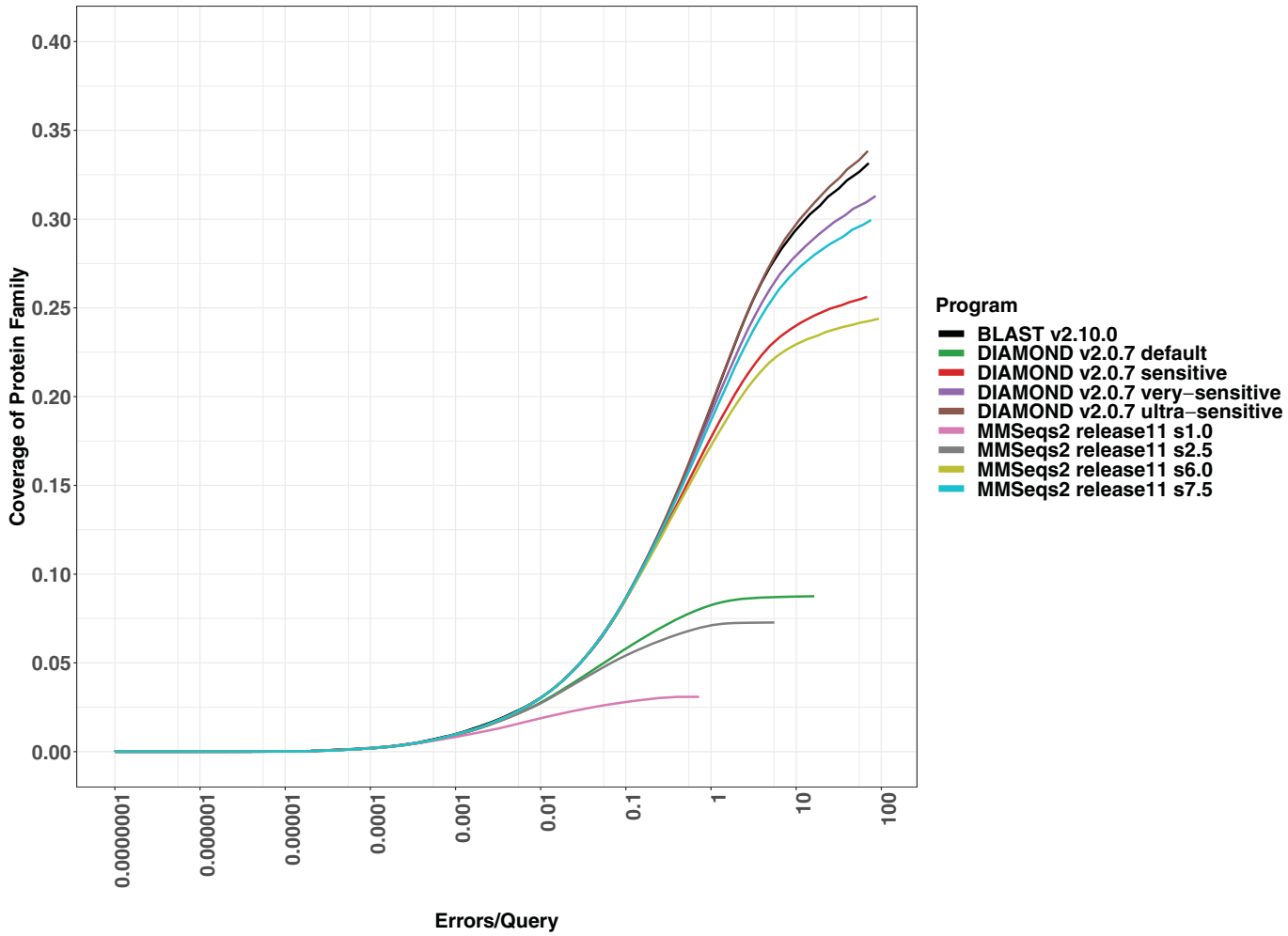
Extended Data Fig. 4 | Metagenomic benchmark results using Illumina HiSeq 4000 paired end sequencing (2x150 bp) reads from Stewart et al., 2019. Computational speedup and alignment sensitivity comparisons for translated searches of 150bp Illumina short reads from rumen metagenome samples (Supplementary Benchmark 1). Alignment sensitivity (AUC1) is measured as the fraction of the query's protein family covered until the first false positive, averaged over all queries in the benchmark dataset. The y-axis denotes the x-fold computational speedup achieved over BLASTX v2.10.0.



Extended Data Fig. 5 | Metagenomic benchmark results using Illumina HiSeq 2500 paired end sequencing (2x250 bp) reads from Bahram et al., 2018. Computational speedup and alignment sensitivity comparisons for translated searches of 250bp Illumina short reads from topsoil metagenome samples (Supplementary Benchmark 2). Alignment sensitivity (AUC1) is measured as the fraction of the query's protein family covered until the first false positive, averaged over all queries in the benchmark dataset. The y-axis denotes the x-fold computational speedup achieved over BLASTX v2.10.0.



Extended Data Fig. 6 | ROC curves for metagenomic benchmark using Illumina HiSeq 4000 paired end sequencing (2x150bp) reads from Stewart et al., 2019. We show the true average error rate per query (x-axis) against the average coverage of the protein family (y-axis) depending on the e-value threshold for Supplementary Benchmark 1.



Extended Data Fig. 7 | ROC curves for metagenomic benchmark using Illumina HiSeq 2500 paired end sequencing (2x250bp) reads from Bahram et al., 2018. We show the true average error rate per query (x-axis) against the average coverage of the protein family (y-axis) depending on the e-value threshold for Supplementary Benchmark 2.

nature research

Corresponding author(s): Dr. Hajk-Georg Drost

Last updated by author(s): Feb 5, 2021

Reporting Summary

Nature Research wishes to improve the reproducibility of the work that we publish. This form provides structure for consistency and transparency in reporting. For further information on Nature Research policies, see our Editorial Policies and the Editorial Policy Checklist.

Statistics

For all statistical analyses, confirm that the following items are present in the figure legend, table legend, main text, or Methods section.

n/a	Confirmed
☐	☒ The exact sample size (*n*) for each experimental group/condition, given as a discrete number and unit of measurement
☐	☒ A statement on whether measurements were taken from distinct samples or whether the same sample was measured repeatedly
☒	☐ The statistical test(s) used AND whether they are one- or two-sided *Only common tests should be described solely by name; describe more complex techniques in the Methods section.*
☒	☐ A description of all covariates tested
☒	☐ A description of any assumptions or corrections, such as tests of normality and adjustment for multiple comparisons
☒	☐ A full description of the statistical parameters including central tendency (e.g. means) or other basic estimates (e.g. regression coefficient) AND variation (e.g. standard deviation) or associated estimates of uncertainty (e.g. confidence intervals)
☒	☐ For null hypothesis testing, the test statistic (e.g. *F*, *t*, *r*) with confidence intervals, effect sizes, degrees of freedom and *P* value noted *Give P values as exact values whenever suitable.*
☒	☐ For Bayesian analysis, information on the choice of priors and Markov chain Monte Carlo settings
☒	☐ For hierarchical and complex designs, identification of the appropriate level for tests and full reporting of outcomes
☒	☐ Estimates of effect sizes (e.g. Cohen's *d*, Pearson's *r*), indicating how they were calculated

Our web collection on statistics for biologists contains articles on many of the points above.

Software and code

Policy information about availability of computer code

Data collection	The open source softwares DIAMOND (v2.0.0, https://github.com/bbuchfink/diamond), SWIPE (v2.1.0), MMSeqs2 (e9678f625b16a806e1ae0bf04d7daf733f1142f2, https://github.com/soedinglab/MMseqs2/) were used for creation of the query dataset and the SCOP annotation. SpEED (v1.0) was used for computing spaced seeds.
Data analysis	The open source softwares DIAMOND (v2.0.0, v2.0.5, v2.0.7, v0.7.12, v0.4.7, https://github.com/bbuchfink/diamond), MMSeqs2 (e9678f625b16a806e1ae0bf04d7daf733f1142f2, https://github.com/soedinglab/MMseqs2/), BLAST (v2.10.0, v2.2.31), QuickBLAST (v0.0.0) were used for generating and evaluating the benchmark data. EMBOSS needleall v6.6.0.0 was used to compute Needleman Wunsch alignments.

For manuscripts utilizing custom algorithms or software that are central to the research but not yet described in published literature, software must be made available to editors and reviewers. We strongly encourage code deposition in a community repository (e.g. GitHub). See the Nature Research guidelines for submitting code & software for further information.

Data

Policy information about availability of data

All manuscripts must include a data availability statement. This statement should provide the following information, where applicable:

- Accession codes, unique identifiers, or web links for publicly available datasets
- A list of figures that have associated raw data
- A description of any restrictions on data availability

The sequence and annotation data that support the findings of this study are available in figshare with the identifier doi:10.6084/m9.figshare.c.5053112.v1. The SCOPe ASTRAL40 dataset can be downloaded at http://scop.berkeley.edu/downloads/scopeseq-2.07/astral-scopedom-seqres-gd-sel-gs-bib-40-2.07.fa. The download URLs for the two databases UniRef50 and NCBI NR are ftp://ftp.uniprot.org/pub/databases/uniprot/uniref/uniref50/uniref50.fasta.gz and ftp://

ftp.ncbi.nlm.nih.gov/blast/db/FASTA/nr.gz. The sequencing reads of the supplementary benchmarks are part of the samples with ENA accessions SAMEA5383815, SAMEA5383897, SAMEA5383886, SAMEA5383828, SAMEA5383925, SAMEA5383848, SAMEA5383824, SAMEA5383873, SAMEA5384011, SAMEA5383807, SAMEA103892455, SAMEA103892562, SAMEA103892552, SAMEA103892441, SAMEA103892588, SAMEA103892582, SAMEA103892581, SAMEA103892571, SAMEA103892491, SAMEA103892619.

Field-specific reporting

Please select the one below that is the best fit for your research. If you are not sure, read the appropriate sections before making your selection.

☒ Life sciences ☐ Behavioural & social sciences ☐ Ecological, evolutionary & environmental sciences

For a reference copy of the document with all sections, see nature.com/documents/nr-reporting-summary-flat.pdf

Life sciences study design

All studies must disclose on these points even when the disclosure is negative.

Sample size	Datasets of less than 100,000 proteins have been considered sufficient to assess the sensitivity of protein aligners in reference literature, for example: Altschul et al., Nucleic Acids Research, 1997, Vol. 25, No. 17 3389–3402 / Altschul et al., FEBS Journal 272 (2005) 5101–5109 / Brenner et al., Proc. Natl. Acad. Sci. USA Vol. 95, pp. 6073–6078, May 1998 / Steinegger et al., Nature Biotechnology 35, 1026–1028 (2017). In comparison, our query and target datasets comprise 1.7 million and 7.7 million proteins respectively.
Data exclusions	No data was excluded.
Replication	The benchmarking computations were repeated independently at least 3 times which successfully verified the consistency of the results.
Randomization	Randomization does not apply because the study does not involve subjects that are assigned to treatment groups.
Blinding	Blinding does not apply because the study does not involve subjects that are assigned to treatment groups.

Reporting for specific materials, systems and methods

We require information from authors about some types of materials, experimental systems and methods used in many studies. Here, indicate whether each material, system or method listed is relevant to your study. If you are not sure if a list item applies to your research, read the appropriate section before selecting a response.

Materials & experimental systems

n/a	Involved in the study
☒	☐ Antibodies
☒	☐ Eukaryotic cell lines
☒	☐ Palaeontology and archaeology
☒	☐ Animals and other organisms
☒	☐ Human research participants
☒	☐ Clinical data
☒	☐ Dual use research of concern

Methods

n/a	Involved in the study
☒	☐ ChIP-seq
☒	☐ Flow cytometry
☒	☐ MRI-based neuroimaging

Article

Petabase-scale sequence alignment catalyses viral discovery

Material from 'Edgar, R.C., Taylor, B., Lin, V. et al. Petabase-scale sequence alignment catalyses viral discovery. Nature 602, 142–147 (2022), Springer Nature Limited'

https://doi.org/10.1038/s41586-021-04332-2

Received: 10 August 2020

Accepted: 10 December 2021

Published online: 26 January 2022

Check for updates

Robert C. Edgar[1,16], Jeff Taylor[2,16], Victor Lin[3,16], Tomer Altman[4,16], Pierre Barbera[5,16], Dmitry Meleshko[6,7,16], Dan Lohr[8,16], Gherman Novakovsky[9,16], Benjamin Buchfink[10,16], Basem Al-Shayeb[11,16], Jillian F. Banfield[12,16], Marcos de la Peña[13,16], Anton Korobeynikov[6,14,16], Rayan Chikhi[15,16] & Artem Babaian[2,16] ✉

Public databases contain a planetary collection of nucleic acid sequences, but their systematic exploration has been inhibited by a lack of efficient methods for searching this corpus, which (at the time of writing) exceeds 20 petabases and is growing exponentially[1]. Here we developed a cloud computing infrastructure, Serratus, to enable ultra-high-throughput sequence alignment at the petabase scale. We searched 5.7 million biologically diverse samples (10.2 petabases) for the hallmark gene RNA-dependent RNA polymerase and identified well over 10^5 novel RNA viruses, thereby expanding the number of known species by roughly an order of magnitude. We characterized novel viruses related to coronaviruses, hepatitis delta virus and huge phages, respectively, and analysed their environmental reservoirs. To catalyse the ongoing revolution of viral discovery, we established a free and comprehensive database of these data and tools. Expanding the known sequence diversity of viruses can reveal the evolutionary origins of emerging pathogens and improve pathogen surveillance for the anticipation and mitigation of future pandemics.

Viral zoonotic disease has had a major impact on human health over the past century, with notable examples including the 1918 Spanish influenza, AIDS, SARS, Ebola and COVID-19. There are an estimated 3×10^5 mammalian virus species from which infectious diseases in humans may arise[2], of which only a fraction are known at present. Global surveillance of virus diversity is required for improved prediction and prevention of future epidemics, and is the focus of international consortia and hundreds of research laboratories[3,4].

Pioneering works expanding the virome of the Earth have each uncovered thousands of novel viruses, with the rate of virus discovery increasing exponentially and driven largely by the increased availability of high-throughput sequencing[5–11]. Sequence analysis remains computationally expensive, in particular the assembly of short reads into contigs, which limits the breadth of samples analysed. Here we propose an alternative alignment-based strategy that is considerably cheaper than assembly and enables processing of massive datasets.

Petabases (1×10^{15} bases) of sequencing data are freely available in public databases such as the Sequence Read Archive (SRA)[1], in which viral nucleic acids are often captured incidental to the goals of the original studies[12]. To catalyse global virus discovery, we developed the Serratus cloud computing infrastructure for ultra-high-throughput sequence alignment, screening 5.7 million ecologically diverse sequencing libraries or 10.2 petabases of data.

Identification of Earth's virome is a fundamental step in preparing for the next pandemic. We lay the foundations for future research by enabling direct access to 883,502 RNA-dependent RNA polymerase (RdRP)-containing sequences, which include the RdRP from 131,957 novel RNA viruses (sequences with greater than 10% divergence from a known RdRP), including 9 novel coronaviruses. Altogether this captures the collective efforts of over a decade of sequencing studies in a free repository, available at https://serratus.io.

Accessing the planetary virome

Serratus is a free, open-source cloud-computing infrastructure optimized for petabase-scale sequence alignment against a set of query sequences. Using Serratus, we aligned more than one million short-read sequencing datasets per day for less than 1 US cent per dataset (Extended Data Fig. 1). We used a widely available commercial computing service to deploy up to 22,250 virtual CPUs simultaneously (see Methods), leveraging SRA data mirrored onto cloud platforms as part of the NIH STRIDES initiative[13].

Our search space spans data deposited over 13 years from every continent and ocean, and all kingdoms of life (Fig. 1). We applied Serratus

[1]Independent researcher, Corte Madera, CA, USA. [2]Independent researcher, Vancouver, British Columbia, Canada. [3]Independent researcher, Seattle, WA, USA. [4]Altman Analytics, San Francisco, CA, USA. [5]Computational Molecular Evolution Group, Heidelberg Institute for Theoretical Studies, Heidelberg, Germany. [6]Center for Algorithmic Biotechnology, St Petersburg State University, St Petersburg, Russia. [7]Tri-Institutional PhD Program in Computational Biology and Medicine, Weill Cornell Medical College, New York, NY, USA. [8]Unaffiliated, Atlanta, GA, USA. [9]Bioinformatics Graduate Program, University of British Columbia, Vancouver, British Columbia, Canada. [10]Computational Biology Group, Max Planck Institute for Biology, Tübingen, Germany. [11]Department of Plant and Microbial Biology, University of California, Berkeley, Berkeley, CA, USA. [12]Department of Earth and Planetary Science, University of California, Berkeley, Berkeley, CA, USA. [13]Instituto de Biología Molecular y Celular de Plantas, Universidad Politécnica de Valencia–CSIC, Valencia, Spain. [14]Department of Statistical Modelling, St Petersburg State University, St Petersburg, Russia. [15]G5 Sequence Bioinformatics, Department of Computational Biology, Institut Pasteur, Paris, France. [16]These authors contributed equally: Robert C. Edgar, Jeff Taylor, Victor Lin, Tomer Altman, Pierre Barbera, Dmitry Meleshko, Dan Lohr, Gherman Novakovsky, Benjamin Buchfink, Basem Al-Shayeb, Jillian F. Banfield, Marcos de la Peña, Anton Korobeynikov, Rayan Chikhi, Artem Babaian. ✉e-mail:

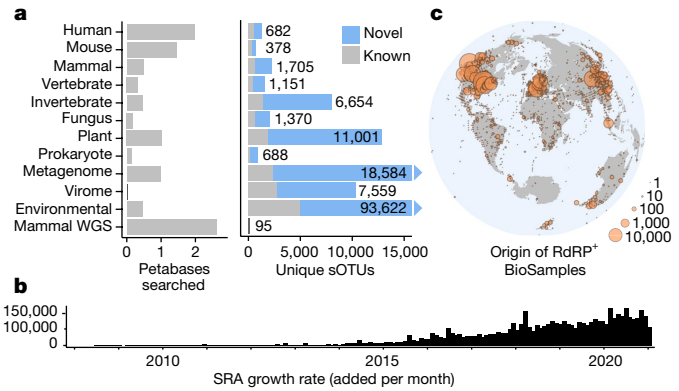


Fig. 1 | Searching the planetary virome. a, Total bases searched from the 5,686,715 SRA sequencing runs analysed in the viral RdRP search grouped by sample taxonomy, where available (see Extended Data Figs. 1, 3, Supplementary Table 1). A total of 8,871 out of 15,016 (59%) of known RdRP sOTUs were observed in the SRA, and 131,957 unique and novel RdRP sOTUs were identified (see Extended Data Fig. 2). sOTUs identified in multiple taxonomic groups are counted in each group separately; numbers shown indicate the number of novel sOTUs in each group. WGS, whole-genome sequencing. **b**, Release dates of the runs included in the analysis reflecting the growth rate of available data. **c**, Sample locations for 635,656 RdRP-containing contigs (27.8% of samples lacked geographical metadata). The high density of RdRP seen in North America, western Europe and eastern Asia reflects the substantial acquisition bias for samples originating from these regions. Interactive RdRP map is available at https://serratus.io/geo.

in two of many possible configurations. First, to identify libraries that contain known or closely related viruses, we searched 3,837,755 (around May 2020) public RNA sequencing (RNA-seq), meta-genome, meta-transcriptome and meta-virome datasets (termed sequencing runs[1]) against a nucleotide pangenome of all coronavirus sequences and RefSeq vertebrate viruses. We then aligned 5,686,715 runs (January 2021) against all known viral RdRP amino acid sequences using a specially optimized version of DIAMOND v2 (ref. [14], Methods); this search was completed within 11 days, at a cost of US$23,980 (Fig. 1a, Methods).

Previous approaches for identifying sequences across the entire SRA rely on pre-computed indexes[15,16] that require exact substring or hash-based matches, which limits their sensitivity to diverged sequences (Extended Data Fig. 1f). Pre-assembled reads (for example, the NCBI Transcriptome Shotgun Assembly database) enable efficient alignment-based searches[5], but are at present available for only a small fraction of the SRA. Serratus aligns a query of up to hundreds of megabytes against unassembled libraries, achieving greater sensitivity to diverged viruses compared to substring (*k*-mer) indexes while using far fewer computational resources than de novo assembly (Fig. 1g, Methods).

A sketch of RdRP

Viral RdRP is a hallmark gene of RNA viruses that lack a DNA stage of replication[17]. We identified RdRP by a well-conserved amino acid sub-sequence that we call the 'palmprint'. Palmprints are delineated by three essential motifs that together form the catalytic core in the RdRP structure[18] (Fig. 2). We constructed species-like operational taxonomic units (sOTUs) by clustering palmprints at a threshold of 90% amino acid identity, chosen to approximate taxonomic species[18].

A total of 3,376,880 (59.38%) sequencing runs contained one or more reads that mapped to the RdRP query (*E*-value $\leq 1 \times 10^{-4}$). We assembled RdRP aligned reads from each library (and their mate-pairs when available), which yielded 4,261,616 'microassembly' contigs. Of these, 881,167 (20.7%) contained a high-confidence palmprint identified by Palmscan (false discovery rate = 0.001)[18], representing 260,808 unique palmprints. Applying Palmscan to reference databases[1,7,19], we obtained 45,824 unique palmprints, which clustered into 15,016 known sOTUs. If a newly acquired palmprint aligned to a known palmprint at an identity of 90% or greater, it was assigned membership to that reference sOTU; otherwise, it was designated as novel. We clustered novel palmprints at 90% identity and obtained 131,957 novel sOTUs, representing an increase in the number of known RNA viruses by a factor of 9.8. Clustering novel palmprints at genus-like 75% and family-like 40% thresholds yielded 78,485 and 3,599 novel OTUs, which represent increases of 8.0× and 1.9×, respectively (Fig. 2b).

We extracted host, geospatial and temporal metadata for each biological sample when available (Fig. 1c), noting that the majority (88%) of novel RdRP sOTUs were observed from metagenomic or environmental runs in which accurate host inference is challenging. Mapping observations of virus marker genes across time and space suggests ecological niches for these viruses, and improved characterization of sequence diversity can improve PCR primer design for in situ virus identification.

We estimate that around 1% of sOTUs are endogenous virus elements (EVEs); that is, viral RdRPs that have reverse-transcribed into a host germline. We did not attempt to systematically distinguish EVEs from viral RdRPs, noting that EVEs with intact catalytic motifs are likely to be recent insertions that can serve as a representative sequence for related exogenous viruses. Most (60.5%) recovered palmprints were found in exactly one run (singletons), and are observed within the expected frequency range predicted by extrapolating from more abundant sequences (Fig. 2b).

The abundance distribution of distinct palmprints is consistent with log-log-linear for each year from 2015 to 2020 (Extended Data Fig. 2e), and over time, singletons are confirmed by subsequent runs at an approximately constant rate (Extended Data Fig. 2g). The majority of novel viruses will be singletons until the diversity represented by the search query and the fraction of the planetary virome sampled in the SRA both approach saturation. Extrapolating one year forward, by when the SRA is expected to have doubled in size, we predict that 430,000 (95% confidence interval [330,000, 561,000]) additional unique palmprints could be identified by running Serratus with its current query (Fig. 2b).

RNA viruses have highly divergent sequences, even within the conserved RdRP[17]. Amino acid sequence alignment can recover the majority of RdRP short reads above 60% identity, but sensitivity falls as sequences diverge further (Extended Data Fig. 2f). Subsequent microassembly fragmentation can in part account for the decreased abundance of novel sOTUs below 60% identity (Fig. 2b); thus, the sensitivity to highly diverged (less than 50% identity) RdRP sequences is limited in the present study. Saturation of virus discovery within the SRA is far from complete, even if data-growth rates are ignored. Intensive searches for so-called highly diverged or 'dark' viruses[20], in combination with iterative reanalysis (conceptually similar to PSI-BLAST[21]), are likely to yield further expansion of the known virome.

The total number of virus species is estimated to be 10^8 to 10^{12} (ref. [22]), so our data captured at most 0.1% of the global virome. However, if exponential data growth combined with increased search sensitivity continues, we are at the cusp of identifying a notable fraction of Earth's total genetic diversity with tools such as Serratus.

Expanding known *Coronaviridae*

The SARS-CoV-2 pandemic has severely affected human society. We further exemplify the potential of Serratus for virus discovery with the *Coronaviridae* (CoV) family, including a recently proposed subfamily[23] that contains a CoV-like virus, Microhyla alphaletovirus 1 (MLeV), in the frog *Microhyla fissipes*, and Pacific salmon nidovirus (PsNV) described in the endangered *Oncorhynchus tshawytscha*[24].

First, we identified 52,772 runs that contain 10 or more CoV-aligned reads or 2 or more CoV *k*-mers (32-mer,[16]). These runs were de-novo-assembled with a new version of synteny-informed SPAdes

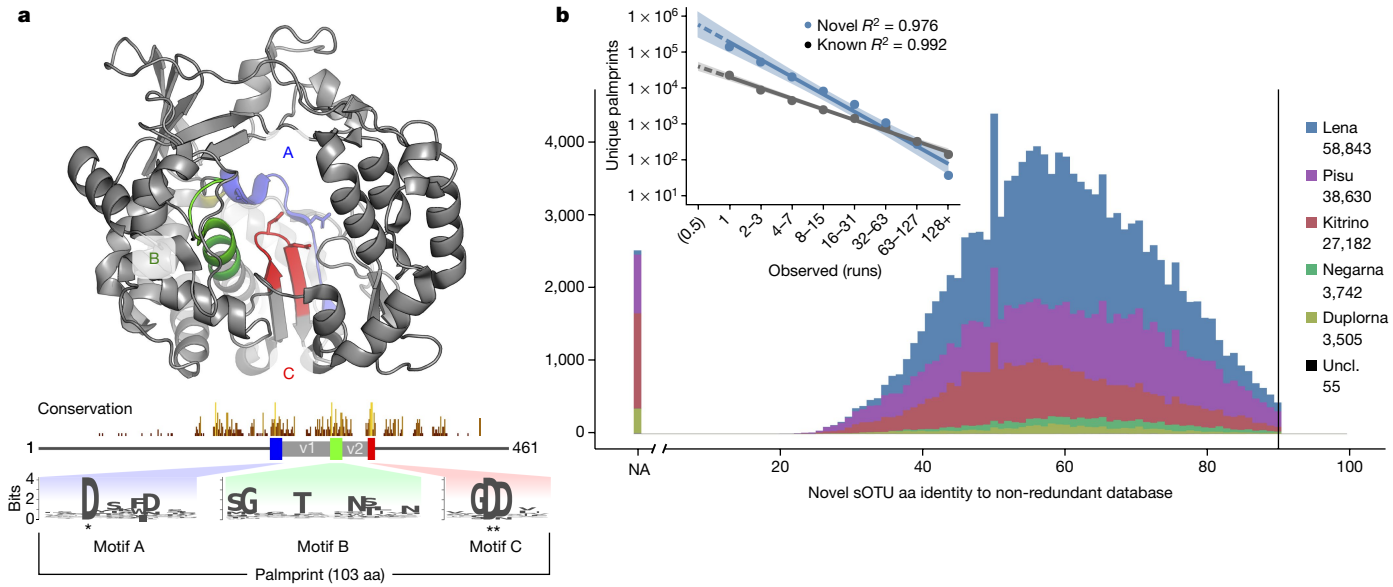


Fig. 2 | RNA-dependent RNA polymerase in the SRA. a, The RdRP palmprint is the protein sequence spanning three well-conserved sequence motifs (A, B and C), including intervening variable regions, exemplified within the full-length poliovirus RdRP structure with essential aspartic acid residues (asterisks) (Protein Data Bank code: 1RA6[49]). Conservation was calculated from RdRP alignment in a previous study[19], trimmed to the poliovirus sequence; motif sequence logos are shown below. aa, amino acids. **b**, Per-phylum histogram of amino acid identity of novel sOTUs aligned to the NCBI non-redundant protein database. Extended Data Figure 3c shows the per-order distribution. Inset, Preston plot and linear regression of palmprint abundances indicates that singleton palmprints (that is, observed in exactly one run) occur within 95% confidence intervals of the value predicted by extrapolation from high-abundance palmprints (linear regression applied to log-transformed data), and this distribution is consistent through time (Extended Data Fig. 2). NA, not applicable; uncl, unclassified.

called coronaSPAdes[25]. This yielded 11,120 identifiable CoV contigs that we annotated for a comprehensive assemblage of *Coronaviridae* in the SRA (see Methods for discussion). With these training data we defined a scoring function to predict the subsequent success of assembly (Extended Data Fig. 3b).

CoV and neighbouring palmprints comprise 70 sOTUs, 44 of which are described in public databases. Seventeen CoV sOTUs contained partial RdRP (inclusive of full palmprint) from an amplicon-based virus discovery study for which the data had not been publicly deposited at the time of writing[26]. The remaining nine sOTUs are novel viruses, with protein domains consistent with a CoV or CoV-like genome organization (Extended Data Fig. 4).

