

NG-meta-profiler: fast processing of metagenomes using NGLess, a domain-specific language

Luis Pedro Coelho¹ (<https://orcid.org/0000-0002-9280-7885>)

Renato Alves^{1,2} (<https://orcid.org/0000-0002-7212-0234>)

Paulo Monteiro³

Jaime Huerta-Cepas^{1,4} (<http://orcid.org/0000-0003-4195-5025>)

Ana Teresa Freitas³ (<https://orcid.org/0000-0002-2997-5990>)

Peer Bork^{1,5,6,7,*} (<https://orcid.org/0000-0002-2627-833X>)

July 12, 2018

¹ Structural and Computational Biology Unit, European Molecular Biology Laboratory, Heidelberg, Germany

² Candidate for Joint PhD degree from EMBL and Heidelberg University, Faculty of Biosciences

³ INESC-ID, Instituto Superior Técnico, University of Lisbon, Portugal

⁴ Centro de Biotecnología y Genómica de Plantas, Universidad Politécnica de Madrid (UPM), Instituto Nacional de Investigación y Tecnología Agraria y Alimentaria (INIA), Madrid, Spain

⁵ Max Delbrück Centre for Molecular Medicine, Berlin, Germany

⁶ Molecular Medicine Partnership Unit, University of Heidelberg and European Molecular Biology Laboratory, Heidelberg, Germany

⁷ Department of Bioinformatics, Biocenter, University of Würzburg, Würzburg, Germany.

* to whom correspondence should be addressed: bork@embl.de

Keywords: metagenomics; next-generation sequencing; domain-specific language

Abstract

NGLess is a domain specific language for describing next-generation sequence processing pipelines. It was developed with the goal of enabling user-friendly computational reproducibility.

Using this framework, we developed NG-meta-profiler, a fast profiler for metagenomes which performs sequence preprocessing, mapping to bundled databases, filtering of the mapping results, and profiling (taxonomic and functional). It is significantly faster than either MOCAT2 or htseq-count and (as it builds on NGLess) its results are perfectly reproducible. These pipelines can easily be customized and extended with other tools.

NGLess and NG-meta-profiler are open source software (under the liberal MIT licence) and can be downloaded from <http://ngless.embl.de> or installed through bioconda.

Introduction

Over the last decade, metagenomics has increasingly been applied for the study of microbial communities. Most work has focused on human-associated habitats¹, with a particular emphasis on the human gut microbiome^{2,3}. However, the same methodologies have been used for studying other host-associated microbiota⁴⁻⁶ or the marine microbiome⁷. Due to its size and complexity, several computational approaches have been proposed to handle these data, including bioinformatic pipelines combining different tools and approaches⁸⁻¹¹.

A typical metagenomics processing workflow can be divided into two distinct phases: in the first phase, raw data is processed (often using prebuilt reference databases) to generate a table of feature abundances (a profile). These features can be either taxonomic or functional annotations. Secondly, these profiles are analysed (often in regards to relevant metadata) using statistical methods and packages such as phyloseq¹², vegan¹³, or LEfSe¹⁴. In this work, we are focused on the first phase: namely obtaining functional and taxonomic profiles from raw metagenomic reads.

To this end, we present NG-meta-profiler, a collection of pre-configured pipelines based on the domain-specific language NGLess (Next Generation Language for less effortful analysis). Although NG-meta-profiler can be used as a standalone tool, the syntax and semantics of NGLess have been designed to be simple and human readable, allowing users to read or create their own pipelines, even without deep bioinformatics and programming knowledge. In other scientific contexts, domain-specific languages have been empirically found to increase productivity and user satisfaction^{15,16}. At the same time, NGLess is designed to enable perfect reproducibility of the computational process, an increasingly important concern¹⁷⁻¹⁹.

In version 1.0¹, NGLess implements the following tasks: (1) preprocessing, (2) assembly, (3) open-reading frame (ORF) finding, (4) mapping to sequence databases, (5) filtering of mapping results, (6) profiling (up-to-date taxonomic and functional profiling databases are provided), (7) summary plots.

Using this framework, we have developed NG-meta-profiler, a collection of pipelines for taxonomic and functional profiling of metagenomes. These standard analyses can be run with a single command. However, they can also serve as a starting point for customization by the user, including extending them with novel tools.

¹ The currently available version is named 0.9. We will to rename it to version 1.0 upon acceptance of this manuscript for publication.

Results

A bioinformatics-aware language leads to faster and more integrated tools

NGLess is a domain-specific language which was designed specifically for next-generation sequence (NGS) processing (see Supplemental File 1 and the online manual for a full description of the language). It contains builtin types which map to concepts in the sequencing domain (e.g., *short read*) as well as primitives that perform common operations (e.g., preprocessing a set of short reads).

The built-in knowledge of the sequence processing domain allows for best-practices to be automatic. For example, our tool always collects quality control statistics without the user having to specify it as an additional computational step.

Domain knowledge enables the interpreter to perform computations more efficiently. For example, even though users write their pipeline script in a purely linear fashion, the interpreter can automatically detect when parallelization opportunities are available and take advantage of multiple processors. When relying on external tools, NGLess can automatically detect possibilities to avoid the use of intermediate files whenever possible, while handling all format conversions internally.

Finally, error detection and reporting are significantly improved by having the tool be semantically aware of its goals. Given that debugging consumes a significant fraction of the time invested in developing computational pipelines, fast error detection can speed up the overall project. For example, it is possible to check whether inputs are readable and outputs are writable prior to starting interpretation. This benefits the user whenever they have made a mistake as errors are detected and reported immediately.

While introducing a novel language implies that the user needs to learn a new tool, the language is designed to be easily understandable to scientists familiar with the field. Alternatively, the Python interface to NGLess allows users familiar with that programming language to access NGLess functionality. Similarly, NG-meta-profiler is a command line tool that can be used directly without knowledge of NGLess.

NG-meta-profiler: a fast metagenomics profiler

NG-meta-profiler is a collection of predefined pipelines for analyzing metagenomic data, resulting in taxonomic and functional profiles. These pipelines are defined in the NGLess language and currently there are workflows available for human, mouse, pig, and dog gut as well as marine metagenomes.

We describe an abridged version of the human gut metagenome profiling pipeline in detail. At a high level, NG-meta-profiler, performs the following operations: (1) preprocess the reads (performing quality-based trimming and filtering of short reads and, if appropriate, discarding reads that align to the host), (2) map the reads against a predefined gene catalog selected for that biome, and (3) use the predefined gene annotations to build a profile (see Fig. 1).