We operationally designate MLeV, PsNV and the nine novel viruses broadly as group E, noting that all were found in samples from non-mammalian aquatic vertebrates (Fig. 3). Notably, *Ambystoma mexicanum* (axolotl) nidovirus (AmexNV) was assembled in 18 runs, 11 of which yielded common contigs of approximately 19 kb. Easing the criteria of requiring an RdRP match in a contig, 28 out of 44 (63.6%) of the runs from the associated studies were AmexNV-positive[27,28]. Consistent assembly break points in AmexNV, PsNV and similar viruses suggest that the viral genomes of this clade of CoV-like viruses are organized in at least two segments, one containing ORF1ab with RdRP, and a shorter segment containing a lamin-associated domain protein, spike and N′ accessory genes (Fig. 3). An assembly gap with common break points is present in the published PsNV genome[24]. Together, these seven monophyletic species possibly represent a distinct clade of segmented CoV-like nidoviruses, although molecular validation of this hypothesis is required.

While our manuscript was under review, public transcriptome screening by Miller et al.[29] identified three group-E CoV sequences that are not included in our sOTU analysis. One CoV$^+$ library had failed at the alignment step, and microassembly from two others yielded incomplete palmprint sub-sequences and therefore lacked the required specificity for the systematic palmprint classification. A high-sensitivity reanalysis of microassemblies for any group-E RdRP sequence fragment captured the two CoV sequences that we missed from the Miller et al. study[29], and found another approximately 25 putative-novel CoV species from 53 fragmented contigs (Supplementary Table 1e).

In addition to identifying genetic diversity within CoV, we cross-referenced CoV$^+$ library metadata to identify possible zoonoses and vectors of transmission. Discordant libraries—ones in which a CoV is identified and the viral expected host[30] does not match the sequencing library source taxa—were rare, accounting for only 0.92% of cases (Supplementary Table 1f).

An important limitation for these analyses is that the nucleic acid reads do not prove that viral infection has occurred in the nominal host species. For example, we identified five libraries in which a porcine, avian, or bat coronavirus was found in plant samples. The parsimonious explanation is that CoV was present in faeces or fertilizer originating from a mammalian or avian host applied to these plants. However, this exemplifies a merit of exhaustive search in identifying transmission vectors and for monitoring the geotemporal distribution of viruses.

Rapid expansion into the viral unknowns

The global mortality from viral hepatitis exceeds that of HIV/AIDS, tuberculosis or malaria[31]. Hepatitis delta virus (HDV) has a small circular RNA genome (around 1,700 nucleotides (nt)) that folds into a rod-like shape and encodes three genes: a delta antigen protein, and two self-cleaving delta ribozymes (drbz)[32].

Before 2018, HDV was the sole known member of its genus; 13 drbz-containing members have since been characterized[33–38], and recently a second class of ribozyme (known as hammerhead or hhrbz) characteristic of plant viroids was identified in delta-like viruses that we refer to as epsilon viruses[39]. By sequence search for the delta antigen protein and ribozymes, we identified 14 delta viruses, 39 epsilon viruses and 311 enigmatic sequences with delta-virus-like synteny that we term zeta viruses (Fig. 4, Extended Data Fig. 5). The evolutionary histories of these mammalian delta viruses are explored further elsewhere[37].

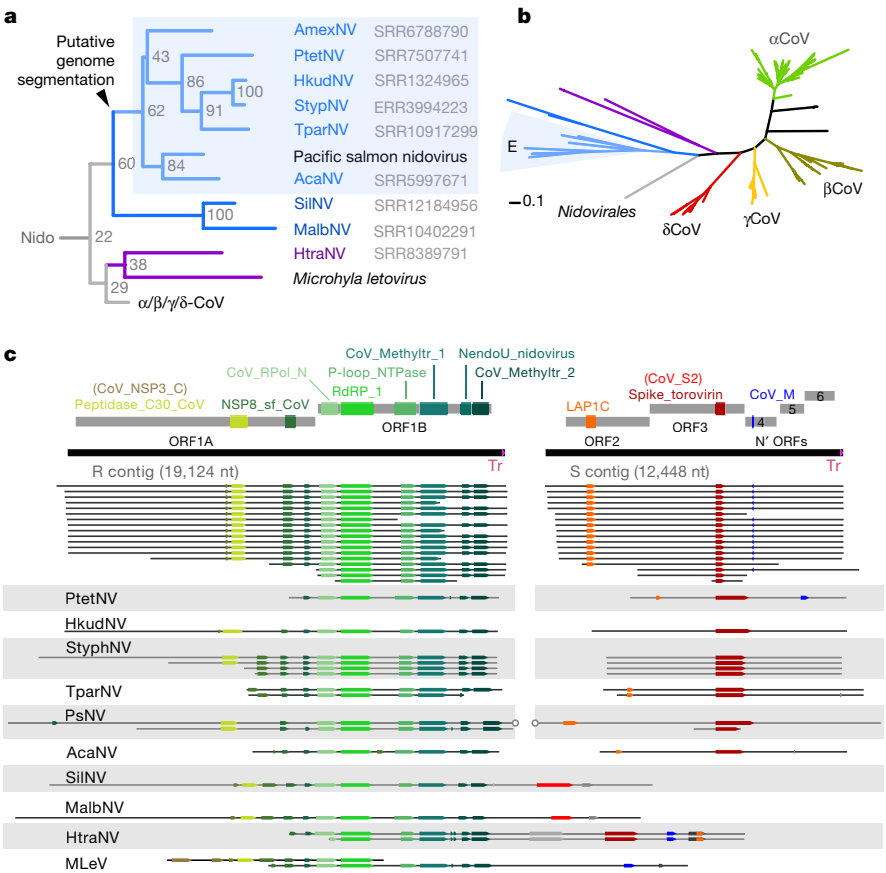


Fig. 3 | Expanding *Coronaviridae*. **a**, Phylogram for group-E sequences. Six viruses were similar to PsNV in *Ambystoma mexicanum* (axolotl; AmexNV), *Puntigrus tetrazona* (tiger barb; PtetNV), *Hippocampus kuda* (seahorse; HkudNV), *Syngnathus typhle* (broad-nosed pipefish; StypNV), *Takifugu pardalis* (fugu fish; TparNV) and the *Acanthemblemaria* sp. (blenny; AcaNV). More-distant members identified were in *Hypomesus transpacificus* (the endangered delta smelt; HtraNV), *Silurus* sp. (catfish; SilNV) and *Monopterus albus* (asian swamp eel; MalbNV). **b**, Unrooted phylogram for *Coronaviridae* annotated with genera (Greek letters) and group-E CoV-like nidoviruses (see also Extended Data Fig. 4). Maximum likelihood tree generated by clustering the RdRP amino acid sequences at 97% identity to show sub-species variability. **c**, Genome structure of AmexNV and the contigs recovered from group-E CoV-like viruses annotated with HMM matches. AmexNV contigs contain an identical 129-nt trailing sequence (Tr). All the putatively segmented CoV-like are monophyletic with PsNV. A gap in the PsNV reference sequence[24] is shown with circles, overlapping the common contig ends seen in these viruses.

The zeta virus circular genomes are highly compressed, ranging from 324 to 789 nt and predicted to fold into rod-like structures. They contain a hhrbz in each orientation and encode two open reading frames (ORFs), one sense and one anti-sense. Both ORFs generally lack stop codons and encompass the entire genome, potentially producing an endless tandem repeat of antigen. The atypical coiled-coil domain of the HDV antigen[40] is conserved in the antigens of new delta and epsilon viruses, whereas epsilon and zeta genomes show analogous hhrbzs (Extended Data Fig. 6), suggesting that these sequences share common ancestry. These abundant elements may help to solve a long-standing question about the origins of circular RNA subviral agents in higher eukaryotes (Extended Data Fig. 6), historically regarded as molecular fossils of a prebiotic RNA world[41].

To evaluate the feasibility of applying Serratus in the context of microbiome research, we sought to locate bacteriophages that are related to recently reported huge phages[42], searching for terminase amino acid sequences. Targeted assembly of 287 high-scoring runs returned 252 terminase-containing contigs of greater than 140 kb. Phylogenetics of these sequences resolved new groups of phages with large genomes (Fig. 4e). Although most phages were from a single animal genus, we identified closely related phages that crossed animal orders, including related phages in a human from Bangladesh (ERR866585) and in groups of cats (PRJEB9357) and dogs (PRJEB34360) from England, sampled five years apart. Similarly, we recovered two approximately 554-kb Lak megaphage genomes (among the largest animal microbiome phages reported so far) that are extremely closely related to sequences previously reported from pigs, baboons and humans[43] (Extended Data Fig. 7). These two genomes were circularized and manually curated to completion. The large carrying capacity of such phages and broad distribution underlines their potential for extensive lateral gene transfer amongst animal microbiomes and modification of host bacterial function. These sequences substantially expand the inventory of phages with genomes whose length range overlaps with those of bacteria.

Discussion

Since the completion of the human genome, the growth of DNA sequencing databases has outpaced Moore's Law. Serratus provides rapid and focused access to genomic sequences captured over more than a decade by the global research community, which would otherwise be inaccessible in practice. This work and further extensions of petabase-scale genomics[15,16,44] are shaping a new era in computational biology, enabling expansive gene discovery, pathogen surveillance and pangenomic evolutionary analyses.

Optimal translation of such massive datasets into meaningful biomedical advances requires free and open collaboration among scientists[45]. The current pandemic underscores the need for prompt,

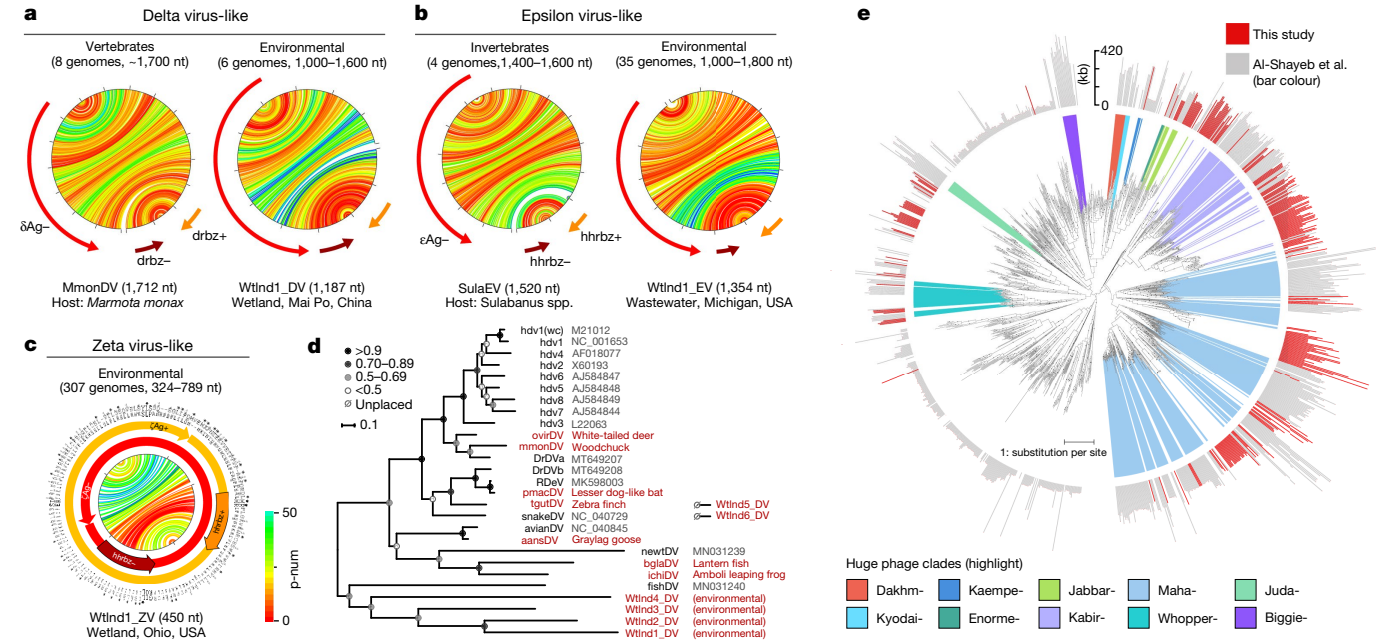


Fig. 4 | Expanding delta viruses and huge phages. a, Genome structure for the *Marmota monax* delta virus (MmonDV) and a delta-virus-like genome detected in an environmental dataset, each containing a negative-sense delta-antigen (δAg) ORF; two delta ribozymes (drbz); and characteristic rod-like folding, where each line shows the predicted base-pairing within the RNA genome, coloured by base-pairing confidence score (p-num)[50]. **b**, Similar genome structure for the Sulabanus spp. epsilon virus-like (SulaEV) and an epsilon-virus-like genome from an environmental dataset, each containing a negative-sense epsilon-antigen (εAg) ORF; two hammerhead ribozymes (hhrbz); and rod-like folding. **c**, Example of the compact genome structure of a Zeta virus-like from an environmental dataset containing two predicted zeta-antigen (ζAg+/–; protein alignment is shown in the outer circles) ORFs without stop codons; two hhrbzs overlapping with the ORFs; and rod-like folding. Further novel genomes are shown in Extended Data Figs. 5, 6. **d**, Maximum-likelihood phylogenetic tree of delta viruses derived from a delta-antigen protein alignment with bootstrap values. Two divergent environmental delta viruses could not yet be placed. **e**, Tree showing huge phage clade expansion. Black dots indicate branches with bootstrap values greater than 90. Outer ring indicates genome or genome fragment length: grey are sequences from Al-Shayeb et al.[42] and reference sequences, shadings indicate previously defined clades of phages with very large genomes (200–735 kb). The Kabirphages (light purple) are shown in expanded view in Extended Data Fig. 7.

unrestricted and transparent data sharing. With these goals in mind, we deposited 7.3 terabytes of virus alignments and assemblies into an open-access database that can be explored via a graphical web interface at https://serratus.io or programmatically through the Tantalus R package and its PostgreSQL interface.

The 'metagenomics revolution' of virus discovery is accelerating[7,11]. Innovative fields such as high-throughput viromics[46] can leverage vast collections of virus sequences to inform policies that predict and mitigate emerging pandemics[47]. Combining ecoinformatics with virus, host and geotemporal metadata offers a proof-of-concept for a global pathogen surveillance network, arising as a by-product of centralized and open data sharing.

Human population growth and encroachment on animal habitats is bringing more species into proximity, leading to an increased rate of zoonosis[2] and accelerating the Anthropocene mass extinction[48]. While Serratus enhances our capability to chronicle the full genetic diversity of our planet, the genetic diversity of the biosphere is diminishing. Thus, investment in the collection and curation of biologically diverse samples, with an emphasis on geographically underrepresented regions, has never been more pressing—if not for the conservation of endangered species, then to better conserve our own.

Online content

Any methods, additional references, Nature Research reporting summaries, source data, extended data, supplementary information, acknowledgements, peer review information; details of author contributions and competing interests; and statements of data and code availability are available at https://doi.org/10.1038/s41586-021-04332-2.

1. Leinonen, R., Sugawara, H. & Shumway, M. The Sequence Read Archive. *Nucleic Acids Res.* **39**, D19–D21 (2011).
2. Anthony, S. J. et al. A strategy to estimate unknown viral diversity in mammals. *mBio* **4**, e00598-13 (2013).
3. Johnson, C. K. et al. Global shifts in mammalian population trends reveal key predictors of virus spillover risk. *Proc. R. Soc. B* **287**, 20192736 (2020).
4. Carroll, D. et al. The Global Virome Project. *Science* **359**, 872–874 (2018).
5. Shi, M. et al. The evolutionary history of vertebrate RNA viruses. *Nature* **556**, 197–202 (2018).
6. Wahba, L. et al. An extensive meta-metagenomic search identifies SARS-CoV-2-homologous sequences in pangolin lung viromes. *mSphere* **5**, 00160-20 (2020).
7. Wolf, Y. I. et al. Doubling of the known set of RNA viruses by metagenomic analysis of an aquatic virome. *Nat. Microbiol.* **5**, 1262–1270 (2020).
8. Mitchell, A. L. et al. MGnify: the microbiome analysis resource in 2020. *Nucleic Acids Res.* **48**, D570–D578 (2020).
9. Chen, I.-M. A. et al. The IMG/M data management and analysis system v.6.0: new tools and advanced capabilities. *Nucleic Acids Res.* **49**, D751–D763 (2021).
10. Camarillo-Guerrero, L. F., Almeida, A., Rangel-Pineros, G., Finn, R. D. & Lawley, T. D. Massive expansion of human gut bacteriophage diversity. *Cell* **184**, 1098–1109 (2021).
11. Nayfach, S. et al. A genomic catalog of Earth's microbiomes. *Nat. Biotechnol.* **39**, 499–509 (2021).
12. Moore, R. A. et al. The sensitivity of massively parallel sequencing for detecting candidate infectious agents associated with human tissue. *PLoS One* **6**, e19838 (2011).
13. NIH. *STRIDES Initiative—Data Science at NIH* https://datascience.nih.gov/strides (2021).
14. Buchfink, B., Reuter, K. & Drost, H.-G. Sensitive protein alignments at tree-of-life scale using DIAMOND. *Nat. Methods* **18**, 366–368 (2021).
15. Karasikov, M. et al. MetaGraph: indexing and analysing nucleotide archives at petabase-scale. Preprint at https://www.biorxiv.org/content/10.1101/2020.10.01.322164v2 (2020).
16. Katz, K. S. et al. STAT: a fast, scalable, MinHash-based k-mer tool to assess Sequence Read Archive next-generation sequence submissions. *Genome Biol.* **22**, 270 (2021).
17. Koonin, E. V. & Dolja, V. V. Virus world as an evolutionary network of viruses and capsidless selfish elements. *Microbiol. Mol. Biol. Rev.* **78**, 278–303 (2014).
18. Babaian, A. & Edgar, R. C. Ribovirus classification by a polymerase barcode sequence. Preprint at https://www.biorxiv.org/content/10.1101/2021.03.02.433648v1 (2021).
19. Wolf, Y. I. et al. Origins and evolution of the global RNA virome. *mBio* **9**, e0239-18 (2018).
20. Obbard, D. J., Shi, M., Roberts, K. E., Longdon, B. & Dennis, A. B. A new lineage of segmented RNA viruses infecting animals. *Virus Evol.* **6**, vez061 (2020).

21. Altschul, S. F. et al. Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Res.* **25**, 3389–3402 (1997).
22. Koonin, E. V. et al. Global organization and proposed megataxonomy of the virus world. *Microbiol. Mol. Biol. Rev.* **84**, e00061-19 (2020).
23. Bukhari, K. et al. Description and initial characterization of metatranscriptomic nidovirus-like genomes from the proposed new family Abyssoviridae, and from a sister group to the *Coronavirinae*, the proposed genus Alphaletovirus. *Virology* **524**, 160–171. (2018).
24. Mordecai, G. J. et al. Endangered wild salmon infected by newly discovered viruses. *eLife* **8**. e47615 (2019).
25. Meleshko, D., Hajirasouliha, I. & Korobeynikov, A. coronaSPAdes: from biosynthetic gene clusters to RNA viral assemblies. *Bioinformatics* **38**, 1–8 (2022).
26. Tao, Y. et al. Broad-range virus detection and discovery using microfluidic PCR coupled with high-throughput sequencing. Preprint at https://www.biorxiv.org/content/10.1101/2020.06.10.145052v1 (2020).
27. Tsai, S. L., Baselga-Garriga, C. & Melton, D. A. Midkine is a dual regulator of wound epidermis development and inflammation during the initiation of limb regeneration. *eLife* **9**, e50765 (2020).
28. Sabin, K. Z., Jiang, P., Gearhart, M. D., Stewart, R. & Echeverri, K. AP-1 cFos/JunB /miR-200a regulate the pro-regenerative glial cell response during axolotl spinal cord regeneration. *Commun. Biol.* **2**, 91 (2019).
29. Miller, A. K. et al. Slippery when wet: cross-species transmission of divergent coronaviruses in bony and jawless fish and the evolutionary history of the Coronaviridae. *Virus Evol.* **7**. veab050 (2021).
30. Mukherjee, S. et al. Genomes OnLine Database (GOLD) v.8: overview and updates. *Nucleic Acids Res.* **49**, D723–D733 (2021).
31. Stanaway, J. D. et al. The global burden of viral hepatitis from 1990 to 2013: findings from the Global Burden of Disease Study 2013. *Lancet* **388**, 1081–1088 (2016).
32. Taylor, J. M. Infection by hepatitis delta virus. *Viruses* **12**, 648 (2020).
33. Szirovicza, L. et al. Snake deltavirus utilizes envelope proteins of different viruses to generate infectious particles. *mBio* **11**, e03250-19 (2020).
34. Wille, M. et al. A divergent hepatitis D-like agent in birds. *Viruses* **12**, 720 (2018).
35. Chang, W.-S. et al. Novel hepatitis D-like agents in vertebrates and invertebrates. *Virus Evol.* **5**, vez021 (2019).
36. Paraskevopoulou, S. et al. Mammalian deltavirus without hepadnavirus coinfection in the neotropical rodent *Proechimys semispinosus*. *Proc. Natl Acad. Sci. USA* **117**, 17977–17983 (2020).
37. Bergner, L. M. et al. Diversification of mammalian deltaviruses by host shifting. *Proc. Natl Acad. Sci. USA* **118**, e2019907118 (2021).
38. Iwamoto, M. et al. Identification of novel avian and mammalian deltaviruses provides new insights into deltavirus evolution. *Virus Evol.* **7**, veab003 (2021).
39. De la Peña, M., Ceprián, R., Casey, J. L. & Cervera, A. Hepatitis delta virus-like circular RNAs from diverse metazoans encode conserved hammerhead ribozymes. *Virus Evol.* **7**, veab016 (2021).
40. Zuccola, H. J., Rozzelle, J. E., Lemon, S. M., Erickson, B. W. & Hogle, J. M. Structural basis of the oligomerization of hepatitis delta antigen. *Structure* **6**, 821–830 (1998).
41. Flores, R., Gago-Zachert, S., Serra, P., Sanjuán, R. & Elena, S. F. Viroids: survivors from the RNA world? *Annu. Rev. Microbiol.* **68**, 395–414 (2014).
42. Al-Shayeb, B. et al. Clades of huge phages from across Earth's ecosystems. *Nature* **578**, 425–431 (2020).
43. Devoto, A. E. et al. Megaphages infect *Prevotella* and variants are widespread in gut microbiomes. *Nat. Microbiol.* **4**, 693–700 (2019).
44. Bradley, P., Den Bakker, H. C., Rocha, E. P. C., McVean, G. & Iqbal, Z. Ultra-fast search of all deposited bacterial and viral genomic data. *Nat. Biotechnol.* **37**, 152–159 (2019).
45. Baker, D. et al. No more business as usual: agile and effective responses to emerging pathogen threats require open data and open analytics. *PLoS Pathog.* **16**, e1008643 (2020).
46. Letko, M., Seifert, S. N., Olival, K. J., Plowright, R. K. & Munster, V. J. Bat-borne virus diversity, spillover and emergence. *Nat. Rev. Microbiol.* **18**, 461–471 (2020).
47. Letko, M., Marzi, A. & Munster, V. Functional assessment of cell entry and receptor usage for SARS-CoV-2 and other lineage B betacoronaviruses. *Nat. Microbiol.* **5**, 562–569 (2020).
48. Chase, J. M., Blowes, S. A., Knight, T. M., Gerstner, K. & May, F. Ecosystem decay exacerbates biodiversity loss with habitat loss. *Nature* **584**, 238–243 (2020).
49. Thompson, A. A. & Peersen, O. B. Structural basis for proteolysis-dependent activation of the poliovirus RNA-dependent RNA polymerase. *EMBO J.* **23**, 3462–3471 (2004).
50. Zuker, M. Mfold web server for nucleic acid folding and hybridization prediction. *Nucleic Acids Res.* **31**, 3406–3415 (2003).

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

© The Author(s), under exclusive licence to Springer Nature Limited 2022

Methods

Serratus alignment architecture

Serratus (v0.3.0) (https://github.com/ababaian/serratus) is an open-source cloud-infrastructure designed for ultra-high-throughput sequence alignment against a query sequence or pangenome (Extended Data Fig. 1). Serratus compute costs are dependent on search parameters (expanded discussion available: https://github.com/ababaian/serratus/wiki/pangenome_design). The nucleotide vertebrate viral pangenome search (bowtie2, database size: 79.8 MB) reached processing rates of 1.29 million SRA runs in 24 h at a cost of US$0.0062 per dataset (Extended Data Fig. 1). The translated-nucleotide RdRP search (DIAMOND[14]; database size: 7.1 MB) reached processing rates exceeding 0.5 million SRA runs in 12 h at a cost of US$0.0042 per dataset. All 5,686,715 runs analysed in the RdRP search were completed within 11 days for a total cost of US$23,980 or around US$2,350 per petabase. For a detailed breakdown of Serratus project costs and recommendations for managing cloud-computing costs, see Serratus wiki: https://github.com/ababaian/serratus/wiki/budget. Tutorials on how to find particular novel viruses using Serratus data are available at https://github.com/ababaian/serratus /wiki/Find_novel_viruses.

Computing cluster architecture

The processing of each sequencing library is split into three modules: 'dl' (download), 'align' and 'merge'. The dl module acquires compressed data (.sra format) via prefetch (v2.10.4), from the Amazon Web Services (AWS) Simple Storage Service (S3) mirror of the SRA, decompresses to FASTQ with fastq-dump (v2.10.4) and splits the data into chunks of 1 million reads or read-pairs ('fq-blocks') into a temporary S3 cache bucket. To mitigate excessive disk usage caused by a few large datasets, a total limit of 100 million reads per dataset was imposed. The align module reads individual fq-blocks and aligns to an indexed database of user-provided query sequences using either bowtie2 (v2.4.1, *--very-sensitive-local*)[51] for nucleotide search, or DIAMOND (v2.0.6 development version, *--mmap-target-index --target-indexed --masking 0 --mid-sensitive -s 1 -c1 -p1 -k1 -b 0.75*)[14] for translated-protein search. Finally, the merge module concatenates the aligned blocks into a single output file (.bam for nucleotide, or .pro for protein) and generates alignment statistics with a Python script (see details about Summarizer in 'Generating viral summary reports' below).

Computing resource allocation

Each component is launched from a separate AWS autoscaling group with its own launch template, allowing the user to tailor instance requirements per task. This enabled us to minimize the use of costly block storage during compute-bound tasks such as alignment. We used the following Spot instance types; dl: 250 GB SSD block storage, 8 virtual CPUs (vCPUs), 32 GB RAM (r5.xlarge) around 1,300 instances; align: 10 GB SSD block storage, 8 vCPUs, 8 GB RAM (c5.xlarge) around 4,300 instances; merge: 150 GB SSD block storage, 4 vCPUs, 4 GB RAM (c5.large) around 60 instances. Users should note that it may be necessary to submit a service ticket to access more than the default EC2 instance limit.

AWS Elastic Compute Cloud (EC2) instances have higher network bandwidth (up to 1.25 GB s^{-1}) than block storage bandwidth (250 MB s^{-1}). To exploit this, we used S3 buckets as a data buffering and streaming system to transfer data between instances following methods developed in a previous cloud architecture (https://github.com/FredHutch/sra-pipeline). This, combined with splitting of FASTQ files into individual blocks, effectively eliminated file input/output (i/o) as a bottleneck, as the available i/o is multiplied per running instance (conceptually analogous to a RAID0 configuration or a Hadoop distributed file system[52]).

Using S3 as a buffer also allowed us to decouple the input and output of each module. S3 storage is cheap enough that in the event of unexpected issues (for example, exceeding EC2 quotas) we could resolve system problems in real time and resume data processing. For example, shutting down the align modules to hotfix a genome indexing problem without having to re-run the dl modules, or if an alignment instance is killed by a Spot termination, only that block needs to be reprocessed instead of the entire sequencing run.

Work queue and scheduling

The Serratus scheduler node controls the number of desired instances to be created for each component of the workflow, based on the available work queue. We implemented a pull-based work queue. After boot-up, each instance launches a number of 'worker' threads equal to the number of CPUs available. Each worker independently manages itself via a boot script, and queries the 'scheduler' for available tasks. Upon completion of the task, the worker updates the scheduler of the result: success, or fail, and queries for a new task. Under ideal conditions, this allows for a worst-case response rate in the hundreds of milliseconds, keeping cluster throughput high. Each task typically lasts several minutes depending on the pangenome.

The scheduler itself was implemented using Postgres (for persistence and concurrency) and Flask (to pool connections and translate REST queries into SQL). The Flask layer allowed us to scale the cluster past the number of simultaneous sessions manageable by a single Postgres instance. The work queue can also be managed manually by the user, to perform operations such as re-attempting the downloading of an SRA accession after a failure or to pause an operation while debugging. Up to 300,000 SRA jobs can be processed in the work queue per batch process.

The system is designed to be fully self-scaling. An 'autoscaling controller' was implemented, which scales-in or scales-out the desired number of instances per task every five minutes on the basis of the work queue. As a backstop, when all workers on an instance fail to receive work instructions from the scheduler, the instance self shuts-down. Finally a 'job cleaner' component checks the active jobs against currently running instances. If an instance has disappeared owing to SPOT termination or manual shutdown, it resets the job allowing it to be processed up by the next available instance.

To monitor cluster performance in real-time, we used Prometheus (v2.5.0) and node exporter to retrieve CPU, disk, memory and networking statistics from each instance, to expose performance information about the work queue, and Python exporter to export information from the Flask server. This allowed us to identify and diagnose performance problems within minutes to avoid costly overruns.

Generating viral summary reports

We define a viral pangenome as the entire collection of reference sequences belonging to a taxonomic viral family, which may contain both full-length genomes and sequence fragments such as those obtained by RdRP amplicon sequencing.

We developed a Summarizer module written in Python to provide a compact, human- and machine-readable synopsis of the alignments generated for each SRA dataset. The method was implemented in Serratus_summarizer.py for nucleotide alignment and Serratus_psummarizer.py for amino acid alignments. Reports generated by the Summarizer are text files with three sections described in detail online (https://github.com/ababaian/serratus/wiki/.summary-Reports). In brief, each contains a header section with alignment metadata and one-line summaries for each virus family pangenome, reference sequence and gene, respectively, with gene summaries provided for protein alignments only.

For each summary line we include descriptive statistics gathered from the alignment data such as the number of aligned reads, estimated read depth, mean alignment identity and coverage; that is, the distribution of reads across each reference sequence or pangenome. Coverage is measured by dividing a reference sequence into 25 equal

bins and depicted as an ASCII text string of 25 symbols, one per bin; for example *oaooomoUU:oWWUUWOWamWAAUW*. Each symbol represents $\log_2(n+1)$, where n is the number of reads aligned to a bin in this order _..:*uwaomUWAOM*^. Thus, '_' indicates no reads, '.' exactly one read, ':' two reads, 'u' 3–4 reads, 'w' 5–7 reads and so on; '^' represents $>2^{13} = 8{,}192$ reads in the bin. For a pangenome, alignments to its reference sequences are projected onto a corresponding set of 25 bins. For a complete genome, the projected pangenome bin number 1, 2, ..., 25 is the same as the reference sequence bin number. For a fragment, a bin is projected onto the pangenome bin implied by the alignment of the fragment to a complete genome. For example, if the start of a fragment aligns halfway into a complete genome, bin 1 of the fragment is projected to bin floor(25/2) = 12 of the pangenome. The introduction of pangenome bins was motivated by the observation that bowtie2 selects an alignment at random when there are two or more top-scoring alignments, which tends to distribute coverage over several reference sequences when a single viral genome is present in the reads. Coverage of a single reference genome may therefore be fragmented, and binning to a pangenome better assesses coverage over a putative viral genome in the reads while retaining pangenome sequence diversity for detection.

Identification of viral families within a sequencing dataset

The Summarizer implements a binary classifier predicting the presence or absence of each virus family in the query on the basis of pangenome-aligned short reads. For a given family F, the classifier reports a score in the range [0,100] with the goal of assigning a high score to a dataset if it contains F and a low score if it does not. Setting a threshold on the score divides datasets into disjoint subsets representing predicted positive and negative detections of family F. The choice of threshold implies a trade-off between false positives and false negatives. Sorting by decreasing score ranks datasets in decreasing order of confidence that F is present in the reads.

Naively, a natural measure of the presence of a virus family is the number of alignments to its reference sequences. However, alignments may be induced by non-homologous sequence similarity, for example low-complexity sequence.

The score for a family was therefore designed to reflect the overall coverage of a pangenome because coverage across all or most of a pangenome is more likely to reflect true homology; that is, the presence of a related virus. Ideally, coverage would be measured individually for each base in the reference sequence, but this could add undesirable overhead in compute time and memory for a process that is executed in the Linux alignment pipe (FASTQ decompression → aligner → Summarizer → alignment file compression). Coverage was therefore measured by binning as described above, which can be implemented with minimal overhead.

A virus that is present in the reads with coverage too low to enable an assembly may have less practical value than an assembled genome. Also, genomes with lower identity to previously known sequences will tend to contain more novel biological information than genomes with high identity but highly diverged genomes will tend to have fewer aligned reads. With these considerations in mind, the classifier was designed to give higher scores when coverage is high, read depth is high and/or identity is low. This was accomplished as follows. Let H be the number of bins with at least 8 alignments to F, and L be the number of bins with from 1 to 7 alignments. Let S be the mean alignment percentage identity, and define the identity weight $w = (S/100)^{-3}$, which is designed to give higher weight to lower identities, noting that w is close to 1 when identity is close to 100% and increases rapidly at lower identities. The classification score for family F is calculated as $Z_F = \max(w(4H + L)),100)$. By construction, Z_F has a maximum of 100 when coverage is consistently high across a pangenome, and is also high when identity is low and coverage is moderate, which may reflect high read depth but many false negative alignments due to low identity. Thus, Z_F is greater than zero when there is at least one alignment to F and assigns higher scores to SRA datasets that are more likely to support successful assembly of a virus belonging to F.

Sensitivity to novel viruses as a function of identity

We aimed to assess the sensitivity of our pipeline as a function of sequence identity by asking what fraction of novel viruses is detected at increasingly low identities compared to the reference sequences used for the search. Several variables other than identity affect sensitivity, including read length, whether reads are mate-paired, sequencing error rate, coverage bias and the presence of other similar viruses that may cause some variants to be unreported in the contigs. Coverage bias can render a virus with high average read depth undetectable, in particular if the query is RdRP-only and the RdRP gene has low coverage or is absent from the reads. Successful detection might be defined in different ways, depending on the goals of the search; for example, a single local alignment of a reference to a read (maximizing sensitivity, but not always useful in practice); a microassembled palmprint; a full assembly contig that contains a complete palmprint or otherwise classifiable fragment of a marker gene; or an assembly of a complete genome. We assessed alignment sensitivity of *bowtie2 --very-sensitive-local* and Serratus-optimized DIAMOND[14] as a function of identity by simulating typical examples in a representative scenario: unpaired reads of length 100 with a base call error rate of 1%. We manually selected test-reference pairs of RefSeq complete *Ribovirus* genomes at RdRP amino acid identities 100%, 95% ... 20%, generating simulated length-100 reads at uniformly distributed random locations in the test genome with a mean coverage of 1,000×. For bowtie2, the complete reference genome was used as a reference; for DIAMOND the reference was the translated amino acid sequence of the RdRP gene (400 amino acids), which was identified by aligning to the 'wolf18' dataset. These choices model the coronavirus pangenome used as a bowtie2 query and the rdrp1 protein reference used as a DIAMOND query, respectively. Sensitivity was assessed as the fraction of reads aligned to the reference. With bowtie2, the number of unmapped reads reflects a combination of lack of alignment sensitivity and divergence in gene content as some regions of the genome may lack homology to the reference. With DIAMOND, the number of unmapped reads reflects a combination of lack of alignment sensitivity and the fraction of the genome that is not RdRP, which varies by genome length 1g. They show that the fraction of aligned reads by bowtie2 drops to around 2% to 4% at 90% RdRP amino acid identity, and maps no reads for most of the lower identity test–reference pairs. DIAMOND maps around 5% to 10% of reads down to 50% RdRP amino acid identity, then less than 1% at lower identities; around 30% to 35% is the lower limit of practical detection.

Defining viral pangenomes and the SRA search space

Nucleotide search pangenomes. To create a collection of viral pangenomes, a comprehensive set of complete and partial genomes representing the genetic diversity of each viral family, we used two approaches.

For *Coronaviridae*, we combined all RefSeq ($n = 64$) and GenBank ($n = 37{,}451$) records matching the NCBI Nucleotide[53] server query "txid11118[Organism:exp]" (date accessed: 1 June 2020). Sequences of fewer than 200 nt were excluded as well as sequences identified to contain non-CoV contaminants during preliminary testing (such as plasmid DNA or ribosomal RNA fragments). Remaining sequences were clustered at 99% identity with UCLUST (USEARCH: v11.0.667)[54] and masked by Dustmasker (ncbi-blast:2.10.0) (*--window 30 and --window 64*)[55]. The final query contained 10,101 CoV sequences (accessions in Supplementary Table 1a; masked coordinates in Supplementary Table 1b). SeqKit (v0.15) was used for working with fasta files[56].

For all other vertebrate viral family pangenomes, RefSeq sequences ($n = 2{,}849$) were downloaded from the NCBI Nucleotide server with the query "*Viruses*[Organism] AND srcdb refseq[PROP] NOT wgs[PROP] NOT cellular organisms[ORGN] NOT AC 000001:AC 999999[PACC]

AND ("vhost human"[Filter] AND "vhost vertebrates"[Filter])" (date accessed: 17 May 2020). Retroviruses (n = 80) were excluded as preliminary testing yielded excessive numbers of alignments to transcribed endogenous retroviruses. Each sequence was annotated with its taxonomic family according to its RefSeq record; those for which no family was assigned by RefSeq (n = 81) were designated as 'unknown'.

The collection of these pangenomes was termed 'cov3m', and was the nucleotide sequence reference used for this study.

Amino acid viral RdRP search panproteome. For the translated-nucleotide search of viral RNA-dependent RNA polymerase (RdRP; hereinafter viral RdRP is implied) we combined sequences from several sources. (1) The 'wolf18' collection is a curated snapshot (around 2018) of RdRP from GenBank (ref. [19] accessed: ftp://ftp.ncbi.nlm.nih.gov/pub/wolf/_suppl/rnavir18/RNAvirome.S2.afa). (2) The 'wolf20' collection is RdRPs from assembled from marine metagenomes (ref. [7] accessed: ftp://ftp.ncbi.nlm.nih.gov/pub/wolf/_suppl/yangshan/gb_rdrp.afa). (3) All viral GenBank protein sequences were aligned with DIAMOND *--ultra-sensitive*[14] against the combined wolf18 and wolf20 sequences (E-value < 1×10^{-6}). These produced local alignments that contained truncated RdRP, so each RdRP-containing GenBank sequence was then re-aligned to the wolf18 and wolf20 collection to 'trim' them to 'wolf' RdRP boundaries. (4) The above algorithm was also applied to all viral GenBank nucleotide records to capture additional RdRP not annotated as such by GenBank. A region of HCV capsid protein shares similarity to HCV RdRP; sequences annotated as HCV capsid were therefore removed. Eight novel coronavirus RdRP sequences identified in a pilot experiment were added manually. The combined RdRP sequences from the above collections were clustered (UCLUST) at 90% amino acid identity and the resulting representative sequences (centroids, n = 14,653) used as the rdrp1 search query.

In addition, we added delta virus antigen proteins from NC 001653, M21012, X60193, L22063, AF018077, AJ584848, AJ584847, AJ584844, AJ584849, MT649207, MT649208, MT649206, NC 040845, NC 040729, MN031240, MN031239, MK962760, MK962759 and eight additional homologues we identified in a pilot experiment.

SRA search space and queries. To run Serratus, a target list of SRA run accessions is required. We defined 11 (not-mutually exclusive) queries as our search space, which were named human, mouse, mammal, vertebrate, invertebrate, eukaryotes, prokaryotes/others, bat (including genomic sequences), virome, metagenome and mammalian genome (Supplementary Table 1c). Our search was restricted to Illumina sequencing technologies and to RNA-seq, meta-genomic and meta-transcriptome library types for these organisms (except for the mammalian genome query, which was genome or exome). Before each Serratus deployment, target lists were depleted of accessions already analysed. Reprocessing of a failed accession was attempted at least twice. In total, we aligned 3,837,755/4,059,695 (94.5%) of the runs in our nucleotide-pangenome search (around May 2020) and 5,686,715/5,780,800 (98.37%) of the runs in our translated-nucleotide RdRP search (around January 2021).

User interfaces for the Serratus databases

We implemented an on-going, multi-tiered release policy for code and data generated by this study, as follows. All code, electronic notebooks and raw data are immediately available at https://github.com/ababaian/serratus and on the s3://serratus-public bucket, respectively. Upon completion of a project milestone, a structured data release is issued containing raw data into our viral data warehouse s3://lovelywater/. For example, the .bam nucleotide alignment files from 3.84 million SRA runs are stored in s3://lovelywater/bam/X.bam; and the protein .summary files are in s3://lovelywater/psummary/X.psummary, where X is a SRA run accession. These structured releases enable downstream and third-party programmatic access to the data.

Summary files for every searched SRA dataset are parsed into a publicly accessible AWS Relational Database (RDS) instance that can be queried remotely via any PostgreSQL client. This enables users and programs to perform complex operations such as retrieving summaries and metadata for all SRA runs matching a given reference sequence with above a given classifier score threshold. For example, one can query for all records containing at least 20 aligned reads to hepatitis delta virus (NC 001653.2) and the associated host taxonomy for the corresponding SRA datasets:

```
SELECT sequence_accession, run_id, tax_id, n_readsFROM nsequence
JOINsrarun ON (nsequence.run_id = srarun.run) WHERE n_reads >= 20
```

For users unfamiliar with SQL, we developed Tantalus (https://github.com/serratus-bio/tantalus, an R programming-language package that directly interfaces the Serratus PostgreSQL database to retrieve summary information as data-frames. Tantalus also offers functions to explore and visualize the data.

Finally, the Serratus data can be explored via a graphical web interface by accession, virus or viral family at https://serratus.io/explorer. Under the hood, we developed a REST API to query the database from the website. The website uses React+D3.js to serve graphical reports with an overview of viral families found in each SRA accession matching a user query.

All four data access interfaces are under ongoing development, receiving community feedback via their respective GitHub issue trackers to facilitate the translation of this data collection into an effective viral discovery resource. Documentation for data access methods is available at https://serratus.io/access.

Geocoding BioSamples. To generate the map in Fig. 1c, we parsed and extracted geographical information from all 16 million BioSample XML submissions[57]. Geographic information is either in the form of coordinates (latitude and longitude) or freeform text (for example, 'France', 'Great Lakes'). For each BioSample, coordinate extraction was attempted using regular expressions. If that failed, text extraction was attempted using a manually curated list of keywords that capture BioSample attribute names that are likely to contain geographical information. If that failed, then we were unable to extract geographical information for that BioSample. Geocoding the text to coordinates was done using Amazon Location Service on a reduced set of distinct filtered text values (52,028 distinct values from 2,760,241 BioSamples with potential geographical text). BioSamples with geocoded coordinates were combined with BioSamples with submitted coordinate information to form a set of 5,325,523 geospatial BioSamples. This is then cross-referenced with our subset of SRA accessions with an RdRP match to generate the figure.

All intermediate and resulting data from this step are stored on the SQL database described above. Development work is public at https://github.com/serratus-bio/biosample-sql.

Viral alignment, assembly and annotation

Upon identification of CoV reads in a run from alignment, we assembled 52,772 runs containing at least 10 reads that aligned to our CoV pangenome or at least 2 reads with CoV-positive k-mers[16]. A total of 11,120 of the resulting assemblies contained identifiable CoV contigs, of which only 4,179 (37.58%) contained full-length CoV RdRP (Supplementary Table 1d). The discrepancy between alignment-positive, assembly-positive and RdRP-positive libraries arises owing to random sampling of viral reads and assembly fragmentation. In this respect, alignment or k-mer based methods are more sensitive than assembly in detecting for the presence of low-abundance viruses (genome coverage < 1) with high identity to a reference sequence. Scoring libraries for genome coverage and depth is a good predictor of ultimate assembly success (Extended Data Fig. 3); thus, it can be used to efficiently prioritize computationally expensive assembly in the future, as has been previously demonstrated for large-scale SRA alignment analyses[58].

DIAMOND optimization and output. To optimize DIAMOND[14] for small (<10 MB) databases such as the RdRP search database, we built a probabilistic hash set that stores 8-bit hash values for the database seeds, using SIMD instructions for fast probing. This index is loaded as a memory mapped file to be shared among processes and allows us to filter the query reads for seeds contained in the database, thus omitting the full construction of the query seed table. We also eliminated the overhead of building seed distribution histograms that is normally required to allocate memory and construct the query table in a single pass over the data using a deque-like data structure. In addition, query reads were not masked for simple repeats, as the search database is already masked. These features are available starting from DIAMOND v2.0.8 with the command line flags --target-indexed --masking 0. In a benchmark of 4 sets of 1 million reads from a bat metagenome (ERR2756788), the implemented optimization produced a speed-up of ×1.47 and reduced memory use by 64%, compared to the public unmodified DIAMOND v2.0.6, using our optimized set of parameters in both cases (see 1.1.1). Together, the optimized parameters and implementation reduced DIAMOND runtime against RdRP search from 197.96 s (s.d. = 0.18 s), to 21.29 s (s.d. = 0.23 s) per million reads, a speed-up of a factor of 9.3. This effectively reduced the computational cost of translated-nucleotide search for Serratus from US$0.03 to US$0.0042 per library.

DIAMOND output files (we label .pro) were specified with the command -f 6 qseqid qstart qend qlen qstrand sseqid sstart send slen pident evalue cigar qseq_translated full_qseq full_qseq_mate.

coronaSPAdes. RNA viral genome assembly faces several distinct challenges stemming from technical and biological bias in sequencing data. During library preparation, reverse transcription introduces 50 end coverage bias, and GC-content skew and secondary structures lead to unequal PCR amplification[59]. Technical bias is confounded by biological complexity such as intra-sample sequence variation due to transcript isoforms and/or to the presence of multiple strains.

To address the assembly challenges specific to RNA viruses, we developed coronaSPAdes (v3.15.3), which is described in detail in a companion manuscript[25]. In brief, rnaviralSPAdes and the more specialized variant, coronaSPAdes, combines algorithms and methods from several previous approaches based on metaSPAdes[60], rnaSPAdes[61] and metaviralSPAdes[62] with a HMMPathExtension step. coronaSPAdes constructs an assembly graph from an RNA-seq dataset (transcriptome, meta-transcriptome, and meta-virome are supported), removing expected sequencing artifacts such as low complexity (poly-A/poly-T) tips, edges, single-strand chimeric loops or double-strand hairpins[61] and subspecies-bases variation[62].

To deal with possible misassemblies and high-covered sequencing artefacts, a secondary HMMPathExtension step is performed to leverage orthogonal information about the expected viral genome. Protein domains are identified on all assembly graphs using a set of viral hidden Markov models (HMMs), and similar to biosyntheticSPAdes[63], HMMPathExtension attempts to find paths on the assembly graph that pass through significant HMM matches in order.

coronaSPAdes is bundled with the Pfam SARS-CoV-2 set of HMMs[64], although these may be substituted by the user. This latter feature of coronaSPAdes was used for HDV assembly, in which the HMM model of HDAg, the hepatitis delta antigen, was used instead of the Pfam SARS-CoV-2 set. Note that despite the name, the HMMs from this set are quite general, modelling domains found in all coronavirus genera in addition to RdRP, which is found in many RNA virus families. Hits from these HMMs cover most bases in most known coronavirus genomes, enabling the recovery of strain mixtures and splice variants.

Microassembly of RdRP-aligned reads. Reads aligned by DIAMOND[14] in the translated-nucleotide RdRP search are stored in the .pro alignment file. All sets of mapped reads (3,379,127 runs) were extracted, and each non-empty set was assembled with rnaviralSPAdes (v3.15.3)[25] using default parameters. This process is referred to as 'microassembly', as a collection of DIAMOND hits is orders of magnitude smaller than the original SRA accession (40 ± 534 KB compressed size, ranging from a single read up to 53 MB). Then bowtie2[51] (default parameters) was used to align the DIAMOND read hits of an accession back to the microassembled contigs of that accession. Palmscan (v1.0.0, *-rdrp -hicon*)[18] was run on microassembled contigs, resulting in high-confidence palmprints for 337,344 contigs. Finally mosdepth (v0.3.1)[65] was used to calculate a coverage pileup for each palmprint hit region within microassembled contigs.

Classification of assembled RdRP sequences. Our methods for RdRP classification are described and validated in a companion paper[18]. In brief, we defined a barcode sequence, the polymerase palmprint (PP), as an approximately 100-amino-acid segment of the RdRP palm subdomain delineated by well-conserved catalytic motifs. We implemented an algorithm, Palmscan, to identify palmprint sequences and discriminate RdRPs from reverse transcriptases. The combined set of RdRP palmprints from public databases and our assemblies was classified by clustering into operational taxonomic units (OTUs) at 90%, 75% and 40% identity, giving species-like, genus-like and family-like clusters (sOTUs, gOTUs and fOTUs), respectively. Tentative taxonomy of novel OTUs was assigned by aligning to palmprints of named viruses and taking a consensus of the top hits above the identity threshold for each rank.

Quality control of assembled RdRP sequences. Our goal was to identity novel viral RdRP sequences and novel sOTUs in SRA libraries. From this perspective, we considered the following to be erroneous to varying degrees: sequences that are (a) not polymerases; (b) not viral; (c) with differences due to experimental artefacts; or (d) with sufficient differences to cause a spurious inference of a novel sOTU. We categorized potential sources of such errors and implemented quality control procedures to identify and mitigate them, as follows.

Point errors are single-letter substitution and indel errors that may be caused by PCR or sequencing per se. Random point errors are not reproduced in multiple non-PCR duplicate reads and are unlikely to assemble because such errors almost always induce identifiable structures in the assembly graph (tips and bubbles) that are pruned during graph simplification. In rare cases, a contig may contain a read with random point errors. Such contigs will have low coverage of around 1, and we therefore recorded coverage as a quality control metric and assessed whether low-coverage assemblies were anomalous compared to high-coverage assemblies by measures such as the frequencies with which they are reproduced in multiple libraries compared to exactly one library, finding no noticeable difference when coverage is low.

Chimeras of polymerases from different species could arise from PCR amplification or assembly. We used the UCHIME2 (usearch v8.0.1623) algorithm[66] to screen assembled palmprint sequences, finding no high-scoring putative chimeras. Mosaic sequences formed by joining a polymerase to unrelated sequence would either have an intact palmprint, in which case the mosaic would be irrelevant to our analysis, or would be rejected by Palmscan owing to the lack of delimiting motifs.

Reverse transcriptases are homologous to RdRP. Retroviral insertions into host genomes induce ubiquitous sequence similarity between host genomes and viral RdRP. Palmscan was designed to discriminate RdRP from sequences of reverse transcriptase origin. Testing on a large decoy set of non-RdRP sequences with recognizable sequence similarity showed that the Palmscan false discovery rate for RdRP identification is 0.001. We estimated the probability of false positive matches in unrelated sequence by generating sufficient random nucleotide and amino acid sequences to show that the expected number of false positive palmprint identifications is zero in a dataset of comparable size to our assemblies. We also regard the low observed frequency of palmprints in DNA whole-genome sequencing data (in 2.6 Pbp or 25.8% of reads, accounted for 100 known palmprints and 95 novel palmprints

or 0.13% of the total identified) as a de facto confirmation of the low probability false positives in unrelated sequence.

Endogenous viral elements (EVEs; that is, insertions of viral sequence into host genomes that are potentially degraded and non-functional) cannot be distinguished from viral genomes on the basis of the palmprint sequence alone. To assess the frequency of EVEs in our data, we re-assembled 890 randomly chosen libraries yielding one or more palmprints using all reads, extracted the 23,530 resulting contigs with a positive palmprint hit by Palmscan, and classified them using Virsorter2 (v2.1)[67]. Of these contigs, 11,914 were classified as viral, confirming the Palmscan identification; 49 as Viridiplantae (green plants); 46 as Metazoa; 25 as Fungi and the remainder were unclassified. Thus, 120/12,034 = 1% of the classified contigs were predicted as non-viral, suggesting that the frequency of EVEs in the reported palmprints is around 1%.

Annotation of CoV assemblies. Accurate annotation of CoV genomes is challenging owing to ribosomal frameshifts and polyproteins that are cleaved into maturation proteins[68], and thus previously annotated viral genomes offer a guide to accurate gene-calls and protein functional predictions. However, although many of the viral genomes we were likely to recover would be similar to previously annotated genomes in Refseq or GenBank, we anticipated that many of the genomes would be taxonomically distant from any available reference. To address these constraints, we developed an annotation pipeline called DARTH (version maul)[69] which leverages both reference-based and ab initio annotation approaches.

In brief, DARTH consists of the following phases: standardize the ordering and orientation of assembly contigs using conserved domain alignments, perform reference-based annotation of the contigs, annotate RNA secondary structure, ab initio gene-calling, generate files for aiding assembly and annotation diagnostics, and generate a master annotation file. It is important to put the contigs in the 'expected' orientation and ordering to facilitate comparative analysis of synteny and as a requirement for genome deposition. To perform this standardization, DARTH generates the six-frame translation of the contigs using the transeq (EMBOSS:6.6.0.0)[70] and uses HMMER3 (v3.3.2)[71] to search the translations for Pfam domain models specific to CoV[64]. DARTH compares the Pfam accessions from the HMMER alignment to the NCBI SARS-CoV-2 reference genome (NCBI Nucleotide accession NC_045512.2) to determine the correct ordering and orientation, and produces an updated assembly FASTA file. DARTH performs reference-based annotation using VADR (v1.1)[72], which provides a set of genome models for all CoV RefSeq genomes[73]. VADR provides annotations of gene coordinates, polyprotein cleavage sites, and functional annotation of all proteins. DARTH supplements the VADR annotation by using Infernal[74] to scan the contigs against the SARS-CoV-2 Rfam release[75] which provides updated models of CoV 50 and 30 untranslated regions (UTRs) along with stem-loop structures associated with programmed ribosomal frame-shifts. Although VADR provides reference-based gene-calling, DARTH also provides ab initio gene-calling by using FragGeneScan (v1.31)[76], a frameshift-aware gene caller. DARTH also generates auxiliary files that are useful for assembly quality and annotation diagnostics, such as indexed BAM files created with SAMtools (v1.7)[77] representing self-alignment of the trimmed reads to the canonicalized assembly using bowtie2[51], and variant-calls using bcftools from SAMtools. DARTH generates these files so that the can be easily loaded into a genome browser such as JBrowse[78] or IGV[79]. As the final step DARTH generates a single Generic Feature Format (GFF) 3.0 file[80] containing combined set of annotation information described above, ready for use in a genome browser, or for submitting the annotation and sequence to a genome repository.

Phage assembly. Each metagenomic dataset was individually de-novo-assembled using MEGAHIT (v1.2.9)[81], and filtered to remove contigs smaller than 1 kb in size. ORFs were then predicted on all contigs using Prodigal (v2.6.3)[82] with the following parameters: -m -p meta. Predicted ORFs were initially annotated using USEARCH[54] to search all predicted ORFs against UniProt[83], UniRef90 and KEGG[84]. Sequencing coverage of each contig was calculated by mapping raw reads back to assemblies using bowtie2[51]. Terminase sequences from Al-Shayeb et al.[42] were clustered at 90% amino acid identity to reduce redundancy using CD-HIT (v4.8.1)[85], and HMM models were built with hmmbuild (from the HMMER3 suite[71]) from the resulting set. Terminases in the assemblies from Serratus were identified using hmmsearch, retaining representatives from contigs greater than 140 kb in size. Some examples of prophage and large phages that did not co-cluster with the sequences from Al-Shayeb et al. were also recovered because they were also present in a sample that contained the expected large phages. The terminases were aligned using MAFFT (v7.407)[86] and filtered by TrimAL (v1.14)[87] to remove columns comprising more than 50% gaps, or 90% gaps, or using the automatic gappyout setting to retain the most conserved residues. Maximum likelihood trees were built from the resulting alignments using IQTREE (v1.6.6)[88].

Deploying the assembly and annotation workflow. The Serratus search for known or closely related viruses identified 37,131 libraries (14,304 by nucleotide and 23,898 by amino acid) as potentially positive for CoV (score ≥ 20 and ≥10 reads). To supplement this search we also used a recently developed index of the SRA called STAT[16], which identified an additional 18,584 SRA datasets not in the defined SRA search space. The STAT BigQuery (accessed 24 June 2020) was: *WHERE tax id=11118 AND total count >1*.

We used AWS Batch to launch thousands of assemblies of NCBI accessions simultaneously. The workflow consists of four standard parts: a job queue, a job definition, a compute environment, and finally, the jobs themselves. A CloudFormation template (https://gitlab.pasteur.fr/rchikhi_pasteur/serratus-batch-assembly/-/blob/10934001/template/template.yaml) was created for building all parts of the cloud infrastructure from the command line. The job definition specifies a Docker image, and asks for 8 virtual CPUs (vCPUs, corresponding to threads) and 60 GB of memory per job, corresponding to a reasonable allocation for coronaSPAdes. The compute environment is the most involved component. We set it to run jobs on cost-effective Spot instances (optimal setting) with an additional cost-optimization strategy (SPOT_CAPACITY_OPTIMIZED setting), and allowing up to 40,000 vCPUs total. In addition, the compute environment specifies a launch template which, on each instance, (i) automatically mounts an exclusive 1 TB EBS volume, allowing sufficient disk space for several concurrent assemblies, and (ii) downloads the 5.4 GB CheckV (v0.6.0)[89] database, to avoid bloating the Docker image.

The peak AWS usage of our Batch infrastructure was around 28,000 vCPUs, performing around 3,500 assemblies simultaneously. A total of 46,861 accessions out of 55,715 were assembled in a single day. They were then analysed by two methods to detect putative CoV contigs. The first method is CheckV[89], followed selecting contigs associated to known CoV genomes. The second method is a custom script (https://gitlab.pasteur.fr/rchikhi_pasteur/serratus-batch-assembly/-/blob/10934001/stats/bgc_parse_and_extract.py) that parses coronaSPAdes BGC candidates and keeps contigs containing CoV domain(s). For each accession, we kept the set of contigs obtained by the first method (CheckV) if it is non-empty, and otherwise we kept the set of contigs from the second method (BGC).

A majority (76%) of the assemblies were discarded for one of the following reasons: (i) no CoV contigs were found by either filtering method; (ii) reads were too short to be assembled; (iii) Batch job or SRA download failed; or (iv) coronaSPAdes ran out of memory. A total of 11,120 assemblies were considered for further analysis.

The average cost of assembly was between US$0.30 and US$0.40 per library, varying depending on library type (RNA-seq versus metagenomic). This places an estimate of 46–95-fold higher cost for

assembly alone compared to a cost of US$0.0042 or US$0.0065 for an alignment-based search.

Taxonomic and phylogenetic analyses

Taxonomy prediction for coronavirus genomes. We developed a module, SerraTax, to predict taxonomy for CoV genomes and assemblies (https://github.com/ababaian/serratus/tree/master/containers/serratax). SerraTax was designed with the following requirements in mind: provide taxonomy predictions for fragmented and partial assemblies in addition to complete genomes; report best-estimate predictions balancing over-classification and under-classification (too many and too few ranks, respectively); and assign an NCBI Taxonomy Database[90] identifier (TaxID).

Assigning a best-fit TaxID was not supported by any previously published taxonomy prediction software to the best of our knowledge; this requires assignment to intermediate ranks such as sub-genus and ranks below species (commonly called strains, but these ranks are not named in the Taxonomy database), and to unclassified taxa, for example, TaxID 2724161, unclassified Buldecovirus, in cases in which the genome is predicted to fall inside a named clade but outside all named taxa within that clade.

SerraTax uses a reference database containing domain sequences with TaxIDs. This database was constructed as follows. Records annotated as CoV were downloaded from UniProt[83], and chain sequences were extracted. Each chain name, for example Helicase, was considered to be a separate domain. Chains were aligned to all complete coronavirus genomes in GenBank using UBLAST (usearch: v11.0.667)[54] to expand the repertoire of domain sequences. The reference sequences were clustered using UCLUST[54] at 97% sequence identity to reduce redundancy.

For a given query genome, ORFs are extracted using the getorf (EMBOSS:6.6.0) software[70]. ORFs are aligned to the domain references and the top 16 reference sequences for each domain are combined with the best-matching query ORF. For each domain, a multiple alignment of the top 16 matches plus query ORF is constructed on the fly by MUSCLE (v3.8.31[91]) and a neighbour-joining tree is inferred from the alignment, also using MUSCLE. Finally, a consensus prediction is derived from the placement of the ORF in the domain trees. Thus, the presence of a single domain in the assembly suffices to enable a prediction; if more domains are present they are combined into a consensus.

Taxonomic assignment by phylogenetic placement. To generate an alternate taxonomic annotation of an assembled genome, we created a pipeline based on phylogenetic placement, SerraPlace.

To perform phylogenetic placement, a reference phylogenetic tree is required. To this end, we collected 823 reference amino acid RdRP sequences, spanning all *Coronaviridae*. To this set we added an outgroup RdRP sequence from the Torovirus family (NC 007447). We clustered the sequences to 99% identity using USEARCH (ref. [54], UCLUST algorithm, v11.0.667), resulting in 546 centroid sequences. Subsequently, we performed multiple sequence alignment on the clustered sequences using MUSCLE. We then performed maximum likelihood tree inference using RAxML-NG (ref. [92], 'PROTGTR+FO+G4', v0.9.0), resulting in our reference tree.

To apply SerraPlace to a given genome, we first use HMMER (ref. [71], v3.3) to generate a reference HMM, based on the reference alignment. We then split each contig into ORFs using esl-translate, and use hmmsearch (*P* value cut-off 0.01) and seqtk (commit 7c04ce7) to identify those query ORFs that align with sufficient quality to the previously g enerated reference HMM. All ORFs that pass this test are considered valid input sequences for phylogenetic placement. This produces a set of likely placement locations on the tree, with an associated likelihood weight. We then use Gappa (v0.6.1,[93]) to assign taxonomic information to each query, using the taxonomic information for the reference sequences. Gappa assigns taxonomy by first labelling the interior nodes of the reference tree by a consensus of the taxonomic labels of all descendant leaves of that node. If 66% of leaves share the same taxonomic label up to some level, then the internal node is assigned that label. Then, the likelihood weight associated with each sequence is assigned to the labels of internal nodes of the reference tree, according to where the query was placed.

From this result, we select that taxonomic label that accumulated the highest total likelihood weight as the taxonomic label of a sequence. Note that multiple ORFs of the same genome may result in a taxonomic label, in which case, we select the longest sequence as the source of the taxonomic assignment of the genome.

Phylogenetic inference. We performed phylogenetic inferences using a custom snakemake (v6.6.0) pipeline (available at https://github.com/lczech/nidhoggr), using ParGenes (v1.1.2)[94]. ParGenes is a tree search orchestrator, combining ModelTestNG (v0.1.3)[95] and RAxML-NG, and enabling higher levels of parallelization for a given tree search.

To infer the maximum likelihood phylogenetic trees, we performed a tree search comprising 100 distinct starting trees (50 random, 50 parsimony), as well as 1,000 bootstrap searches. We used ModelTest-NG to automatically select the best evolutionary model for the given data. The pipeline also automatically produces versions of the best maximum likelihood tree annotated with Felsenstein's Bootstrap[96] support values, and Transfer Bootstrap Expectation values[97].

Reporting summary

Further information on research design is available in the Nature Research Reporting Summary linked to this paper.

Data availability

All Serratus data, raw and processed, are released into the public domain immediately in accordance with the Bermuda Principles and freely available at https://serratus.io/access. Assembled genomes for this study are available on GenBank under project PRJEB44047.

Code availability

Serratus (v0.3.0) is available at https://github.com/ababaian/serratus. Archival copies of all code and software generated for this study are freely available at https://github.com/serratus-bio. Electronic notebooks for experiments are available at https://github.com/ababaian/serratus.

51. Langmead, B. & Salzberg, S. L. Fast gapped-read alignment with Bowtie 2. *Nat. Methods* **9**, 357–359 (2012).
52. Schatz, M. C. CloudBurst: highly sensitive read mapping with MapReduce. *Bioinformatics* **25**, 1363–1369 (2009).
53. NCBI Resource Coordinators. Database resources of the National Center for Biotechnology Information. *Nucleic Acids Res.* **46**, D8–D13 (2018).
54. Edgar, R. C. Search and clustering orders of magnitude faster than BLAST. *Bioinformatics* **26**, 2460–2461 (2010).
55. Morgulis, A., Gertz, E. M., Schäffer, A. A. & Agarwala, R. A fast and symmetric DUST implementation to mask low-complexity DNA sequences. *J. Comput. Biol.* **13**, 1028–1040 (2006).
56. Shen, W., Le, S., Li, Y. & Hu, F. SeqKit: a cross-platform and ultrafast toolkit for FASTA/Q file manipulation. *PLoS ONE* **11**, e0163962 (2016).
57. Courtot, M., Gupta, D., Liyanage, I., Xu, F. & Burdett, T. BioSamples database: FAIRer samples metadata to accelerate research data management. *Nucleic Acids Res.* https://doi.org/10.1093/nar/gkab1046 (2021).
58. Levi, K., Rynge, M., Abeysinghe, E. & Edwards, R. A. Searching the Sequence Read Archive using Jetstream and Wrangler. In *Proc. Practice and Experience on Advanced Research Computing* 1–7 (Association for Computing Machinery, 2021).
59. Hunt, M. et al. IVA: accurate de novo assembly of RNA virus genomes. *Bioinformatics* **31**, 2374–2376 (2015).
60. Nurk, S., Meleshko, D., Korobeynikov, A. & Pevzner, P. A. metaSPAdes: a new versatile metagenomic assembler. *Genome Res.* **27**, 824–834 (2017).
61. Bushmanova, E., Antipov, D., Lapidus, A. & Prjibelski, A. D. rnaSPAdes: a de novo transcriptome assembler and its application to RNA-seq data. *GigaScience* **8**, giz100 (2019).
62. Antipov, D., Raiko, M., Lapidus, A. & Pevzner, P. A. metaviralSPAdes: assembly of viruses from metagenomic data. *Bioinformatics* **36**, 4126–4129 (2020).

63. Meleshko, D. et al. BiosyntheticSPAdes: reconstructing biosynthetic gene clusters from assembly graphs. *Genome Res.* **29**, 1352–1362 (2019).
64. Pfam team. *Pfam SARS-CoV-2 Special Update (part 2)* https://xfam.wordpress.com/2020/04/06/pfam-sars-cov-2-special-update-part-2/ (2020).
65. Pedersen, B. S. & Quinlan, A. R. Mosdepth: quick coverage calculation for genomes and exomes. *Bioinformatics* **34**, 867–868 (2018).
66. Edgar, R. C. UCHIME2: improved chimera prediction for amplicon sequencing. Preprint at https://doi.org/10.1101/074252 (2016).
67. Guo, J. et al. VirSorter2: a multi-classifier, expert-guided approach to detect diverse DNA and RNA viruses. *Microbiome* **9**, 37 (2021).
68. Thiel, V. et al. Mechanisms and enzymes involved in SARS coronavirus genome expression. *J. Gen. Virol.* **84**, 2305–2315 (2003).
69. Altman, T. *DARTH Coronavirus Annotation Pipeline* https://bitbucket.org/tomeraltman/DARTH/src/master/ (2020).
70. Rice, P., Longden, I. & Bleasby, A. EMBOSS: the European Molecular Biology Open Software Suite. *Trends Genet.* **16**, 276–277 (2000).
71. Eddy, S. R. Accelerated Profile HMM Searches. *PLoS Comput. Biol.* **7**, e1002195 (2011).
72. Schäffer, A. A. et al. VADR: validation and annotation of virus sequence submissions to GenBank. *BMC Bioinformatics* **21**, 211 (2020).
73. Nawrocki, E. *Coronavirus Annotation using VADR* https://github.com/nawrockie/VADR/wiki/Coronavirus-annotation#build (2020).
74. Nawrocki, E. P. & Eddy, S. R. Infernal 1.1: 100-fold faster RNA homology searches. *Bioinformatics* **29**, 2933–2935 (2013).
75. Rfam team. *Rfam Coronavirus Special Release* https://xfam.wordpress.com/2020/04/27/rfam-coronavirus-release/ (2020).
76. Rho, M., Tang, H. & Ye, Y. FragGeneScan: predicting genes in short and error-prone reads. *Nucleic Acids Res.* **38**, e191 (2010).
77. Li, H. et al. The Sequence Alignment/Map format and SAMtools. *Bioinformatics* **25**, 2078–2079 (2009).
78. Buels, R. et al. JBrowse: a dynamic web platform for genome visualization and analysis. *Genome Biol.* **17**, 66 (2016).
79. Robinson, J. T., Thorvaldsdóttir, H., Wenger, A. M., Zehir, A. & Mesirov, J. P. Variant review with the Integrative Genomics Viewer. *Cancer Res.* **77**, e31–e34 (2017).
80. Eilbeck, K. et al. The Sequence Ontology: a tool for the unification of genome annotations. *Genome Biol.* **6**, R44 (2005).
81. Li, D. et al. MEGAHIT v1.0: a fast and scalable metagenome assembler driven by advanced methodologies and community practices. *Methods* **102**, 3–11 (2016).
82. Hyatt, D., LoCascio, P. F., Hauser, L. J. & Uberbacher, E. C. Gene and translation initiation site prediction in metagenomic sequences. *Bioinformatics* **28**, 2223–2230 (2012).
83. The UniProt Consortium. UniProt: the universal protein knowledgebase in 2021. *Nucleic Acids Res.* **49**, D480–D489 (2021).
84. Altman, T., Travers, M., Kothari, A., Caspi, R. & Karp, P. D. A systematic comparison of the MetaCyc and KEGG pathway databases. *BMC Bioinformatics* **14**, 112 (2013).
85. Li, W., Fu, L., Niu, B., Wu, S. & Wooley, J. Ultrafast clustering algorithms for metagenomic sequence analysis. *Brief. Bioinformatics* **13**, 656–668 (2012).
86. Nakamura, T., Yamada, K. D., Tomii, K. & Katoh, K. Parallelization of MAFFT for large-scale multiple sequence alignments. *Bioinformatics* **34**, 2490–2492 (2018).
87. Capella-Gutiérrez, S., Silla-Martínez, J. M. & Gabaldón, T. trimAl: a tool for automated alignment trimming in large-scale phylogenetic analyses. *Bioinformatics* **25**, 1972–1973 (2009).
88. Nguyen, L.-T., Schmidt, H. A., von Haeseler, A. & Minh, B. Q. IQ-TREE: a fast and effective stochastic algorithm for estimating maximum-likelihood phylogenies. *Mol. Biol. Evol.* **32**, 268–274 (2015).
89. Nayfach, S. et al. CheckV assesses the quality and completeness of metagenome-assembled viral genomes. *Nat. Biotechnol.* **39**, 578–585 (2020).
90. Schoch, C. L. et al. NCBI Taxonomy: a comprehensive update on curation, resources and tools. *Database* **2020**, baaa062 (2020).
91. Edgar, R. C. MUSCLE: multiple sequence alignment with high accuracy and high throughput. *Nucleic Acids Res.* **32**, 1792–1797 (2004).
92. Kozlov, A. M., Darriba, D., Flouri, T., Morel, B. & Stamatakis, A. RAxML-NG: a fast, scalable, and user-friendly tool for maximum likelihood phylogenetic inference. *Bioinformatics* **35**, 4453–4455 (2019).
93. Czech, L., Barbera, P. & Stamatakis, A. Genesis and Gappa: processing, analyzing and visualizing phylogenetic (placement) data. *Bioinformatics* **36**, 3263–3265 (2020).
94. Morel, B., Kozlov, A. M. & Stamatakis, A. ParGenes: a tool for massively parallel model selection and phylogenetic tree inference on thousands of genes. *Bioinformatics* **35**, 1771–1773 (2018).
95. Darriba, D. et al. ModelTest-NG: a new and scalable tool for the selection of DNA and protein evolutionary models. *Mol. Biol. Evol.* **37**, 291–294 (2019).
96. Felsenstein, J. Confidence limits on phylogenies: an approach using the bootstrap. *Evolution* **39**, 783–791 (1985).
97. Lemoine, F. et al. Renewing Felsenstein's phylogenetic bootstrap in the era of big data. *Nature* **556**, 452–456 (2018).
98. Crisci, M. A. et al. Wide distribution of alternatively coded Lak megaphages in animal microbiomes. Preprint at https://www.biorxiv.org/content/10.1101/2021.01.08.425732v1 (2021).
99. Rapid reconstitution of the fecal microbiome after extended diet-induced changes indicates a stable gut microbiome in healthy adult dogs. *Appl. Environ. Microbiol.* **86**, e00562-20 (2020).

Acknowledgements The Serratus project is an initiative of the hackseqRNA genomics hackathon (https://www.hackseq.com). We thank the many contributors for code snippets and bioinformatic discussion (E. Erhan, J. Chu, S. Jackman, I. Birol, K. Wellman, O. Fornes, C. Xu, M. Huss, K. Ha, M. Krzywinski, E. Nawrocki, R. McLaughlin, C. Morgan-Lang, C. Blumberg and the J. Brister laboratory); A. Rodrigues, S. McMillan, V. Wu, C. Kennett, K. Chao, and N. Pereyaslavsky for AWS support; the J. Joy laboratory, G. Mordecai, J. Taylor, S. Roux, N. Kyrpides, E. Jan, T. Reddy, L. Bergner, R. Orton and D. Streicker for virology discussions; and H.-G. Drost and D. Weigel for supporting the adoption of DIAMOND v2 for Serratus protein alignments as part of an extended feature request. We are grateful to the entire team managing the NCBI SRA and the biology community for data sharing, with particular thanks to the E. Brodie, E. Lilleskov and E. Young laboratories. T.A. thanks Advanced Research Computing resource at the University of British Columbia and B.B. thanks the Max Plank Society for financial support. P.B. was financially supported by the Klaus Tschira Foundation; R.C. by ANR Transipedia, Inception and PRAIRIE grants (PIA/ANR16-CONV-0005, ANR-18-CE45-0020, ANR-19-P3IA-0001); and M.d.l.P. by the Ministerio de Economía y Competitividad of Spain and FEDER grants (BFU2017-87370-P and PID2020-116008GB-I00). A.K. and D.M. were supported by the Russian Science Foundation (grant 19-14-00172) and computation was carried out in part by Resource Centre 'Computer Centre of SPbU'. A.K. and D.M. are grateful to Saint Petersburg State University for the overall support of this work. Project support and computing resources were provided by the University of British Columbia Community Health and Wellbeing Cloud Innovation Centre, powered by AWS.

Author contributions All authors contributed equally to this work. A.B. conceived and led the study. A.B. and J.T. designed and implemented the Serratus architecture. A.B. and R.C.E. constructed the virus pangenomes and RdRP query. R.C.E. developed the SerraTax and Summarizer modules. P.B. developed the SerraPlace tree placement and taxonomy prediction code and calculated maximum likelihood trees. T.A. developed the DARTH annotation pipeline and submitted the annotated genomes to ENA. D.M. and A.K. developed the coronaSPAdes assembler. R.C. implemented the assembly pipeline, and deployed the assembly and annotation pipeline. B.B. optimized the DIAMOND algorithm for RdRP search. A.B., V.L. and D.L. designed and developed https://serratus.io and the SQL server. A.B. and G.N. developed the Tantalus R package. A.B., R.C.E., T.A., P.B., D.M., M.d.l.P., A.K. and R.C. analysed the coronavirus, RdRP and delta virus data. B.A.-S. and J.F.B. designed the phage panproteome, assembled phage genomes and conducted phage phylogenetic analyses. All authors contributed to data interpretation and writing the manuscript.

Competing interests The authors declare no competing interests.

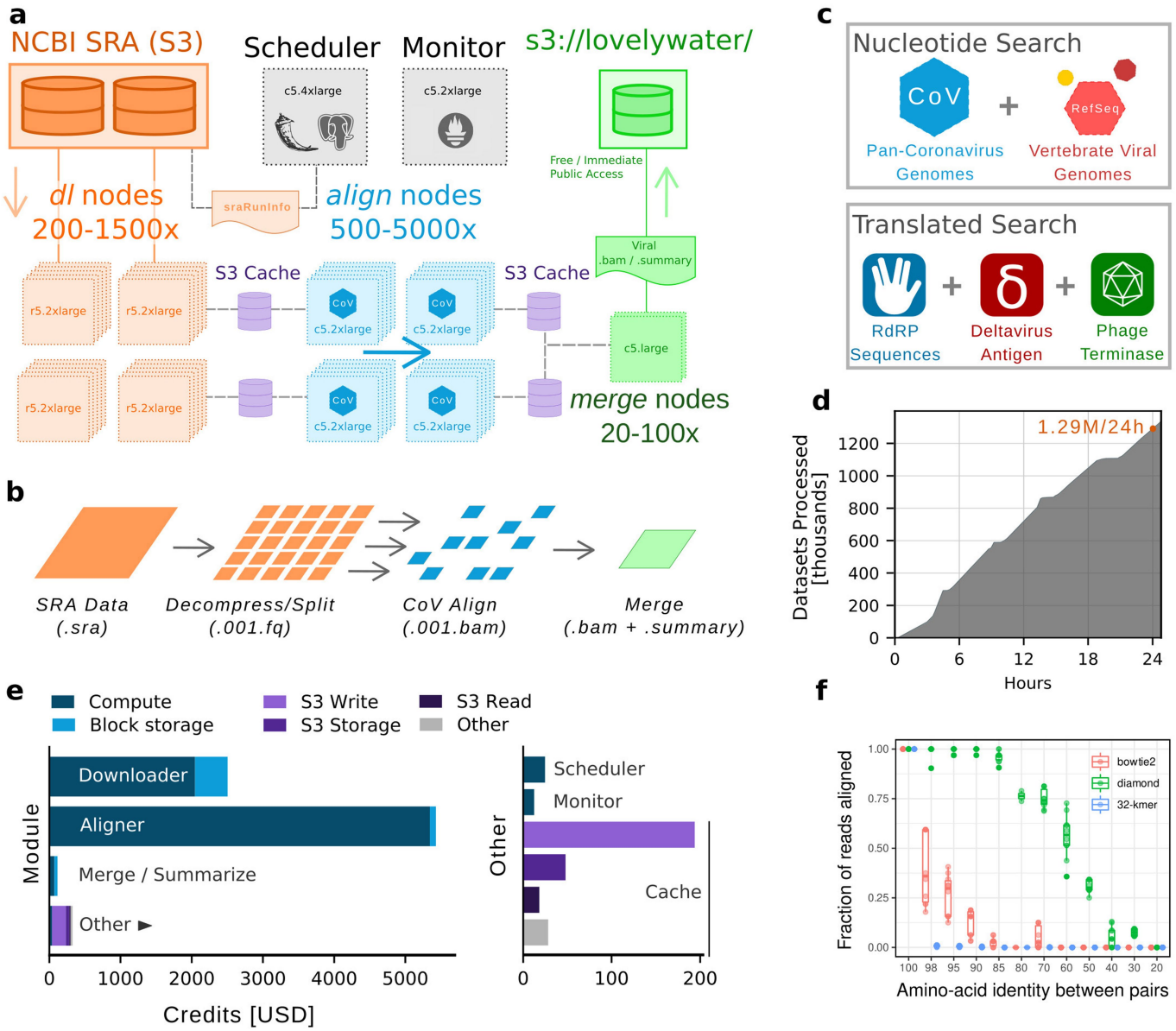
Additional information

Supplementary information The online version contains supplementary material available at https://doi.org/10.1038/s41586-021-04332-2.

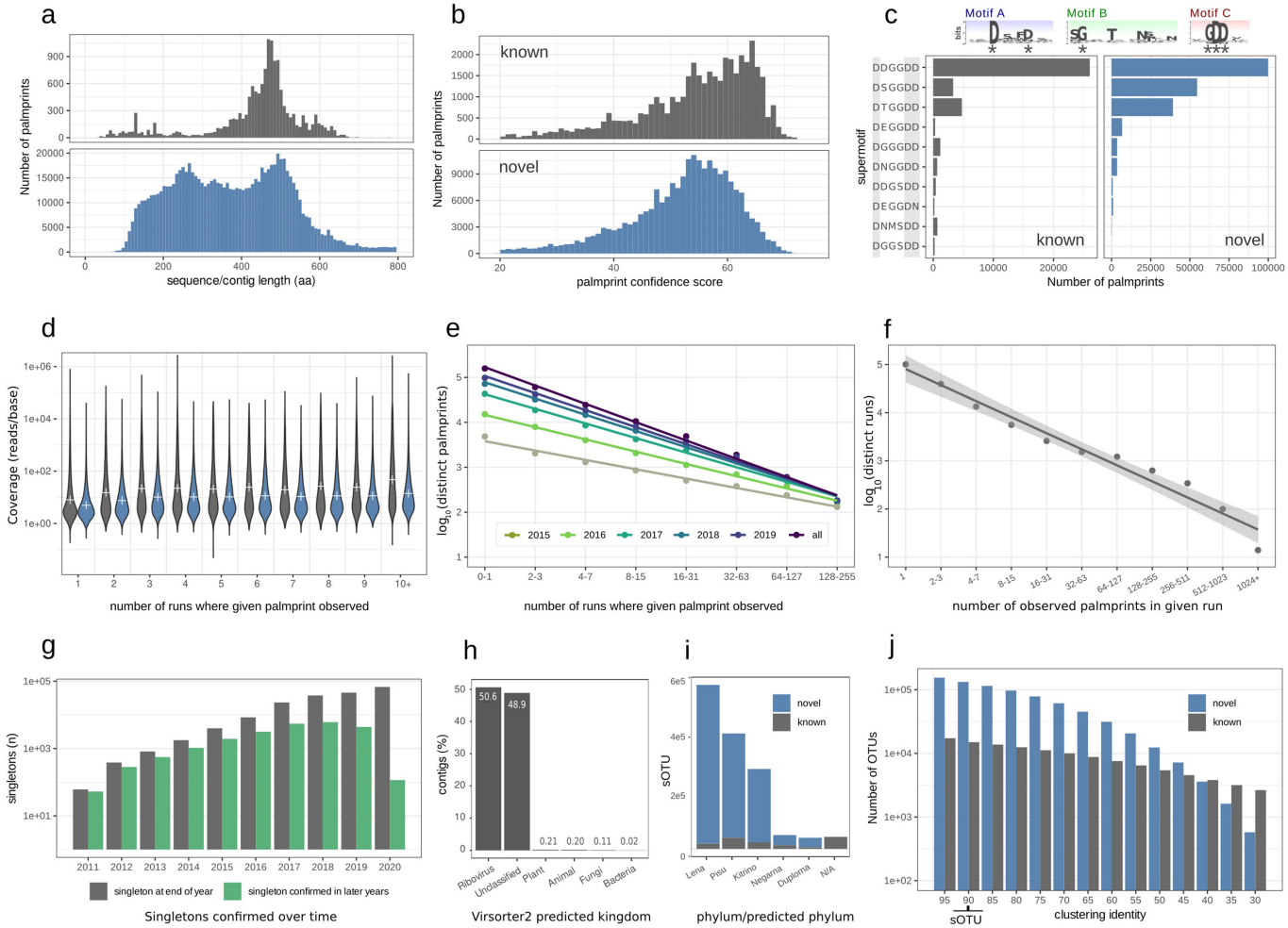
Correspondence and requests for materials should be addressed to Artem Babaian.

Peer review information *Nature* thanks C. Titus Brown, Alice McHardy and the other, anonymous, reviewer(s) for their contribution to the peer review of this work. Peer reviewer reports are available.

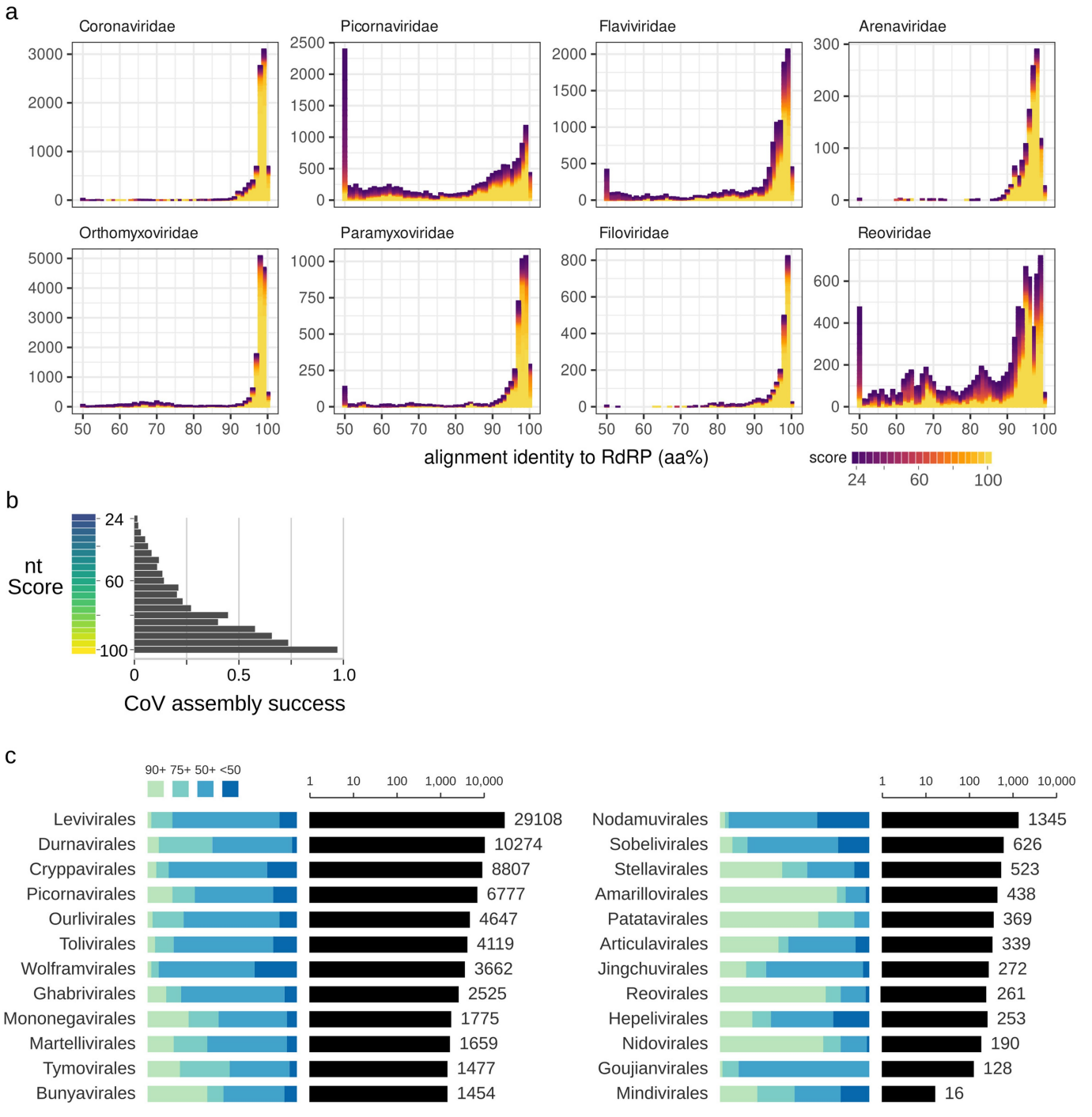
Reprints and permissions information is available at http://www.nature.com/reprints.



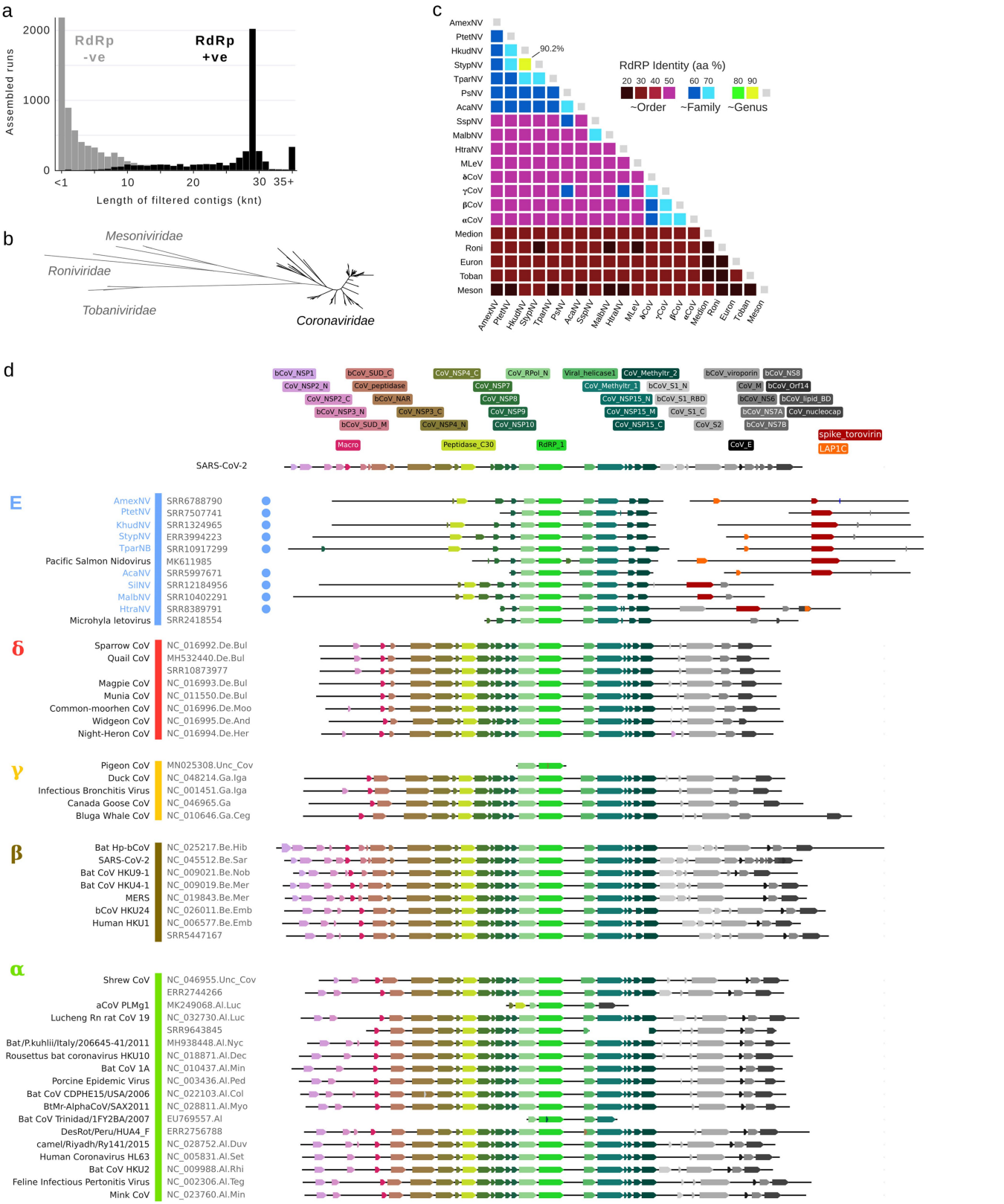
Extended Data Fig. 1 | Overview of the Serratus infrastructure. a Schematic and data workflow (**b**) as described in the methods for sequence alignment. **c** The align module accepts either a nucleotide or protein sequence query. **d** A nucleotide alignment completion rate for Serratus shows stable and linear performance to complete 1.29 million SRA accessions in a 24-hour period and the **e** cost breakdown for this run. Compute costs between modules are an approximate comparison of CPU requirements of each step. The total average cost per completed SRA accession was US$0.0062 for nucleotide search or US$0.0042 for translated-nucleotide search. **f** Tukey boxplot of biological cross-validation to measure alignment sensitivity for bowtie2 (nucleotide search), DIAMOND[14] (translated nucleotide search) or 32-mer for exact search. In brief, two RdRP sequences sharing the nominal amino acid identities form a "pair". 100 bp reads were simulated from the coding sequence of one pair and mapped onto the second pair, with the fraction of reads mapped reported. A fraction of 0.5 indicates that half the simulated reads at the given RdRP percent identity are mappable and thus detectable (see Methods). For each of the 12 percent identity categories, n = 10 biologically independent RdRP pairs were analysed.



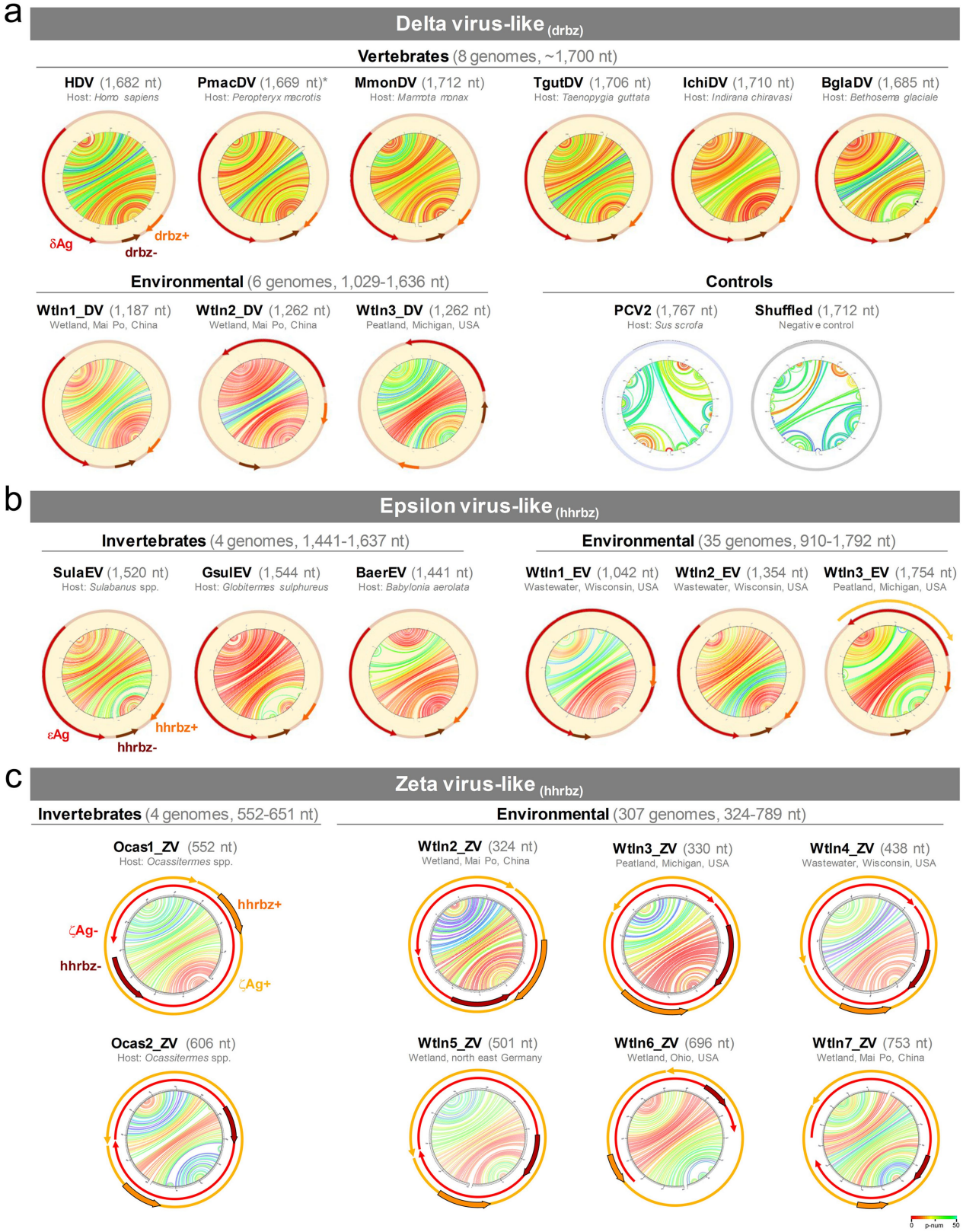
Extended Data Fig. 2 | Analysis of palmprint contigs recovered by Serratus. **a** Length distribution of amino acid sequences in the rdrp1 query (upper histogram) and microassembled contigs (lower histogram, length=nucleotides/3). **b** Distribution of Palmscan confidence scores. **c** Observations of the 10 most frequent "super-motifs" (six well-conserved residues marked with asterisk) reported by Palmscan. **d** Kernel distribution and mean (white cross) of coverage vs. abundance (number of runs where a given palmprint is observed), showing that palmprints have similar underlying coverage distributions at all abundances. **e** Preston plot of distinct palmprints vs. abundance exhibiting similar, approximately log-log-linear relationships to totals at end-of-year 2015 to 2019 and final totals at approx. end of 2020 (all). **f** Preston plot of number of distinct palmprints observed in a given run vs. number of runs with 95% confidence interval. **g** Numbers of singletons and second observations (confirmations) at the end of each year showing that the growth in singletons is matched by a comparable growth in confirmations. **h** Kingdom predicted by Virsorter2 for RdRP+ contigs (by Palmscan) obtained by full assembly of 880 randomly chosen RdRP+ runs. **i** Number of palmprints in each phylum assigned by taxonomy (known) or predicted (novel). **j** Number of OTUs as a function of clustering identity.



Extended Data Fig. 3 | Distribution of select RNA virus families. **a** Histograms of datasets matching select RNA viral family by translated-nucleotide search against RdRP query, binned by the average amino acid identity. Score (gradient colouring) function approximates pangenome/gene coverage (see methods) used for manual inspection and to prioritize assembly. Interactive and queryable versions of these plots for extended virus families are available at https://serratus.io/explorer. **b** Relationship between the nucleotide pangenome score function and the subsequent assembly success (defined by the presence of an RdRP+ contig) measured from 52,772 libraries with reads aligning to *Coronaviridae*. **c** Histogram of all detected sOTUs classified to *Riboviria* order (>40% amino acid identity to a named species) with unclassified sOTUs not shown. Segmented bars (left) show the fraction of sOTUs with similarity to known sOTU, binned into intervals 90+ (>=90%, ~species), 75+ (75% to 90%, ~genus), 50+ (50% to 75%, ~family), and <50% (~novel family). Complete multiple sequence alignments and tree files for per-order and per-family trees is available at https://serratus.io/trees.

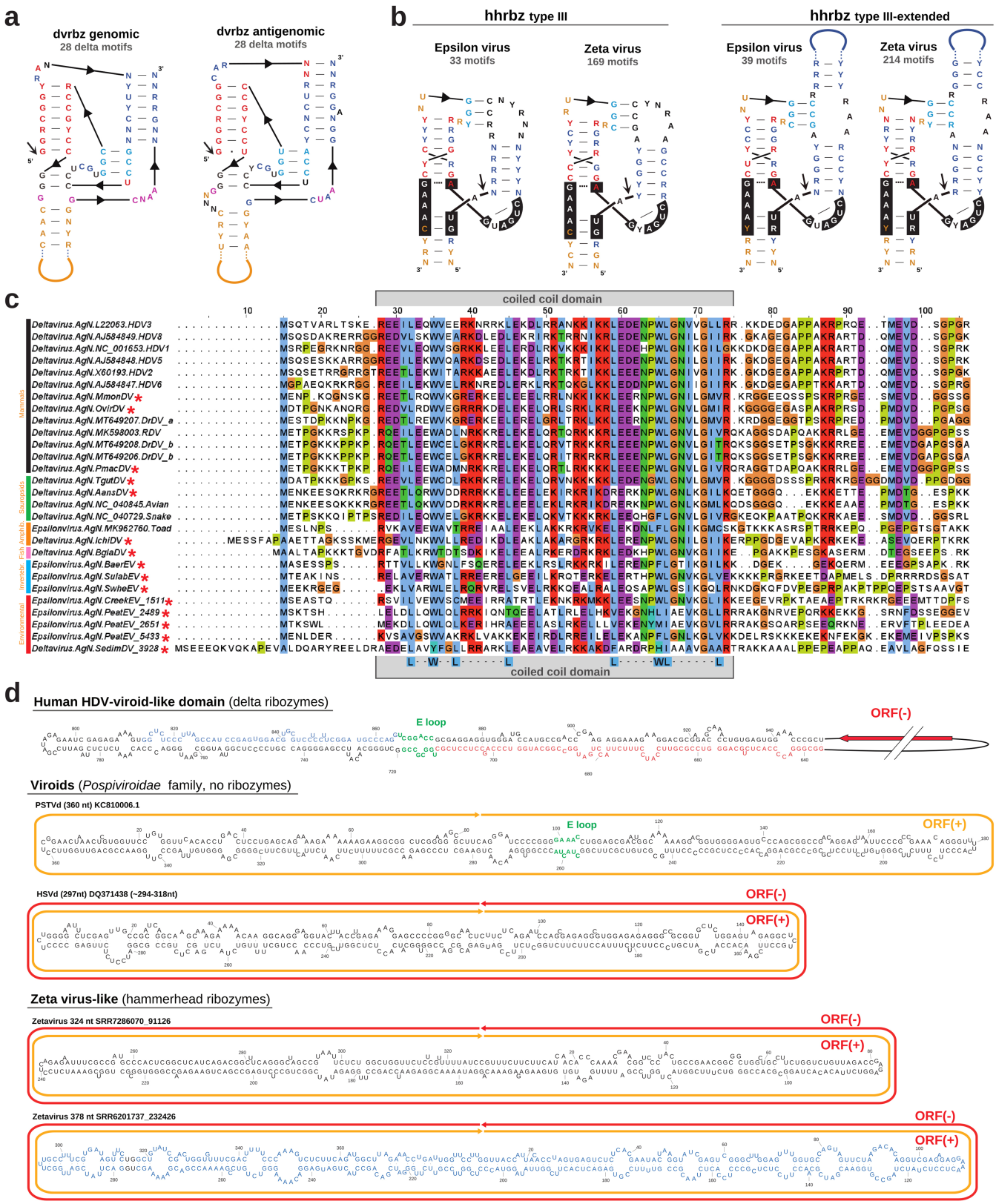


Extended Data Fig. 4 | Genome organization of *Coronaviridae* and neighbours. a Length distribution for 11,120 assembled contigs classified as CoV-positive, showing a peak around the typical CoV genome length, 4,179 (37.58%) of contigs also contained a match for RdRP. **b** Phylogram shown in Figure 3 showing the *Mesoniviridae*, *Tobaniviridae*, and *Roniviridae* outgroups. **c** Triangular matrix showing median RdRP sequence identities between selected *Nidovirales* and group-E sequences. **d** Hidden Markov Model (HMM) protein domain matches from the RdRp in exemplar sequences (contigs or GenBank sequences), grouped by genus. Novel sOTUs identified in this analysis indicated by a coloured circle.



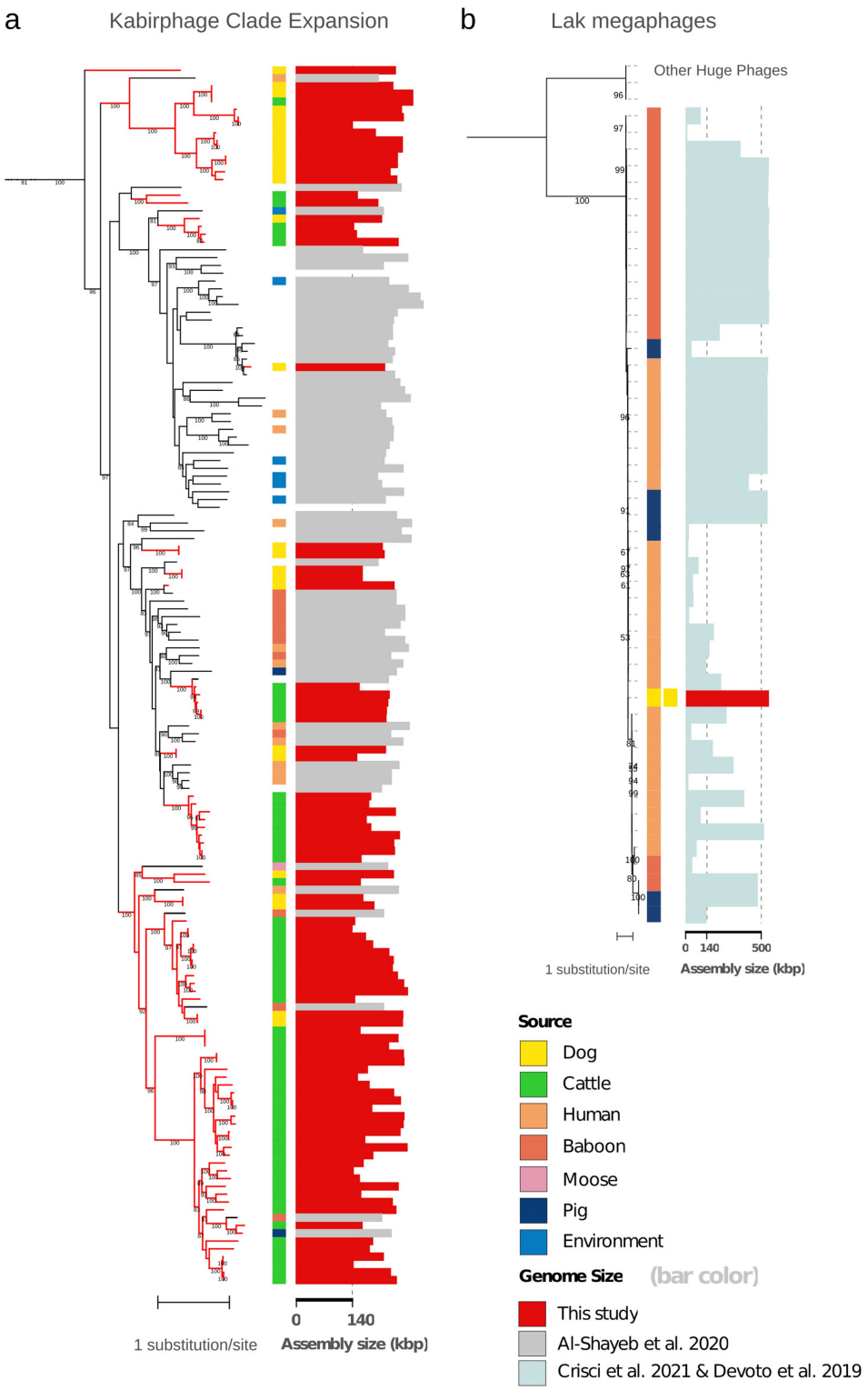
Extended Data Fig. 5 | See next page for caption.

Extended Data Fig. 5 | Newly characterized delta virus and delta-virus-like genomes. Structure and organization of selected examples from the 14 delta virus-, 39 epsilon virus- and 311 zeta virus-like genomes identified in our study. **a** Similar to human delta virus (HDV), delta virus-like genomes from vertebrates (PmacDV SRR7910143; MmonDV SRR2136906; TgutDV SRR5001850; IchiDV SRR8954566 and BglaDV SRR8242383) and environmental datasets (SRR7286070 and SRR6943136) share similar predicted stable rod-like folding, a predicted ORF coding for the delta antigen (δAg) and a delta ribozyme (dvrbz) on each polarity. Folding of the circular DNA virus Porcine Circovirus 2 (PCV2) and a shuffled MmonDV sequence are shown as negative controls. **b** Epsilon virus-like genomes detected in invertebrates (SulaEV SRR8739608; GsulEV SRR7170939 and BaerEV SRR12300397) and environmental datasets (SRR8840728 and SRR6943136) show similar structure and organization to delta viruses, with one or two predicted ORFs (epsilon antigen or Ag) and two hammerhead ribozymes (hhrbz) in equivalent genomic regions. **c** Zeta virus-like genomes detected in invertebrate (*Ocassitermes sp.* ZVs SRR8924823) and environmental datasets (SRR7286070, SRR6943136, SRR8840728, SRR6201737, SRR5864109 and SRR12063536) are smaller than delta and epsilon agents. Up to 90% of the zeta genomes have sizes multiple of 3 and predicted ORFs without stop codons, capable to encode endless tandem-repeated zeta antigens in both polarities (ζAg+ and ζAg− shown as yellow and red arrows, respectively). Both genomic zeta polarities keep hhrbzs (shown as arrows overlapping the ORFs) similar to the epsilon ribozymes (Extended Fig 6). Larger zeta virus-like genomes (>651 nt) were less abundant (7% of all zeta genomes) and frequently show stop codons, or their sizes are not multiple of 3.



Extended Data Fig. 6 | See next page for caption.

Extended Data Fig. 6 | Evolutionary history of delta-virus-like agents. **a** Consensus structures (weighted nucleotide conservation threshold of 90%) of delta virus ribozymes, including the 14 genomes described in this work. **b** Consensus structures of the two hammerhead ribozyme families (type III and extended-type III[39]) detected in epsilon and zeta agents. Most positions of epsilon and zeta motifs are sequence conserved for each ribozyme family. **c** MSA of the predicted antigen (N-term domain) from delta and epsilon agents (genomes detected in this study are indicated with a red asterisk). The antiparallel coiled-coil of the HDV is delimited with a grey box, and conserved residues involved in hydrophobic interactions are shown at the bottom[40], supporting a highly divergent connection between delta and epsilon genomes. **d** Human HDV delta virus is known to contain a viroid-like domain related to the *Pospiviroidae* family of plant viroids. Both families of agents conserve a tertiary structure reminiscent of the E-loop 5S rRNA (nucleotides in green) and are replicated by the RNA Pol II of the host[41]. *Pospiviroids*, despite lacking hhrbzs, share with zeta genomes a small rod structure, and in some cases, the presence of predicted endless tandem-repeat ORFs, most notably in both polarities of numerous variants of the Hop Stunt Viroid (HSVd). Whereas viroids have been historically regarded as non-protein-coding RNAs, our reported observations warrant further investigation.



Extended Data Fig. 7 | Huge phage and Lak phage detail. Expanded view of maximum likelihood terminase large subunit protein phylogenetic trees for (**a**) the expansion of the Kabirphage clade by newly recovered sequences from different animal types (coloured dots). Red branches are public data recovered by Serratus, black branches indicate the previously reported genomes from[42]. **b** Publicly available Lak phage genomes[98] with sequences of two newly reconstructed complete Lak megaphage genomes. These are the first reported Lak megaphages from dogs (assembled from faecal sample metagenome reads from Allaway et al.[99]). The genomes have identical terminase sequences (at the nucleotide level) although the dogs were in different housing areas and were sampled at different times (D. Allaway, personal communication).

nature portfolio

Corresponding author(s): Artem Babaian

Last updated by author(s): Nov 22, 2021

Reporting Summary

Nature Portfolio wishes to improve the reproducibility of the work that we publish. This form provides structure for consistency and transparency in reporting. For further information on Nature Portfolio policies, see our Editorial Policies and the Editorial Policy Checklist.

Statistics

For all statistical analyses, confirm that the following items are present in the figure legend, table legend, main text, or Methods section.

n/a	Confirmed	
☐	☒	The exact sample size (*n*) for each experimental group/condition, given as a discrete number and unit of measurement
☒	☐	A statement on whether measurements were taken from distinct samples or whether the same sample was measured repeatedly
☒	☐	The statistical test(s) used AND whether they are one- or two-sided *Only common tests should be described solely by name; describe more complex techniques in the Methods section.*
☒	☐	A description of all covariates tested
☒	☐	A description of any assumptions or corrections, such as tests of normality and adjustment for multiple comparisons
☐	☒	A full description of the statistical parameters including central tendency (e.g. means) or other basic estimates (e.g. regression coefficient) AND variation (e.g. standard deviation) or associated estimates of uncertainty (e.g. confidence intervals)
☒	☐	For null hypothesis testing, the test statistic (e.g. *F*, *t*, *r*) with confidence intervals, effect sizes, degrees of freedom and *P* value noted *Give P values as exact values whenever suitable.*
☒	☐	For Bayesian analysis, information on the choice of priors and Markov chain Monte Carlo settings
☒	☐	For hierarchical and complex designs, identification of the appropriate level for tests and full reporting of outcomes
☒	☐	Estimates of effect sizes (e.g. Cohen's *d*, Pearson's *r*), indicating how they were calculated

Our web collection on statistics for biologists contains articles on many of the points above.

Software and code

Policy information about availability of computer code

Data collection	Serratus (v0.3.0) is available at https://github.com/ababaian/serratus. Electronic notebooks for experiments are available at https://github.com/ababaian/serratus.
Data analysis	Archival copies of SerraTax, SerraPlace, Batch Assembly workflow, and DARTH are available at https://github.com/serratus-bio/. coronaSPADES (2020-07-15) is available at https://cab.spbu.ru/software/coronaspades. palmDB sequence database (2021-03-14) is available at https://github.com/rcedgar/palmdb and Palmscan (v1.0.0) https://github.com/rcedgar/palmscan. Software used in analysis: bcftools (v1.7), Bowtie2 (v2.4.1), BWA (v0.7.17), CD-HIT (v4.8.1), CheckV (v0.6.0), coronaSPAdes (v3.15.3), D3.js (5.16.0), DARTH (maul), DIAMOND (v2.0.8), Dustmasker (ncbi-blast:2.10.0), EPA-ng (v0.3.7), Gappa (v0.6.1), getorf (EMBOSS:6.6.0.0), Grafana (8.2.5), HMMER3 (v3.3), Infernal (v1.1.4), IQTREE (v.1.6.6), MAFFT (v.7.407), MEGAHIT (v1.2.9), ModelTest-NG (v0.1.3), MUSCLE (v3.8), nidhoggr (v0.1), Palmscan (v1.0.0), ParGenes (v1.1.2), PostgreSQL (10.14), Prodigal (v2.6.3), Prometheus (2.5.0), RAxML-NG (v0.9.0), React (16.13.1), rnaviralSPAdes (v3.15.3), SAMtools (v1.7), seqkit (v0.12.0), seqtk (github@7c04ce7), Serratus (v0.3.0), snakemake (v6.6.0), TrimAL (v1.14), UBLAST (usearch v11.0.667), UCHIME2 (usearch v8.0.1623), USEARCH (v11.0.667), VADR (1.1), Virsorter2 (v2.1)

For manuscripts utilizing custom algorithms or software that are central to the research but not yet described in published literature, software must be made available to editors and reviewers. We strongly encourage code deposition in a community repository (e.g. GitHub). See the Nature Portfolio guidelines for submitting code & software for further information.

Data

Policy information about availability of data

All manuscripts must include a data availability statement. This statement should provide the following information, where applicable:

- Accession codes, unique identifiers, or web links for publicly available datasets
- A description of any restrictions on data availability
- For clinical datasets or third party data, please ensure that the statement adheres to our policy

All Serratus data, raw and processed, is released into the public domain immediately in accordance with the Bermuda Principles and freely available at https://serratus.io/access. Assembled genomes for this study are available on GenBank under project PRJEB44047.

SRA datasets analyzed in detail: ERR2756788, ERR866585, SRR12063536, SRR12300397, SRR2136906, SRR5001850, SRR5864109, SRR6201737, SRR6943136, SRR7170939, SRR7286070, SRR7910143, SRR8242383, SRR8739608, SRR8840728, SRR8924823, SRR8954566; and the sequencing libraries in BioProjects PRJEB9357 and PRJEB34360.

Field-specific reporting

Please select the one below that is the best fit for your research. If you are not sure, read the appropriate sections before making your selection.

☐ Life sciences ☐ Behavioural & social sciences ☒ Ecological, evolutionary & environmental sciences

For a reference copy of the document with all sections, see nature.com/documents/nr-reporting-summary-flat.pdf

Ecological, evolutionary & environmental sciences study design

All studies must disclose on these points even when the disclosure is negative.

Study description	Metagenomic re-analysis of 5.7 million public sequencing datasets to uncover viral sequence diversity. Study design is consistent with a discovery oriented project.
Research sample	Data was obtained from the Sequence Read Archive (SRA); RNA-seq, metagenomic, and metatranscriptomic runs were gathered with queries as defined in Extended Table 1a. In brief, queries for Human (n = 837 694), Mouse (1 058 559), Bat (14 103), Vertebrate (114 078), Invertebrate (184 729), Eukaryotes (184 729), Prokaryotes (2 672 802), Metagenome (566 826), and Virome (52 072); Mammalian DNA (14 103) were searched for the RNA viral hallmark gene, RNA dependent RNA polymerase.
Sampling strategy	No sample-size calculations were performed. We opted to search all available/relevant data exhaustively. For the RNA virus search we limited our search to datasets derived from RNA as the starting material, or metagenomes, with the exception of the Mammalian DNA sequencing control set.
Data collection	Data was collected by running Serratus command-line and documented via a Jupyter electronic notebook by A. Babaian. Notebooks are available at https://github.com/ababaian/serratus/tree/master/notebook
Timing and spatial scale	The underlying sequencing datasets were generated and shared by the global biology community rangng from 2007-2021. This data spans all continents. Data collection from the SRA was performed between 2020-05-30 and 2020-07-11 for the nucleotide search 2021-01-11 and 2021-01-21 for the RdRP search.
Data exclusions	Data was limited to sequencing runs on the ILLUMINA platform to allow for uniform data processing. Whole Genome Sequencing data was not searched except for Chordata (bat) samples or for the Mammalian WGS control experiment. Exclusion criteria was established.
Reproducibility	The SRA Run Info tables to reproduce our search are available at https://serratus.io/access. Note: due to ongoing data migrations, some sequencing runs in the SRA may be temporarily unavailable and need to be re-attempted after a few days. Data in the SRA is archived and freely available for reproduction.
Randomization	No randomization was performed in this study and no controlling for covariants is not relevant to this study design.
Blinding	Blinding does not apply to this study as it is discovery-oriented.

Did the study involve field work? ☐ Yes ☒ No

Reporting for specific materials, systems and methods

We require information from authors about some types of materials, experimental systems and methods used in many studies. Here, indicate whether each material, system or method listed is relevant to your study. If you are not sure if a list item applies to your research, read the appropriate section before selecting a response.

Materials & experimental systems

n/a	Involved in the study
☒	☐ Antibodies
☒	☐ Eukaryotic cell lines
☒	☐ Palaeontology and archaeology
☒	☐ Animals and other organisms
☒	☐ Human research participants
☒	☐ Clinical data
☒	☐ Dual use research of concern

Methods

n/a	Involved in the study
☒	☐ ChIP-seq
☒	☐ Flow cytometry
☒	☐ MRI-based neuroimaging

Sensitive clustering of protein sequences at tree-of-life scale using DIAMOND DeepClust

Benjamin Buchfink[1], Haim Ashkenazy[2], Klaus Reuter[3], John A. Kennedy[3], Hajk-Georg Drost[1,*]

[1]Computational Biology Group, Max Planck Institute for Biology Tübingen, Germany
[2]Department of Molecular Biology, Max Planck Institute for Biology Tübingen, Germany
[3]Max Planck Computing and Data Facility, Garching, Germany

* Correspondence to:

Abstract

The biosphere genomics era is transforming life science research, but existing methods struggle to efficiently reduce the vast dimensionality of the protein universe. We present DIAMOND DeepClust, an ultra-fast cascaded clustering method optimized to cluster the 19 billion protein sequences currently defining the protein biosphere. As a result, we detect 1.7 billion clusters of which 32% hold more than one sequence. This means that 544 million clusters represent 94% of all known proteins, illustrating that clustering across the tree of life can significantly accelerate comparative studies in the Earth BioGenome era.

Main

As the global biosphere is increasingly sequenced and annotated[1,2,3,4,5], an unprecedented quality of evolutionary insights can now be harnessed to transform the life sciences, where discovery is often driven by the dimensionality reduction of massive experimental data into distinct categories (clusters) that capture common features for inference and predictive tasks. In practice, one such application is the grouping of proteins into related sequence classes that enabled recently celebrated breakthroughs such as protein structure prediction[6,7], comparative biosphere genomics[8,9], and classification within metagenomic samples[10,11,12]. These studies are early adopters of leveraging evolutionary information at scale for groundbreaking molecular and

functional applications and provide first examples of organizing the entire protein universe for downstream predictive tasks.

Recently, we introduced DIAMOND v2 to meet the user demands for scaling protein search to the tree of life[8]. With DIAMOND v2, we pledged to support the ongoing efforts of the Earth BioGenome project which aims to capture and assemble the genomes of more than 1.8 million eukaryotic species within this decade[3]. In this community quest to compare query sequences against the entire tree of life when millions of species are available, we identified the ability to cluster this vast protein sequence diversity space as a key factor currently limiting the association of sequences across large sets of divergent species.

Here, we perform a comprehensive experimental study to demonstrate that deep-clustering the protein universe of the assembled biosphere which currently consists of ~19 billion sequences is already possible today. In the Earth BioGenome era, the ability to reduce the sequence space can significantly accelerate protein comparisons when dealing with millions of species and tens of billions of sequences. For this purpose, we estimated that the Earth BioGenome consortium will generate ~27 billion protein sequences when averaging ~15,000 genes per species times ~1.8 million successfully assembled species. Current protein clustering approaches implemented in the standard tools CD-HIT[13], UClust[14], and Linclust[15] are limited when aiming to cluster billions of proteins with such broad sequence diversity in reasonable time and with sufficient clustering sensitivity at lower identity-boundaries. To overcome this limitation and provide a future-proof software solution, we implemented DIAMOND DeepClust, a cascaded clustering method leveraging sensitive protein alignments generated with DIAMOND v2[8] for incremental clustering and near-complete discovery of distant clusterable homologs at tree-of-life scale (Fig. 1)(Supplementary fig. 1-7). Using DIAMOND DeepClust, we reach this clustering milestone to sensitively cluster 19 billion sequences in 18 days on 27 high performance computing (HPC)

nodes (using 250,000 CPU hours in total). This achievement to simultaneously balance speed and sensitivity rather than having to choose between them allows us to substitute Linclust, UClust, and CD-HIT and meet the user demands of the Earth BioGenome project, where clustering sensitivity across large evolutionary distances is paramount. We further optimized our clustering procedure to be memory efficient for laptop users, but also scale linearly in a High Performance Computing (HPC) and Cloud Computing infrastructure to enable sensitive deep-clustering for a vast portfolio of applications (Methods). Finally, we designed an incremental procedure that allows users to add new sequences to a large collection of existing clusters so that the sequencing and assembly community can swiftly add incoming sequences to our biosphere cluster database without the need to re-cluster the entire dataset (Methods).

As a result of clustering ~19 billion sequences with 30% sequence identity and 90% coverage thresholds across the tree of life, we determined ~1.70 billion clusters with 32% of clusters yielding more than one element and 68% denoting singletons (only one unique sequence within each singleton cluster). While this majority of singletons suggests the presence of a large pool of putatively novel proteins (orphan polypeptides) within the protein universe (Experimental Study)(Supplementary fig. 11), these ~1.16 billion unique sequences comprise only ~6% of the full set of 19 billion sequences. The fact that 544 million clusters can capture ~94% of all known proteins illustrates the potential of deep-clustering the protein universe to accelerate protein search across the tree of life.

To put these ~19 billion sequences of our experimental study into perspective in regard to order of magnitude, we projected that clustering the ~27 billion eukaryotic protein sequences of the ~1.8 million Earth BioGenome species with DIAMOND DeepClust would yield ~2.82 billion clusters and would be feasible today on existing HPC systems (Methods). When assuming similar proportions between singletons (6% of 27 billion sequences) and non-singletons (94% of 27 billion

sequences) in this eukaryotic dataset (Supplementary fig. 8-9), we anticipate that the Earth BioGenome project will discover ~1.6 billion unique (singleton) clusters which can be investigated for their putative molecular function. The overall findings of our experimental study suggest that the protein diversity presently available across all major lineages of life can be reduced by a factor of 10 using sensitive deep-clustering and can further be compressed by a factor of 35 when disregarding singletons, or even by a factor of 60 when removing clusters of size below three. This means that 92% of the protein universe (17.8 billion sequences) can be compressed into 335 million representative sequences for downstream analyses (Supplementary fig. 10). These compression levels can additionally be improved when employing more liberal clustering criteria such as 70% coverage and no identity threshold (compared to our conservative 90% coverage and 30% identity setting). A ProtT5[16]-guided analysis of the protein sequence space mapped by our clustering suggests that it is largely composed of sequences that are mostly uncharacterized by curation efforts such as CATH[17] or Pfam[18] (Fig. 2).

Notably, the empowerment enabled by capturing protein diversity across major kingdoms of life, for example, was also demonstrated by recent breakthroughs in protein structure prediction. The predictive power of AlphaFold2 was largely derived from the use of the Big Fantastic Database[6], a public collection of diverse protein sequences containing 345 million clusters and 61 million clusters with at least three members. While currently holding the status of the largest collection of clustered protein sequences, the result of our experimental study yielding 335 million clusters with at least three members represents a 5.5-fold increase in sequence diversity compared to the Big Fantastic Database which can now be directly incorporated into protein structure prediction research. We therefore provide our 1.7 billion clusters dataset as a free and publicly accessible resource (Data availability).

Algorithmically, DIAMOND DeepClust is based on a cascaded clustering strategy engineered to gradually reduce the complexity of large datasets and to maximally exploit evolutionary conserved information (Methods). Previous methodologies to cluster protein sequences such as CD-HIT[13] and UClust[14] are more than ten years old, were not designed to scale to millions of species, and perform poorly when attempting to cluster large datasets deeper than 90% sequence identity. Although MMseqs2/Linclust[15] presented a considerable advancement over CD-HIT and UClust, it still suffers from comparatively low performance when clustering at high alignment sensitivity, thereby introducing an analytics bottleneck when attempting to scale to >27 billion estimated Earth BioGenome sequences covering the full breadth of biospheric protein space.

Source data of figure at the end of the thesis

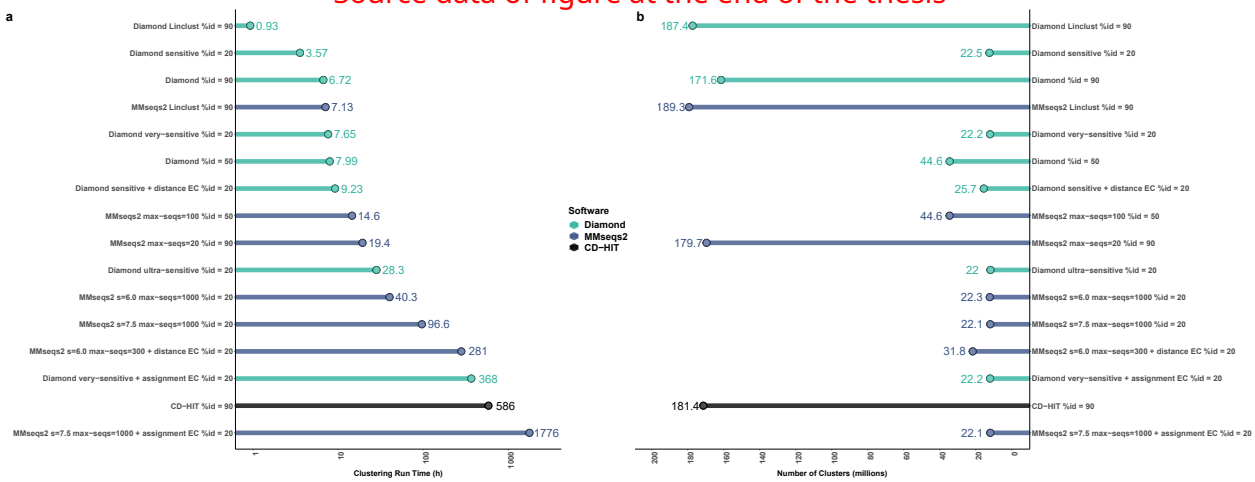


Fig. 1 | Benchmark of the clustering performance of DIAMOND DeepClust, MMseqs2 and CD-HIT using various sensitivity modes and identity thresholds. Computational benchmarks are shown for clustering the NCBI non-redundant (NR) database currently storing ~446 million protein sequences using different clustering criteria. **a,** Clustering run times for clustering the NR database on a 64-core server are shown in hours **b,** The resulting cluster counts of compressing the NCBI NR database according to the respective clustering criteria.

To formally benchmark DIAMOND DeepClust against CD-HIT and MMseqs2/Linclust, we clustered the NCBI non-redundant (NR) database containing ~446M sequences at sequence identity thresholds of 90%, 50% and 20% (Fig. 1). DIAMOND DeepClust solved this problem for deep clustering in 3.6h (sensitive mode) and 7.7h (very-sensitive mode) on a single server equipped with 64 cores compared to 1.7 days and 4 days using MMseqs2, running 11-fold and

13-fold faster respectively. In addition, DIAMOND DeepClust exhibited higher clustering quality as measured by the sensitivity of clusterable homologs found, the completeness of representation by the representative sequences, and the optimality of cluster assignment (Supplementary fig. 2-7). In particular, in the most sensitive run MMseqs2 still did not discover clusterable homologs for 9.8% of the representatives compared to 2.5% for DIAMOND ultra-sensitive (Supplementary fig. 2). Further, for MMseqs2 and DIAMOND, 4.2% of the sequences were not within clustering distance to any representative (Supplementary fig. 3), constituting information that is potentially lost during clustering, while the MMseqs2 workflow to correct such errors increased the runtime to >2 weeks compared to 9.8h using DIAMOND DeepClust (runs labeled as distance error correction), simultaneously degrading the quality and inflating the size of the clustering (cluster count). Lastly, MMseqs2 misassigned 31% of sequences (vs 23% for DIAMOND) to a representative that is not the closest to the sequence (Supplementary fig. 7), while the computation to correct such errors ran for 2.4 months compared to 16 days using DIAMOND DeepClust (runs labeled as assignment error correction). DIAMOND DeepClust ran 82-fold faster than CD-HIT, 3-fold to 8-fold faster than Linclust for clustering at 90% identity and 2-fold faster for clustering at 50% identity.

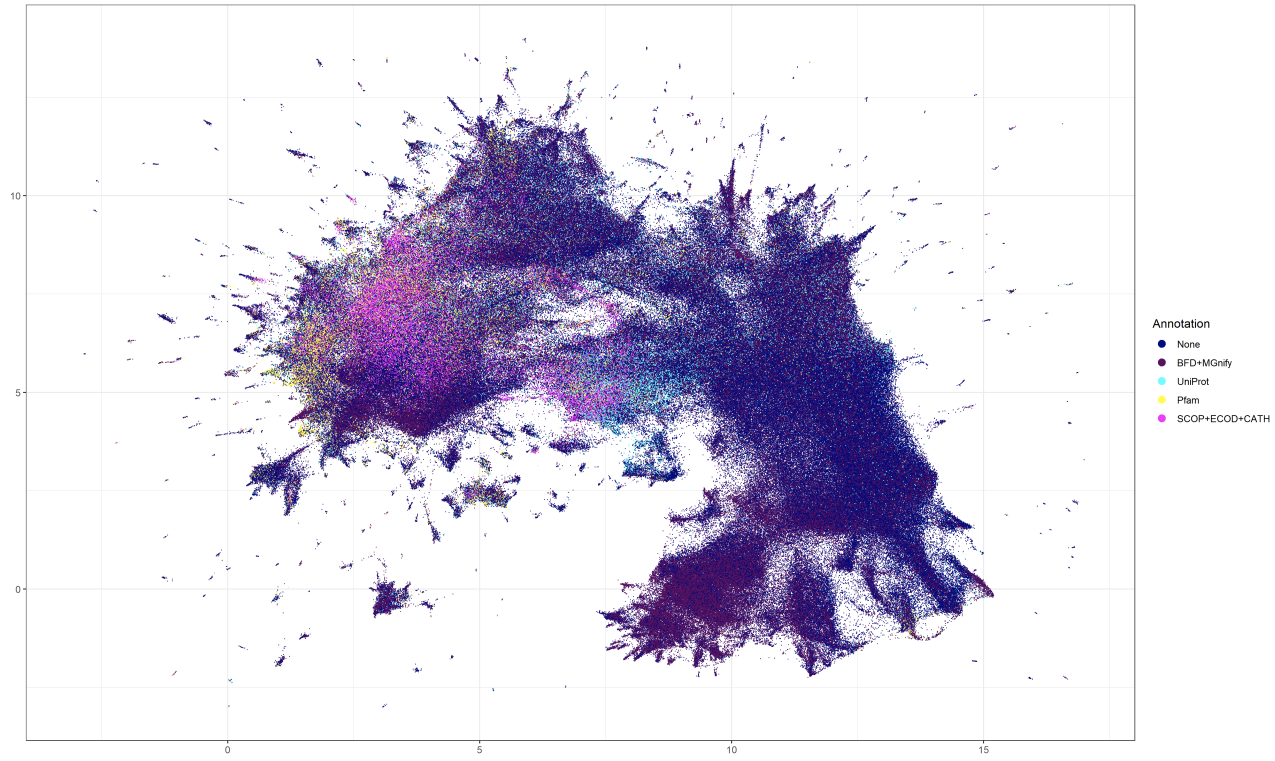


Fig. 2 | Projection of the protein universe clustered by DIAMOND DeepClust onto its protein language embedding space. UMAP projection of cluster representatives from clusters of size ≥5 after transformation into the ProtT5[16] embedding space. Each dot corresponds to a representative-sequence-embedding labeled by whether the sequence can be annotated with knowledge derived from SCOP[19]+ECOD[20]+CATH[17] or Pfam[18], or when no annotation was found whether the respective representative sequence has a homolog in UniProt, or a homolog in the BFD[6]+MGnify[21] databases. The result illustrates that the protein sequence space is dominated by unexplored protein sequences not sufficiently characterized by standard databases.

In conclusion, we designed DIAMOND DeepClust to further optimize the computational steps towards protein alignments against millions of species through dimensionality reduction (clustering) and inspire a new type of research that embraces the biodiversity of life for molecular research and subsequent prediction efforts.

Acknowledgements

The authors would like to thank Detlef Weigel for his generous sponsorship, discussions, and overall support of this work. We furthermore thank the members of the Drost and Weigel labs for extensive discussions during lab meetings and the Max Planck Computing and Data Facility for providing computational resources. We would also like to thank Jasmin Katz for dedicating her Bachelor Thesis to protein clustering and for valuable discussions in the early phases of this project. Finally, we would like to express our deep gratitude to Alexandru Tomescu, Vikram Alva, Artur Gynter, Fernando Dias, and Andreas Grigorjew, Alexandra Dallaire, Kyanna Ouyang, and Lukas Maischak for trialing early versions of DeepClust in the context of their work and for providing valuable feedback and raise constructive discussions. We thank the BMBF-funded de.NBI Cloud within the German Network for Bioinformatics Infrastructure (de.NBI) (031A532B, 031A533A, 031A533B, 031A534A, 031A535A, 031A537A, 031A537B, 031A537C, 031A537D, 031A538A) for computational support. This work was supported by the Max Planck Society.

References

1. Lewin, H. A. *et al.* Earth BioGenome Project: Sequencing life for the future of life. *Proc. Natl. Acad. Sci. U. S. A.* **115**, 4325–4333 (2018).
2. Exposito-Alonso, M., Drost, H.-G., Burbano, H. A. & Weigel, D. The Earth BioGenome project: opportunities and challenges for plant genomics and conservation. *Plant J.* **102**, 222–229 (2020).
3. Lewin, H. A. *et al.* The Earth BioGenome Project 2020: Starting the clock. *Proc. Natl. Acad. Sci. U. S. A.* **119**, (2022).
4. Blaxter, M. *et al.* Why sequence all eukaryotes? *Proc. Natl. Acad. Sci. U. S. A.* **119**, (2022).

5. Stephan, T. *et al.* Darwinian genomics and diversity in the tree of life. *Proc. Natl. Acad. Sci. U. S. A.* **119**, (2022).
6. Jumper, J. *et al.* Highly accurate protein structure prediction with AlphaFold. *Nature* 1–11 (2021).
7. Baek, M. *et al.* Accurate prediction of protein structures and interactions using a three-track neural network. *Science* (2021) doi:10.1126/science.abj8754.
8. Buchfink, B., Reuter, K. & Drost, H.-G. Sensitive protein alignments at tree-of-life scale using DIAMOND. *Nat. Methods* **18**, 366–368 (2021).
9. Vanni, C. *et al.* Unifying the known and unknown microbial coding sequence space. *Elife* **11**, (2022).
10. Chibani, C. M. *et al.* A catalogue of 1,167 genomes from the human gut archaeome. *Nat Microbiol* **7**, 48–61 (2022).
11. Coelho, L. P. *et al.* Towards the biogeography of prokaryotic genes. *Nature* (2021) doi:10.1038/s41586-021-04233-4.
12. Nayfach, S. *et al.* Metagenomic compendium of 189,680 DNA viruses from the human gut microbiome. *Nature Microbiology* 1–11 (2021).
13. Fu, L., Niu, B., Zhu, Z., Wu, S. & Li, W. CD-HIT: accelerated for clustering the next-generation sequencing data. *Bioinformatics* **28**, 3150–3152 (2012).
14. Edgar, R. C. Search and clustering orders of magnitude faster than BLAST. *Bioinformatics* **26**, 2460–2461 (2010).
15. Steinegger, M. & Söding, J. Clustering huge protein sequence sets in linear time. *Nat. Commun.* **9**, 2542 (2018).
16. Elnaggar, A. *et al.* ProtTrans: Towards Cracking the Language of Lifes Code Through Self-Supervised Deep Learning and High Performance Computing. *IEEE Trans. Pattern Anal. Mach. Intell.* **PP**, (2021).
17. Sillitoe, I. *et al.* CATH: increased structural coverage of functional space. *Nucleic Acids*

Res. (2020) doi:10.1093/nar/gkaa1079.

18. Mistry, J. *et al.* Pfam: The protein families database in 2021. *Nucleic Acids Res.* **49**, D412–D419 (2021).
19. Fox, N. K., Brenner, S. E. & Chandonia, J.-M. SCOPe: Structural Classification of Proteins--extended, integrating SCOP and ASTRAL data and classification of new structures. *Nucleic Acids Res.* **42**, D304–9 (2014).
20. Cheng, H. *et al.* ECOD: an evolutionary classification of protein domains. *PLoS Comput. Biol.* **10**, e1003926 (2014).
21. Mitchell, A. L. *et al.* MGnify: the microbiome analysis resource in 2020. *Nucleic Acids Res.* (2019) doi:10.1093/nar/gkz1035.
22. Li, W. & Godzik, A. Cd-hit: a fast program for clustering and comparing large sets of protein or nucleotide sequences. *Bioinformatics* **22**, 1658–1659 (2006).
23. Steinegger, M. & Söding, J. MMseqs2 enables sensitive protein sequence searching for the analysis of massive data sets. *Nat. Biotechnol.* **35**, 1026–1028 (2017).
24. Hauser, M., Steinegger, M. & Söding, J. MMseqs software suite for fast and deep clustering and searching of large protein sequence sets. *Bioinformatics* **32**, 1323–1330 (2016).
25. Roberts, M., Hayes, W., Hunt, B. R., Mount, S. M. & Yorke, J. A. Reducing storage requirements for biological sequence comparison. *Bioinformatics* **20**, 3363–3369 (2004).
26. Rognes, T. Faster Smith-Waterman database searches with inter-sequence SIMD parallelisation. *BMC Bioinformatics* **12**, 221 (2011).
27. Buchfink, B., Xie, C. & Huson, D. H. Fast and sensitive protein alignment using DIAMOND. *Nat. Methods* **12**, 59–60 (2015).
28. Frith, M. C. A new repeat-masking method enables specific detection of homologous sequences. *Nucleic Acids Res.* **39**, e23 (2011).
29. Clopper, C. J. & Pearson, E. S. THE USE OF CONFIDENCE OR FIDUCIAL LIMITS ILLUSTRATED IN THE CASE OF THE BINOMIAL. *Biometrika* **26**, 404–413 (1934).

30. Tautz, D. & Domazet-Lošo, T. The evolutionary origin of orphan genes. *Nat. Rev. Genet.* **12**, 692–702 (2011).
31. McInnes, L., Healy, J. & Melville, J. UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction. *arXiv [stat.ML]* (2018).
32. Eddy, S. R. A new generation of homology search tools based on probabilistic inference. *Genome Inform.* **23**, 205–211 (2009).
33. Chen, I.-M. A. *et al.* The IMG/M data management and analysis system v.7: content updates and new features. *Nucleic Acids Res.* **51**, D723–D732 (2023).
34. Steinegger, M., Mirdita, M. & Söding, J. Protein-level assembly increases protein sequence recovery from metagenomic samples manyfold. *Nat. Methods* **16**, 603–606 (2019).
35. Sayers, E. W. *et al.* Database resources of the national center for biotechnology information. *Nucleic Acids Res.* **50**, D20–D26 (2022).
36. Vanni, C. *et al.* AGNOSTOS-DB: a resource to unlock the uncharted regions of the coding sequence space. *bioRxiv* 2021.06.07.447314 (2021) doi:10.1101/2021.06.07.447314.
37. Levy Karin, E., Mirdita, M. & Söding, J. MetaEuk-sensitive, high-throughput gene discovery, and annotation for large-scale eukaryotic metagenomics. *Microbiome* **8**, 48 (2020).
38. Delmont, T. O. *et al.* Functional repertoire convergence of distantly related eukaryotic plankton lineages revealed by genome-resolved metagenomics. *bioRxiv* 2020.10.15.341214 (2021) doi:10.1101/2020.10.15.341214.
39. Alexander, H. *et al.* Eukaryotic TOPAZ MAGs. Preprint at https://doi.org/10.17605/OSF.IO/C9HJ5 (2022).
40. Camarillo-Guerrero, L. F., Almeida, A., Rangel-Pineros, G., Finn, R. D. & Lawley, T. D. Massive expansion of human gut bacteriophage diversity. *Cell* **184**, 1098–1109.e9 (2021).
41. del Río, Á. R. *et al.* Functional and evolutionary significance of unknown genes from uncultivated taxa. *bioRxiv* 2022.01.26.477801 (2022) doi:10.1101/2022.01.26.477801.

Methods

Algorithmic overview of DIAMOND DeepClust

Representative-based clustering

Following an analogous strategy as the gold standard approaches implemented in CD-HIT[22] and UCLUST[14], we define a clustering of an input dataset of protein sequences as a subset of representative sequences such that any input sequence lies within a user-defined distance threshold of at least one representative sequence. As a result, each input sequence will be assigned to one particular representative sequence. This threshold setting (identity, coverage, and e-value) is also referred to as the clustering criterion. For the purpose of this study, in addition to a basic e-value threshold of 0.00001 with respect to the size of the input database, we require that a pairwise local alignment between sequences satisfy a specific minimum sequence identity and length coverage of the co-clustered (non-representative) sequences. Analogous to MMseqs2, we compute the approximate sequence identity instead of the BLAST-like identity defined as the fraction of match columns in the pairwise alignment, as this allows us to save time for backtracing of alignments (Supplementary Information). The approximate identity is derived from the alignment score and the lengths of the aligned ranges in the sequences as a linear regression and can be considered a better measure of evolutionary distance than the actual sequence identity[23]. Starting from a set of alignments that meet the user-specified or default clustering criterion, we compute a set of representative sequences by first encoding the alignments as a directed graph G where nodes represent individual protein sequences and edges denote pairwise local alignments between them, whereby a directed edge from sequence A to sequence B indicates that A can represent B according to the clustering criterion. In a second step, we apply the greedy vertex cover algorithm on the alignment-graph G to determine a near-minimal covering set of graph vertices. The algorithm repeatedly selects the vertex with the highest node outdegree and removes it from the graph along with its out-neighbors to form a new cluster until the graph is completely clustered. In the final round of cascaded clustering, we also permit recursive merges of clusters to prevent clusterable pairs to remain in the final clustering. We also implemented simple length-sorted clustering but observed better clustering quality for the greedy vertex cover approach (data not shown).

Cascaded clustering

As exhaustive all-vs-all alignment of protein datasets consisting of hundreds of millions of sequences is prohibitively expensive, we approach this issue by adopting cascaded clustering[24] to gradually construct larger sequence clusters in several rounds of comparison with increasing alignment sensitivity (iterating between modes: --fast, *default*, --sensitive, --very-sensitive, --ultra-

sensitive). In the first round, we subsample the seed space using minimizers[25] with a window size of 12 which we empirically found to provide a good balance between speed and sensitivity, and attempt to achieve linear computational scaling of comparisons by considering only seed hits against the longest sequence for identical seeds rather than trialing all possible combinations[15]. This heuristic is sufficient to find meaningful representatives as the seeds at this stage are selected to be highly specific and the longest sequence is a priori the most likely to maximize recruitment of member sequences due to the unidirectional length coverage criterion. We compute representative sequences from the resulting alignments by greedy vertex cover[24], which are then passed on to the next round of cascaded clustering and subjected to an all-vs-all DIAMOND v2 blastp search at increased sensitivity. Depending on the desired clustering depth, two to six of these alignment rounds are chained until reaching sufficient sensitivity such that most representatives within clustering distance have been discovered. We optimized self-alignment of the representative databases by taking advantage of the symmetry of queries and targets in the seeding stage, avoiding the evaluation of redundant seed hits, and thus doubling the performance of this computation. This is accordingly taken into account when the database is processed in blocks by eliminating redundant block combinations.

Distance error correction

Distance errors in the clustering are introduced by sequences that do not fall within the clustering distance of their assigned representative and arise due to the recursive merging of clusters in the cascaded clustering workflow based on alignments of only the representative sequences. These errors do not necessarily present an error in the biological sense, since biological properties of the sequences such as ancestry, structure and function can be conserved despite the fact that the local alignment does not satisfy a certain threshold requirement. Nevertheless, we implemented an additional workflow to optionally correct such errors. We first align all sequences against their assigned representatives to find the sequences failing the clustering criterion.
After the identification of all putatively mis-clustered sequences, we re-align the set of these sequences against the database of all representative sequences using DIAMOND v2 in iterated blastp search mode with increasing sensitivity. If an alignment against a representative sequence is detected, the sequence is reassigned to the cluster of that representative. If multiple representatives satisfy the clustering criterion, the e-value of the local alignment determines the assignment. We collect all sequences that fail to align against any representative, remove them from the clustering and re-cluster this dataset with the cascaded clustering workflow. The resulting sub-clustering is again subjected to the distance error correction workflow, an iterative procedure that continues until convergence to a clustering with no distance errors.

Assignment error correction

For any given clustering a sequence may lie in clustering distance of multiple representative sequences. Cascaded clustering or incremental clustering as performed by tools like CD-HIT or UClust hold no guarantee of assigning a sequence to the cluster of the closest representative, measured by a metric such as the e-value of the local alignment. Although this property of assignment error has no impact when users wish to work only with the set of representative

sequences in the context of dimensionality reduction, it becomes a relevant drawback when attempting to use all cluster members for downstream analyses such as multiple sequence alignments or gene family characterization. To this end, we have implemented a reassignment workflow that will search all non-representative sequences against the representative database and assign each sequence to the closest representative as measured by the e-value (if the e-value is 0 for different representatives the bitscore determines the representative assignment), while maintaining the clustering criterion.

Many-core parallelization of clustering tasks

Analogous to our computational scaling efforts introduced in DIAMOND v2, we elevated our clustering capabilities to run massively parallel on High-Performance-Computing (HPC) and Cloud Computing infrastructures. To accommodate servers with 128 or more compute cores, we have refactored our multi-threading code to fix existing load imbalances during the alignment workflow and allow the software to scale smoothly to 256 threads or more. Optimal scaling of the seed extension stage in DIAMOND v2 is impeded by query proteins that attract a disproportionate number of target hits or incur an unusual cost of Smith Waterman extensions based on the length of their respective sequence, due to the use of static load balancing that is only able to distribute different query sequences among threads. We addressed this issue in DIAMOND DeepClust by implementing a fine granular task-based parallelism in which individual threads that are processing expensive queries can make use of a work-stealing task scheduler to redistribute extension tasks among the thread pool.

Gapped alignment computation

We produced a novel vectorized Smith Waterman implementation based on a modified SWIPE[26] approach that was originally developed for the first DIAMOND version[27], but dropped out of its code base soon after the initial release. While SWIPE vectorized the Smith Waterman algorithm by computing alignments of the same query against multiple targets, we generalize this approach to computing alignments of multiple independent query/target pairs. This is accomplished by using score profiles for the queries that store alignment scores along the sequence for each of the amino acid residues. For computing one column of the dynamic programming matrix, we maintain pointers into these profiles for each SWIPE channel and apply an AVX2-optimized matrix transposition to interleave the query/target scores for each dynamic programming cell into the same register, then compute the cell updates according to the standard SWIPE logic. Contrary to the original SWIPE design, this approach permits the computation of banded and anchored alignments, which in turn also enable optimizations such as the cheap determination of alignment start and end coordinates as well as X-drop termination. Compared to our implementation of the original SWIPE algorithm, we have measured ~20% computational overhead for this approach on the Intel Ice Lake architecture.

Ungapped alignment heuristics

When clustering highly similar sequences at >90% identity, most ungapped segment pairs that make up their alignments can already be found during the seeding stage. We exploit this by clustering sequences without full Smith Waterman extension if one of the ungapped alignments emerging from the seed hits already satisfies the clustering criterion. Conversely, we exclude a target from gapped extension if the sum of identities or sequence coverage of these ungapped segment pairs fails a relaxed clustering criterion based on empirically derived thresholds. The first heuristic for accepting alignments without full extension was also used for the clustering runs at 50% sequence identity. Together, this ungapped alignment strategy allows to reduce the computational burden when dealing with highly similar sequences without the loss of clustering sensitivity.

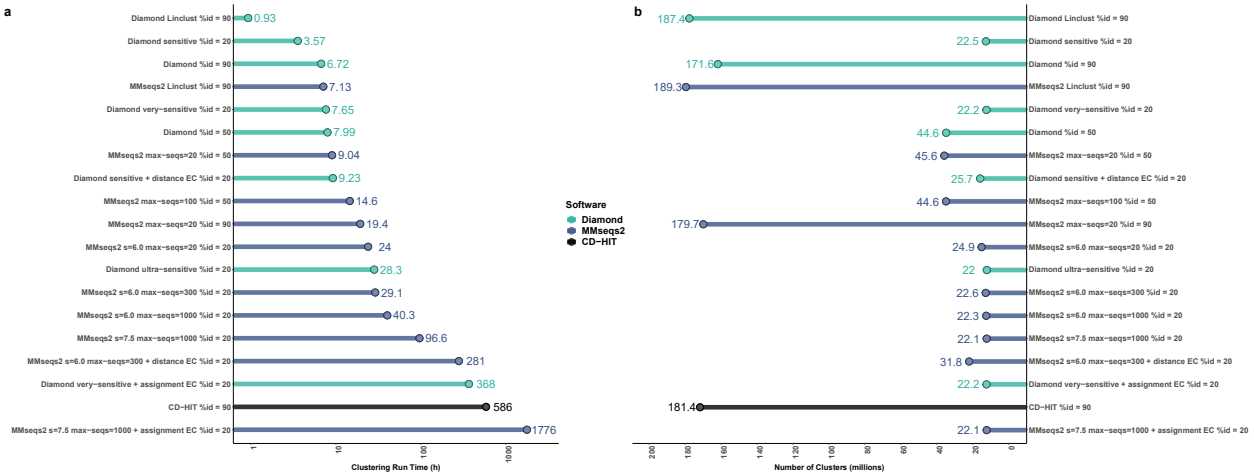
Memory optimization

Next to speed, sensitivity and user-friendliness, feedback from DIAMOND users identified memory efficiency of the search procedure as one the main advantages compared to alternative aligners. This memory optimization feature of DIAMOND allows users to perform large scale searches on their laptops and scale their parallelization efforts seamlessly into an HPC or cloud infrastructure through its distributed memory and parallelization library. Our aim for DIAMOND DeepClust was therefore to continue this memory efficiency streak by designing a memory efficient cascaded clustering to unlock the clustering of large input datasets on a laptop or massively parallel virtual machines when aiming to scale distributed computing in the cloud. In detail, the double indexing approach with runtime-generated and partitioned indexes allows the aligner to operate memory-efficiently without the need to store large index data structures on disk or maintain them in memory. For clustering large datasets with limited memory, DIAMOND will automatically use an incremental procedure on a length-sorted and partitioned database as described under Experimental Study/Clustering, which effectively limits both the use of temporary storage space and main memory to a user-defined maximum (command line option -M). For clustering on HPC systems, the design of the cascaded clustering algorithm as chained rounds of all-vs-all alignments at increasing sensitivity also allows decomposition of this computation into arbitrarily many work packages that can be processed independently on a distributed infrastructure. This process can be automated using the multiprocessing feature introduced in DIAMOND v2[8].

Cluster extension

Sensitively clustering billions of sequences across the tree of life is a computationally heavy task. However, this procedure has to be performed only once and the resulting cluster database can be made publicly available (Data availability). To accommodate the future growth of sequenced species within this decade, we designed a cluster extension workflow to allow users to add new query sequences to existing clusters to extend the initial cluster database without the need of re-clustering all sequences together. In particular, new sequences can be searched against the existing representative set using DIAMOND in iterative search mode (option --iterate). This mode identifies all sequences that can be assigned to an existing cluster. The remaining unaligned sequences can then be clustered independently, and the resulting representatives can be added to the existing clustering.

Benchmarks



Supplementary fig. 1 | Benchmark of the clustering performance of DIAMOND DeepClust, MMseqs2 and CD-HIT using a broader variety of max-seqs parameters. To cover a broader range of parameter settings, we benchmarked various alternative settings of the MMseqs2 max-seqs parameter against DIAMOND DeepClust and CD-HIT. **a,** Run times for clustering the NCBI NR database on a 64-core server are shown in hours **b,** An illustration of the cluster counts resulting from clustering the NCBI NR database with the respective clustering criteria.

Design

Our clustering benchmark is based on the NCBI NR database downloaded in November 2022, containing 513,991,389 sequences and 200,929,118,620 total residues. We hard-masked this database using tantan[28] with default settings and removed all sequences that were masked over >10% of their range, resulting in a reduced database of 445,610,930 sequences. This choice is motivated by the fact that stringent filtering of false positives is important for deep clustering, which is normally handled by DIAMOND and MMseqs2 by applying soft-masking and composition-

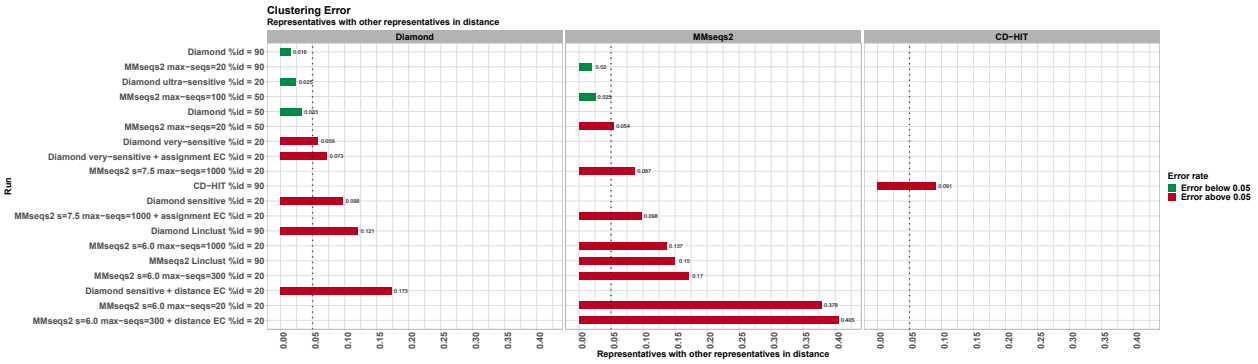
based score correction. For the purpose of our benchmark, we did not want to rely on the tools' internal masking and score correction procedures however, as this would impair the comparability of results. We therefore disabled these features for our benchmarking runs and instead relied on the precomputed hard masking. Removing sequences that are masked over a substantial part of their range is necessary for this design since otherwise they would remain as unclusterable singletons throughout the computation due to the coverage criterion, needlessly inflating the runtime. For each of the benchmarked tools, we report the wall clock time for clustering this database and the resulting number of clusters. We conducted separate runs with a clustering criterion of 90% and 50% sequence identity, as well as no restriction of the sequence identity, while setting a coverage cutoff of 80% of the co-clustered sequence for all runs. The deep clustering runs without restriction of the sequence identity were executed three times using the --sensitive, --very-sensitive and --ultra-sensitive modes of DIAMOND, as well as the -s6.0 and -s7.5 sensitivity settings of MMseqs2, which we chose to roughly correspond to the first two DIAMOND modes[8]. The --max-seqs parameter of MMseqs2 needs to be manually set by the user, so we decided to try a set of possible values whereby further increases were limited by the amount of disk space available on the benchmark system (Supplementary fig. 1). We selected the run that was most comparable to the corresponding DIAMOND run based on the sensitivity error metric (Supplementary fig. 2) for creating (Fig. 1). We limited the evaluation of CD-HIT to clustering at 90% identity since deeper clusterings at lower sequence identity levels cannot be computed with this tool in practical time. For DIAMOND and MMseqs2, we conducted additional runs labeled as *distance error correction*, designed to correct errors where sequences do not satisfy the clustering criterion against their assigned representative. These runs correspond to the recluster workflow in DIAMOND and the --cluster-reassign option of MMseqs2. For DIAMOND and MMseqs2, we conducted additional runs labeled as *assignment error correction*, designed to reassign each non-representative sequence to the cluster of the closest representative (as measured by the e-value of the local alignment) that satisfies the clustering criterion. The actual computations of the reassignment runs were distributed on a compute cluster and the runtimes converted to the equivalent of a single 64-core server. The benchmarks for clustering at 90% identity were run based on an older version of the NR database downloaded in September 2021 containing 425,032,034 protein sequences and 155,806,124,097 total residues. The database was not hardmasked and no sequences were excluded.

Environment

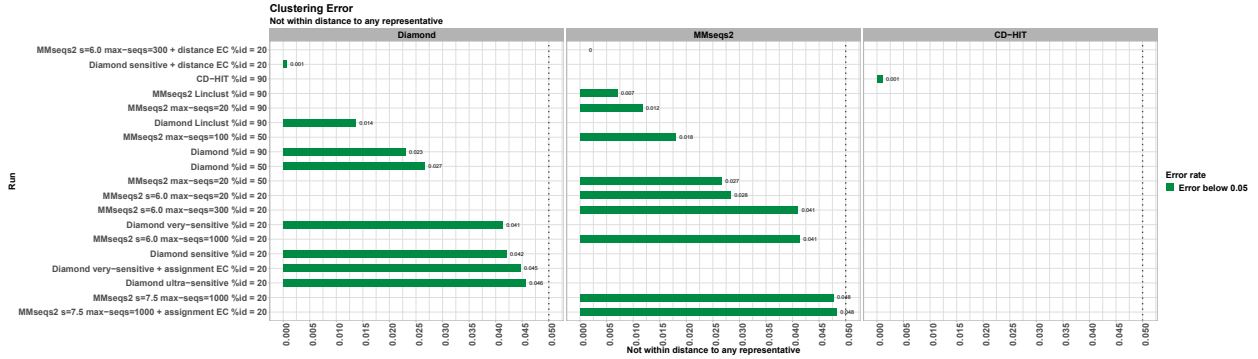
All runs were conducted on a pair of 64-core dedicated virtual cloud nodes with 1 TB RAM and a 2 TB SSD on the HPC Cloud at the Max Planck Computing and Data Facility in Garching. The hypervisors used were dual Intel IceLake-based (Xeon Platinum 8360Y @ 2.4 GHz) compute nodes with a total of 72 cores, 2 TB of RAM and 10 TB SSD storage in RAID 6 configuration. The HPC Cloud is based on OpenStack and CEPH and offers standard cloud computing "building blocks", including virtual machines based on common Linux operating systems, software-defined networks, routers, firewalls, and load balancers, as well as integrated block and S3-compatible object storage services. Analogous infrastructures are currently employed by centralized commercial cloud computing providers such as Google Cloud, Amazon Web Services, and

Microsoft Azure, thereby allowing native adoption of DIAMOND DeepClust on these systems as well.

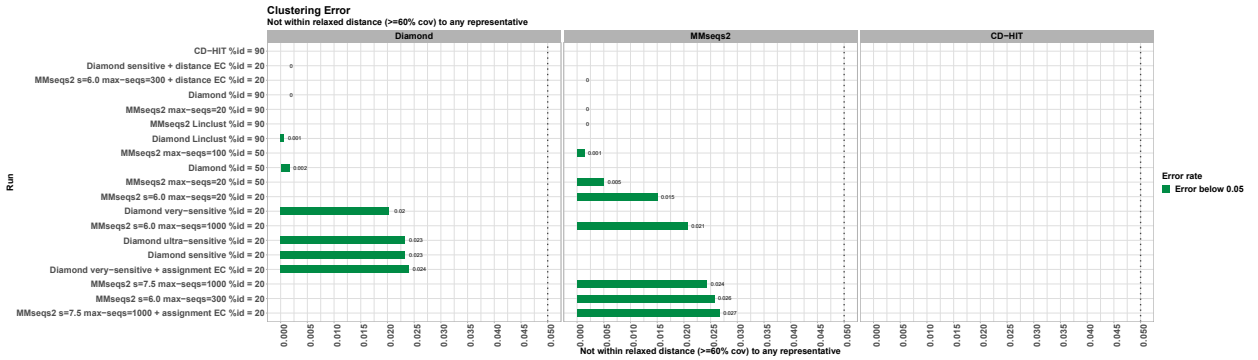
Source data of figures at the end of the thesis



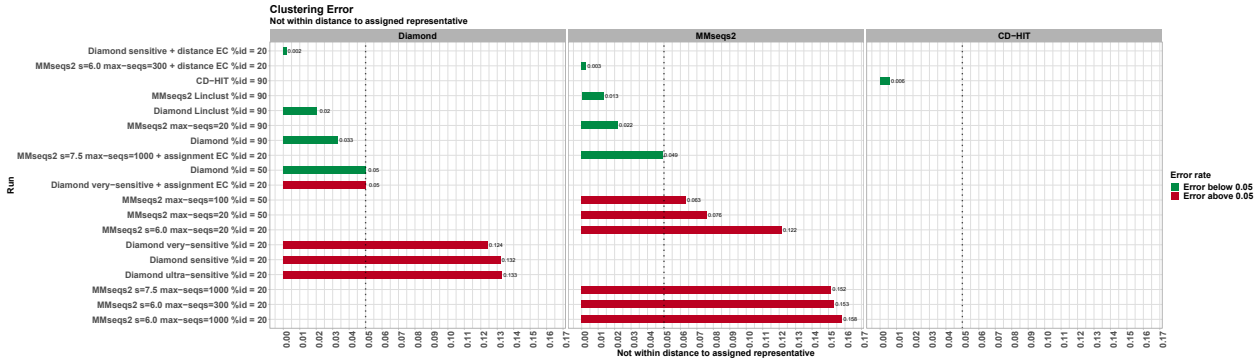
Supplementary fig. 2 | Benchmark of the sensitivity errors with DIAMOND DeepClust and MMseqs2 using various clustering criteria. Shown is the fraction of cluster representative sequences that satisfy the clustering criterion against another representative. Error rates below 0.05 indicate that most representatives are unique in the clustered set and do not correspond to other representatives.



Supplementary fig. 3 | Benchmark of distance errors when clustering with DIAMOND DeepClust and MMseqs2 using various clustering criteria. Shown is the fraction of cluster member sequences that do not satisfy the clustering criterion against any representative sequence.

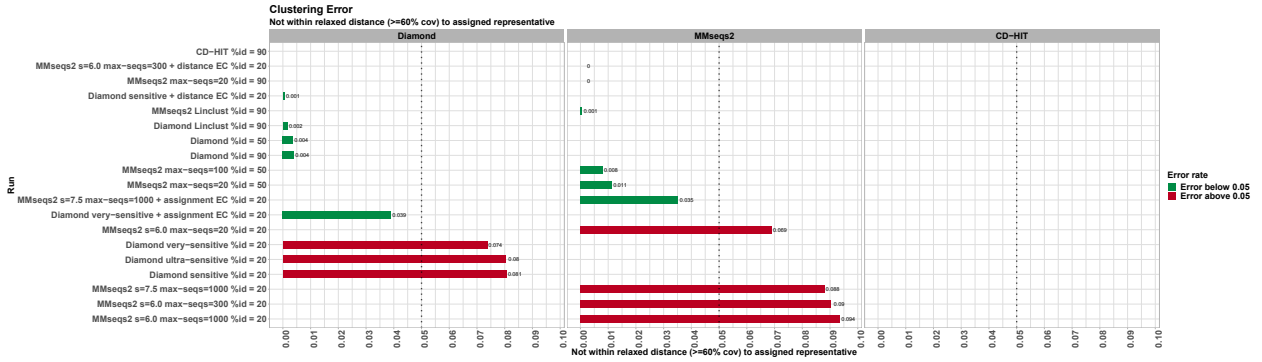


Supplementary fig. 4 | Benchmark of distance errors (relaxed) when clustering with DIAMOND DeepClust and MMseqs2 using various clustering criteria. Shown is the fraction of cluster member sequences that do not satisfy a relaxed clustering criterion against any representative sequence, defined as 60% coverage and a sequence identity threshold lowered by 10 percentage points.



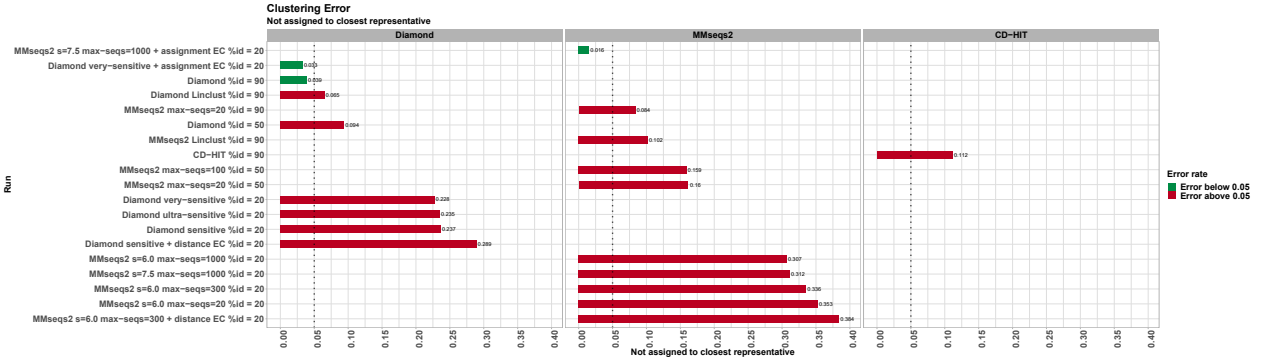
488

489 **Supplementary fig. 5 | Benchmark of distance errors when clustering with DIAMOND DeepClust and**
490 **MMseqs2 using various clustering criteria.** Shown is the fraction of cluster member sequences that do
491 not satisfy the clustering criterion against their assigned representative sequence.



492

493 **Supplementary fig. 6 | Benchmarking of distance errors (relaxed) when clustering with DIAMOND**
494 **DeepClust and MMseqs2 using various clustering criteria.** Shown is the fraction of cluster member
495 sequences that do not satisfy a relaxed clustering criterion against their assigned representative sequence,
496 defined as 60% coverage and a sequence identity threshold lowered by 10 percentage points.



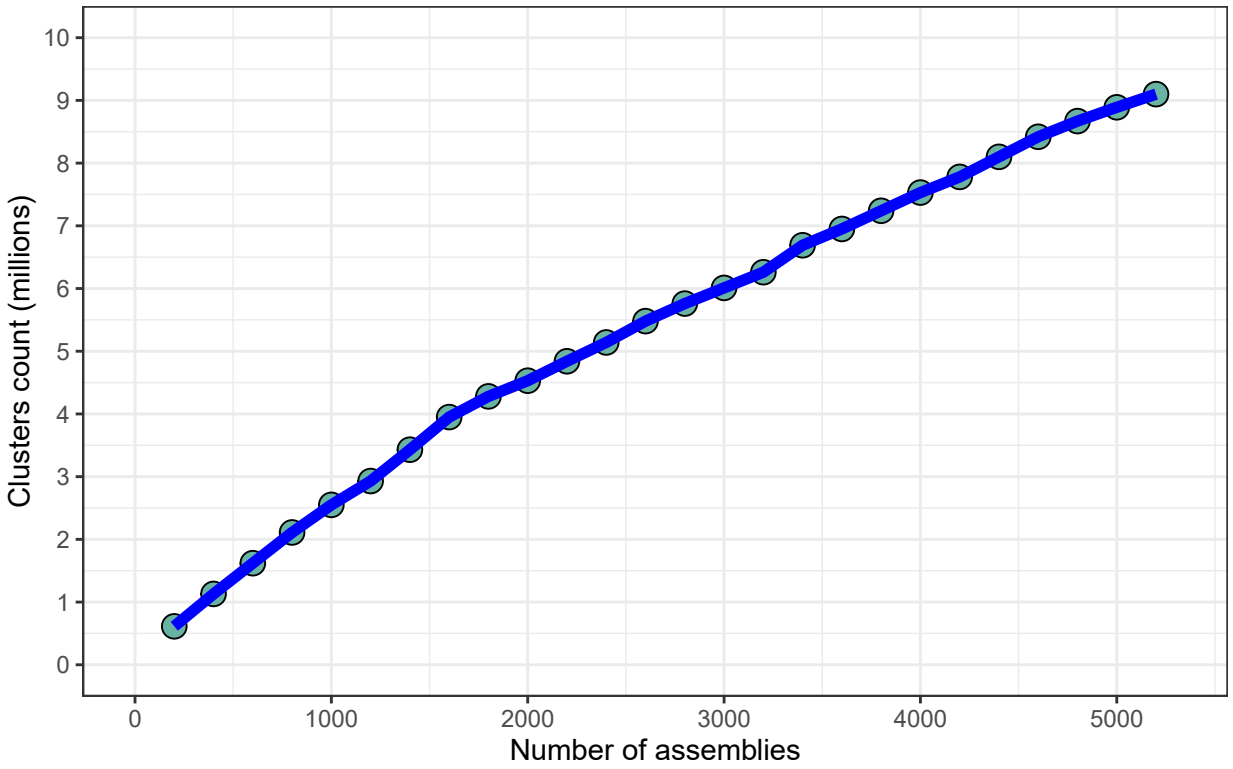
497

498 **Supplementary fig. 7 | Benchmark of assignment errors when clustering with DIAMOND DeepClust**
499 **and MMseqs2 using various clustering criteria.** Shown is the fraction of cluster member sequences that
500 are not assigned to the closest representative that satisfies the clustering criterion, defined by the e-value
501 of the local alignment. Smaller error rates indicate that most cluster members are indeed sufficiently
502 assigned to their closest representative and do not match other (closer) representatives better than the
503 assigned representative.

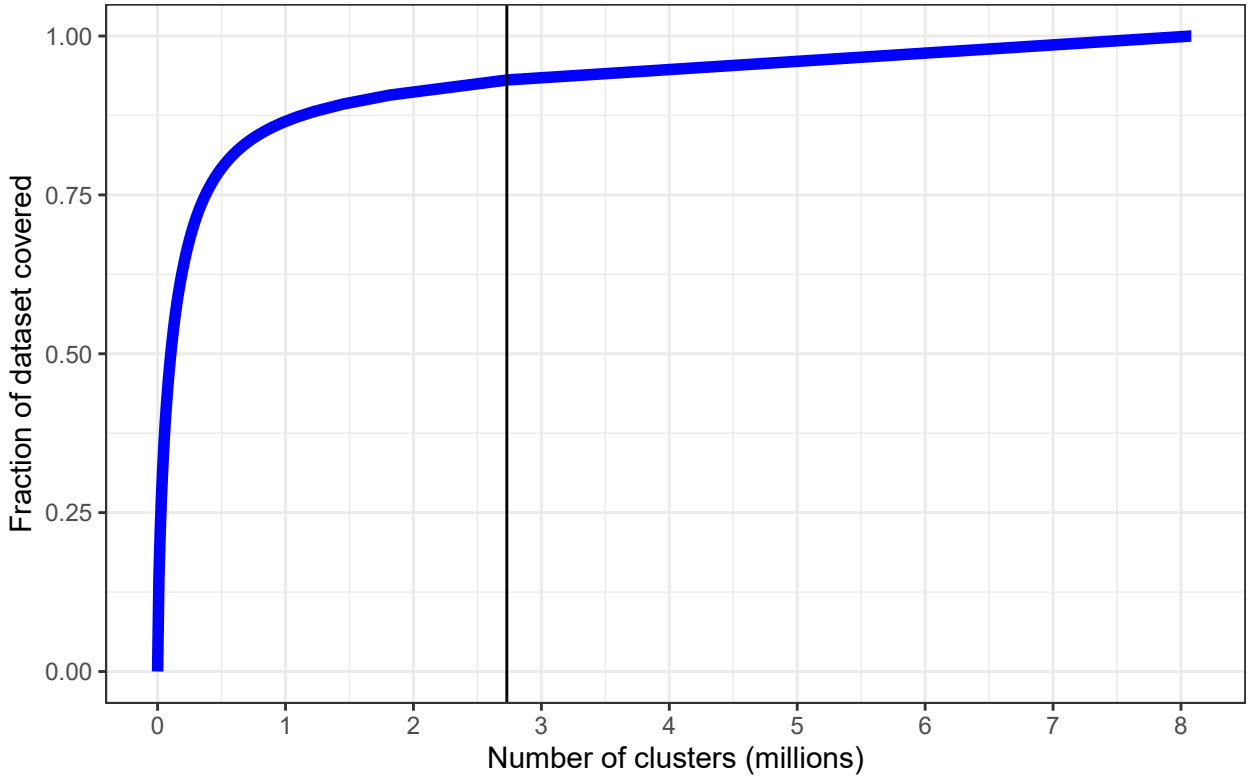
Clustering evaluation

We show four different metrics to measure errors in the computed clusterings. First, we show the fraction of representative sequences that fulfill the clustering criterion against another representative and represent false clusters that could have been merged into one unifying cluster but were missed due to limited alignment sensitivity (Supplementary fig. 2). Second, we show the fraction of cluster member sequences that do not satisfy the clustering criterion against their assigned representative, errors that are caused by the cascaded clustering algorithm (see distance error correction)(Supplementary fig. 5-6). Third, as a relaxed version of the second metric, we show the fraction of cluster member sequences that do not satisfy the clustering criterion against any representative and therefore constitute information that is potentially lost in the representative set (Supplementary fig. 3-4). Fourth, we report the fraction of cluster member sequences that are not assigned to the closest representative that satisfies the clustering criterion, as measured by the e-value of the local pairwise alignment (Supplementary fig. 7). The cascaded clustering algorithm as well as the incremental algorithm used by CD-HIT may both produce such suboptimal assignments by design, causing errors in the clustering that could be undesirable depending on the application (see assignment error correction).

We established the ground truth for these evaluations by computing a full Smith Waterman alignment of the evaluated representative or cluster member sequences against all representative sequences using DIAMOND in --swipe mode which guarantees perfect pairwise alignment sensitivity. Due to the much larger representative set, we used DIAMOND in default mode for the 90% identity runs. On account of the expense of computing the exhaustive Smith Waterman alignments, we evaluated these error metrics on random samples of 3,000 representative and cluster member sequences respectively. We sampled 3,000 clusters for each run from the set of clusters containing at least 5 sequences, and additionally sampled one member sequence out of each of these clusters. Since this evaluation is based on comparing raw alignment output, we added an option to DIAMOND to mimic the alignment score computations of MMseqs2 (Supplementary Information). We computed 95%-confidence intervals for the error metrics based on the procedure of Clopper and Pearson[29] (source data for Supplementary fig. 2-7).



Supplementary Fig. 8 | Eukaryotic protein clusters generated with DIAMOND DeepClust. Shown is the number of clusters resulting from clustering all 77.3 million protein sequences derived from up to 5,155 eukaryotic assemblies. The monotonically increasing graph illustrates that the eukaryotic sequence diversity space is not fully saturated yet, suggesting that the efforts of the Earth BioGenome Project have the potential to add a sufficient proportion of eukaryotic protein diversity to the existing protein universe.

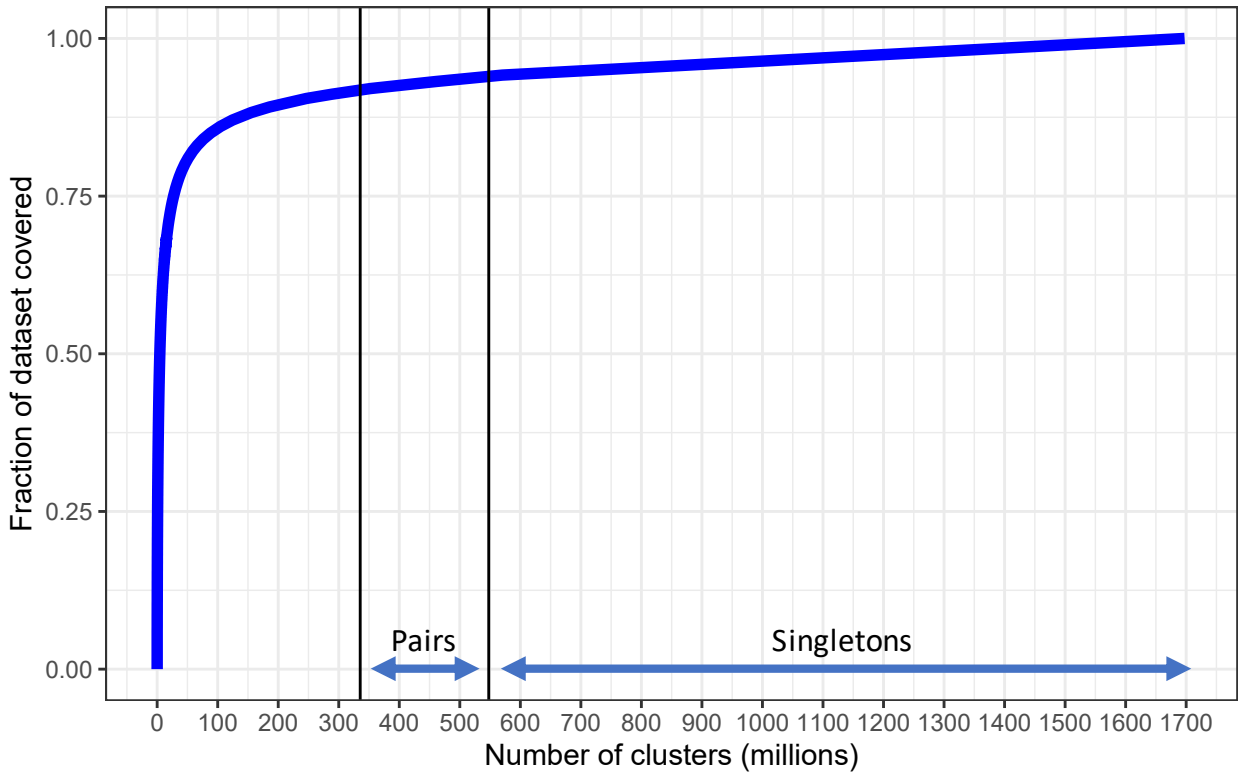


Supplementary Fig. 9 | Cluster representation for eukaryotic proteins from clusters generated with DIAMOND DeepClust. Shown is the number of clusters against the fraction of the 77.3 million eukaryotic proteins that are covered by the clustering. The vertical line illustrates that eukaryotic sequences cluster in similar proportions between non-singletons (93%) vs singletons (7%) as the full protein universe (~19 billion sequences; 94% non-singleton vs 6% singleton) which is currently dominated by protein sequences derived from microbial samples.

Projection for Earth BioGenome Era

Clustering the protein universe of our earth's biosphere allows us to quantify and understand the complexities and degrees of divergence when aiming to apply the comparative method across the tree of life and harness insights of relatedness for downstream analyses. To estimate the future applicability of DIAMOND DeepClust, we projected the computational effort of running it on a future dataset that would cover all protein sequences retrieved from the ~1.8 million eukaryotic species expected to be sequenced by the Earth BioGenome Project. To this end, we downloaded the protein sequences of 5,155 eukaryotic assemblies with annotated genes that were available in GenBank as of July 2022 and randomly partitioned them into groups of 200 assemblies. We clustered the protein sequences of the first group using DIAMOND DeepClust in very-sensitive mode at a 75% coverage cutoff and no identity cutoff and kept adding the sequences of another group to the clustering as described above under Cluster extension, until all assemblies were added. We observed a linear growth of the cluster count in the number of species with no apparent saturation (Supplementary fig. 8). Based on a number of 8,080,544 clusters for this dataset (when clustered together in a single run), we could project a linear growth of the cluster count as an upper bound estimate, resulting in 2.82 billion clusters for 1.8 million species. The clustering

computation is dominated by all-vs-all alignment in the most sensitive clustering round and can thus be assumed to scale roughly quadratically in the number of clusters. On the basis of this projection assumption and a DIAMOND v2 runtime of 2.07h on a 72-core server for the given computation, we project a computation time of at most 18.1 million CPU hours for processing the full Earth BioGenome dataset with DIAMOND DeepClust. Such a computation is already feasible on HPC systems hosted by the Max Planck Society today and illustrates that the scalability of DIAMOND DeepClust will enable users to learn from the protein sequences of millions of species once they are available. We note that the actual cluster count and run time will likely be lower due to saturation effects. While the nature of prokaryotic versus eukaryotic gene expression and regulation is fairly different, our clustering of all available eukaryotic proteins shows that after clustering the ratio between singletons (66%) vs non-singletons (34%) is comparable to our microbes dominated dataset of 19 billion sequences in that non-singleton clusters capture 93% of all eukaryotic proteins (Supplementary Figure 9).

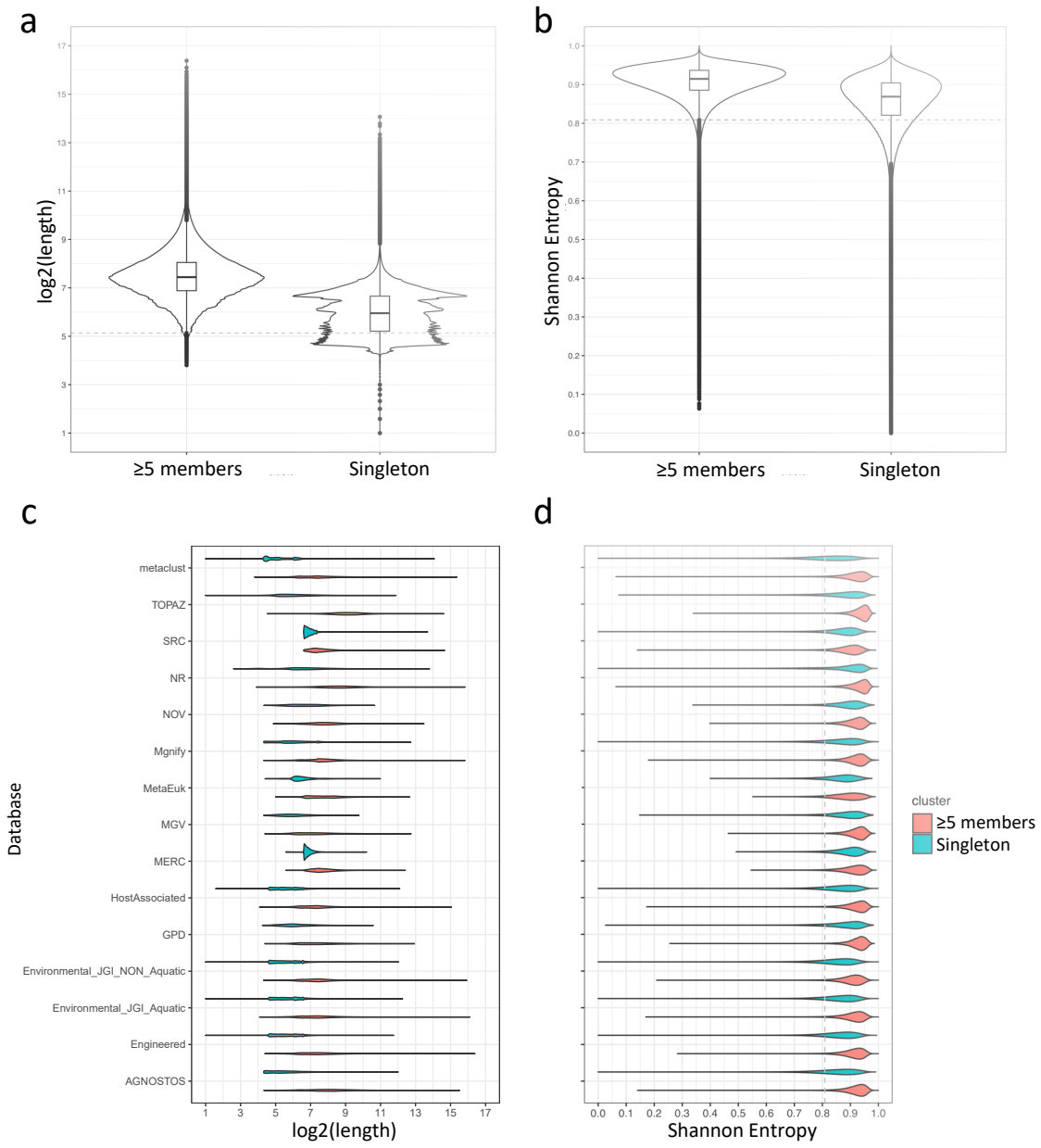


Supplementary fig. 10 | Cluster representation for the experimental study. Shown is the number of clusters generated with DIAMOND DeepClust against the fraction of the 19.4 billion input proteins that are covered by the clustering. The vertical lines indicate the start of clusters of size two and one (from left to right). The result illustrates that ~335 million representatives can capture 92% (~17.48 billion sequences) of the protein universe that comprises 19 billion sequences.

Experimental study

Recently, we introduced DIAMOND v2 to unlock the familiar functionality of a BLASTP search for tree-of-life scale applications in the Earth BioGenome era. To mimic protein alignments at this scale, we aligned ~280 million sequences from the NCBI NR database against ~39 million sequences of the UniRef50 database which resulted in ~32 billion pairwise alignments which could be performed in less than 6h on a HPC system (compared to several months with BLASTP) while matching the sensitivity of BLASTP[8]. While this speedup allowed us to introduce DIAMOND v2 as biosphere-ready protein aligner, searching protein sequences against millions of species and yielding trillions of pairwise alignments remains a computational challenge. Overcoming this bottleneck requires extensive dimensionality reduction of protein sequence space into sequence clusters to perform pairwise alignments only on the set of representative sequences rather than all sequences.

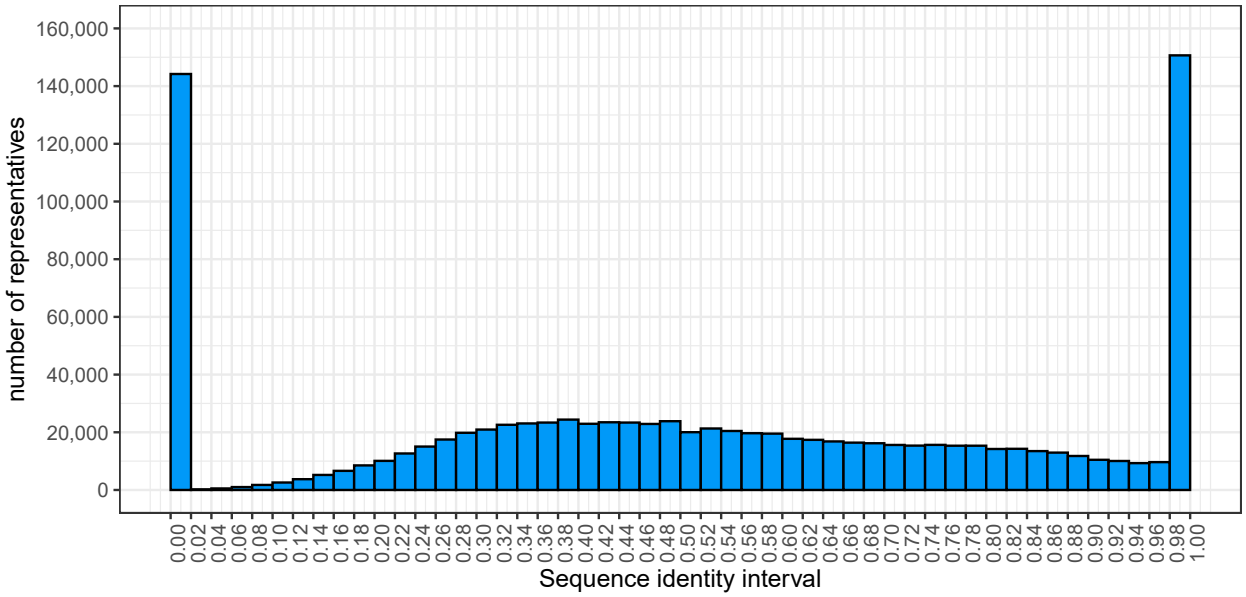
Using DIAMOND DeepClust, we performed an experimental study to showcase the power of dimensionality reduction through sequence clustering when the Earth BioGenome project will have successfully sequenced and assembled all ~1.8 million species. For this purpose, we collected ~22 billion protein sequences across all kingdoms of life (currently mostly comprising of microbes) to match the order of magnitude and sequence diversity space of the estimated ~27 billion eukaryotic protein sequences planned to be generated as part of the Earth BioGenome project (assuming ~1.8 million species times an average of ~15,000 genes per species). We note that compared to our collection of ~22 billion sequences (~19.4 billion deduplicated sequences), the Earth BioGenome set will include a much broader sequence diversity space derived from eukaryotes (while our current dataset is enriched in proteins mostly derived from microbial metagenomic samples).



Supplementary fig. 11 | Characterization of the 1.7 billion clusters generated by DIAMOND DeepClust. Illustrated are sequence length and Shannon entropy for cluster representatives. The length and Shannon entropy (sequence randomness) were computed for the representatives of clusters with at least five members compared with the singletons. IQR outlier cutoff was determined based on the distribution computed for clusters with at least five members, denoted as dashed lines. **a.** Sequence length distribution of non-singleton versus singleton clusters, **b**. Shannon entropy distribution of non-singletons versus singleton clusters. **c.** Sequence length distributions shown in **a**, now grouped by the public database the respective sequence originated from. **d.** Shannon entropy distributions shown in **b** now grouped by the public database the respective sequence originated from.

As a result of clustering ~19 billion deduplicated sequences with 30% sequence identity and 90% coverage thresholds across the tree of life, we determined ~1.70 billion clusters with 32% of clusters yielding more than one element, 12% of clusters holding more than five elements, and 68% of clusters with only one element (singletons)(Supplementary fig. 10). This striking result shows that while 68% of clusters contain unique sequences, these ~1.16 billion singletons

represent only 6% of the 19 billion sequences defining our current protein universe which begs the question whether these distinct proteins are derived from novel orphan genes or whether they represent assembly and annotation artifacts. To inform ongoing mega-assembly consortia about the proportion of novelty versus assembly artifact we compare the sequence and protein properties of singletons vs non-singletons. First, we filtered out obvious artifacts and poorly annotated sequences by running the repeat masking software tantan[28] with default settings on a random sample of 1,000,000 singletons vs non-singletons to evaluate the proportion of low-complexity sequences among putatively novel proteins. Interestingly only 10.5% of proteins were masked over >25% of their range by tantan compared to 9.41% for non-singletons. Next, we compare the length distributions of sequences derived from singleton clusters vs non-singleton clusters, since a well-studied feature of novel proteins is their short average length compared to evolutionarily conserved proteins[30]. As a result, we find that singletons are indeed significantly shorter than non-singleton protein sequences with singletons having a median length of 62 amino acids and non-singletons 174 amino acids (Supplementary fig. 11). In addition to sequence length, the entropy in the protein sequence is also known to vary between novel and evolutionarily established proteins[30]. Therefore, we calculated the Shannon entropy for all singletons vs non-singletons and observed that singletons indeed show significantly higher entropy values than non-singletons, thereby illustrating the uneven nature of these unique novel proteins (orphan polypeptides) (Supplementary fig. 11). Furthermore, we aligned random samples of 1,000,000 singletons vs 1,000,000 non-singletons (from clusters containing ≥5 members) against the ECOD database[20] using DIAMOND in ultra-sensitive mode and found that 11,601 sequences (~1.16%) generated alignments with e-value < 0.001, while 180,421 non-singleton sequences (~18%) generated alignments with the same alignment settings. This fact further establishes that the ~1.16 billion singletons we report in this study require further attention to assess their biological relevance. Since these singletons show signatures previously assigned to novel orphan proteins they encourage further studies. For example, the clustering run used in creating the AlphaFold2 Big Fantastic Database used only 18% of all clusters and constrained the cluster size to at least three elements, while removing 82% of singleton clusters. If, however, further research would reveal that they are the product of poor mega-assembly efforts, our current representation of the protein universe would turn out to be heavily biased by assembly quality, particularly when derived from metagenomic assemblies. Together, our experimental study reveals an unprecedented quantification of the protein universe and will enable future efforts to test whether more high quality genome assemblies based on long-read technologies will yield smaller numbers of singleton clusters when joining the tree of life or whether this unique diversity is an intrinsic feature of life itself.



Supplementary fig. 12 | DIAMOND protein search of DIAMOND DeepClust generated representative sequences against public databases. Shown is the sequence identity distribution when aligning a randomly drawn sample of one million representative sequences from clusters of size ≥5 from our experimental study against the combined BFD+MGnify databases.

To quantify the relatedness of our clusters to existing protein databases, we sampled one million representative sequences from the clusters containing five or more sequences and aligned them against the combined MGnify[21] and BFD[6] databases using DIAMOND in ultra-sensitive mode. We selected the best hit for each query and show the quantity of identities in the alignment divided by the length of the query as a histogram (Supplementary fig. 12) (queries with no hit are assigned zero identities). These results hint toward the possibility that the known protein sequence diversity space of microbes is already fairly saturated today. For the same representative sample, we computed ProtT5[16] embeddings (using the ProtT5-XL-UniRef50 model) for all sequences of length below 1024 and used mean-pooling of these embeddings as the basis for computing a UMAP[31] projection (Fig. 2). We labeled sequences according to whether (a) they could be annotated over ≥60% of their range against the combined SCOP[19]+ECOD[20]+CATH[17] databases using DIAMOND in ultra-sensitive mode, (b) they could be annotated over ≥60% of their range against the Pfam-A database[18] using HMMER[32] at an e-value threshold of 0.001, (c) an alignment against UniProt was found using DIAMOND ultra-sensitive that satisfied an e-value threshold of 0.001, a query coverage threshold of 90% and a sequence identity threshold of 30%, corresponding to the clustering criteria of our clustering and the BFD, (d) an alignment against BFD+MGnify was found satisfying the same criterion, (e) none of the above. If multiple conditions were true, we chose the label according to the order in the previous sentence.

Data retrieval of the protein biosphere

A total of 22,788,215,153 publically accessible protein sequences were retrieved from JGI IMG[33], SRC and MERC[34], MGnify[21], Metaclust[15], NCBI NR[35], AGNOSTOS[36], MetaEuk[37], SMAGs[38], TOPAZ[39],

GPD[40], NovelFams[41]and MGV[12] databases during March-April 2022 (additional details are provided in supplementary table 1).

Data pre-processing and filtering

We first sorted each downloaded FASTA file individually in memory by the length of the sequence in descending order, followed by a global disk-based merge sort on all files using GNU sort, resulting in a combined file of 22,788,215,153 protein sequences. Next, we computed hashes for all sequences which resulted in a deduplicated set of 19,387,935,704 unique sequences.

Clustering

We clustered the combined input file using a cascaded clustering approach in four rounds at increasing sensitivity employing the DIAMOND modes --faster, --fast, default and --sensitive, also using the option to linearize the comparison in the first round as described in the cascaded clustering section. The clustering criterion was 90% coverage of the cluster member sequence and 30% approximate sequence identity, corresponding to the parameters used to generate the Alphafold2 BFD[6]. To limit the use of resources and create checkpoints that could be reverted to in case of an error, we conducted the clustering rounds as an incremental procedure as follows. First, we split the input sequence file into chunks that we processed in sequential steps. Each chunk was first aligned against the current working set of representatives that resulted from the previous steps. Sequences that align against a representative were assigned to its respective cluster. Next, we subjected the remaining sequences that failed to map against an existing representative to all-vs-all alignment at the current round's sensitivity level and determined new representatives using the greedy vertex cover algorithm, which were then added to the working set. The clustering computation of the unclustered input sequences ran for 6.38 days on a single high-memory 72-core node for the first round, 36.8 hours on 16 nodes for the second round, 20.1 hours on 16 nodes for the third round, and 9.63 days on up to 27 nodes for the fourth round. In total, the computation consumed ~255,000 CPU hours. The resulting output of DIAMOND DeepClust generated 1,697,446,279 clusters, where 68% of clusters denote singletons and 20% of clusters had three or more members.

Code availability

DIAMOND DeepClust is available as Open Source Software under the GPL3 license from https://github.com/bbuchfink/diamond.

Data availability

We will make the Experimental Study dataset freely available upon journal publication. All source datasets that were used are publicly available.

Run	Id%	Time (h)	Cluster count
90% identity			
CD-HIT	90	586	181447250
MMseqs2 Linclust	90	7.13	189333075
MMseqs2 max-seqs=20	90	19.4	179657638
Diamond Linclust	90	0.93	187409369
Diamond	90	6.72	171579375
50% identity			
MMseqs2 max-seqs=100	50	14.6	44646306
Diamond	50	7.99	44592448
20% identity			
MMseqs2 s=6.0 max-seqs=1000	20	40.3	22328947
MMseqs2 s=6.0 max-seqs=300 + distance EC	20	281	31811284
MMseqs2 s=7.5 max-seqs=1000	20	96.6	22058400
MMseqs2 s=7.5 max-seqs=1000 + assignment EC	20	1776	22058400
Diamond sensitive	20	3.57	22499618
Diamond sensitive + distance EC	20	9.23	25664139
Diamond very-sensitive	20	7.65	22166526
Diamond ultra-sensitive	20	28.3	21955199
Diamond very-sensitive + assignment EC	20	368	22166526

Source data for figure 8 of the thesis/figure 1 from "Sensitive clustering of protein sequences at tree-of-life scale using DIAMOND DeepClust" (2023). Run time and cluster counts for clustering runs at 90%, 50%, and 20% sequence identity cutoffs. Distance EC refers to an additional round of error correction to fix cases where sequences do not satisfy the clustering criterion against their assigned representative. Assignment EC refers to error correction that assigns each sequence to its closest representative.

Run	Id%	Representatives with other representatives in distance	Not within distance to any representative	Not within distance to assigned representative	Not assigned to closest representative
90% identity					
CD-HIT	90	0.091	0.001	0.006	0.112
MMseqs2 Linclust	90	0.149	0.007	0.013	0.102
MMseqs2 max-seqs=20	90	0.019	0.012	0.022	0.084
Diamond Linclust	90	0.121	0.014	0.020	0.065
Diamond	90	0.015	0.023	0.033	0.039
50% identity					
MMseqs2 max-seqs=100	50	0.025	0.018	0.063	0.159
Diamond	50	0.033	0.027	0.05	0.094
20% identity					
MMseqs2 s=6.0 max-seqs=1000	20	0.136	0.041	0.158	0.307
MMseqs2 s=6.0 max-seqs=300 + distance EC	20	0.404	0	0.002	0.384
MMseqs2 s=7.5 max-seqs=1000	20	0.087	0.047	0.151	0.311
MMseqs2 s=7.5 max-seqs=1000 + assignment EC	20	0.097	0.048	0.049	0.015
Diamond sensitive	20	0.097	0.042	0.132	0.236
Diamond sensitive + distance EC	20	0.173	0.001	0.002	0.289
Diamond very-sensitive	20	0.059	0.041	0.124	0.227
Diamond ultra-sensitive	20	0.024	0.046	0.133	0.235
Diamond very-sensitive + assignment EC	20	0.072	0.044	0.050	0.033

Source data for supplementary figures 2, 3, 5, 7 from "Sensitive clustering of protein sequences at tree-of-life scale using DIAMOND DeepClust" (2023).

Error metrics for clustering runs at 90%, 50% and 20% sequence identity cutoffs. Source data for supplementary fig. 2 (fraction of cluster representative sequences that satisfy the clustering criterion against another representative/sensitivity errors), **supplementary fig. 3** (fraction of cluster member sequences that do not satisfy the clustering criterion against any representative sequence), **supplementary fig. 5** (fraction of cluster member sequences that do not satisfy the clustering criterion against their assigned representative sequence), and **supplementary fig. 7** (fraction of cluster member sequences that are not assigned to the closest representative that satisfies the clustering criterion, defined by the e-value of the local alignment). Additional MMseqs2 runs with different max-seqs parameters not used for comparison due to inferior error metrics are omitted for readability. The full source data including confidence intervals is included in the online version of the paper (https://www.biorxiv.org/content/biorxiv/early/2023/02/07/2023.01.24.525373/DC2/embed/media-2.xlsx?download=true).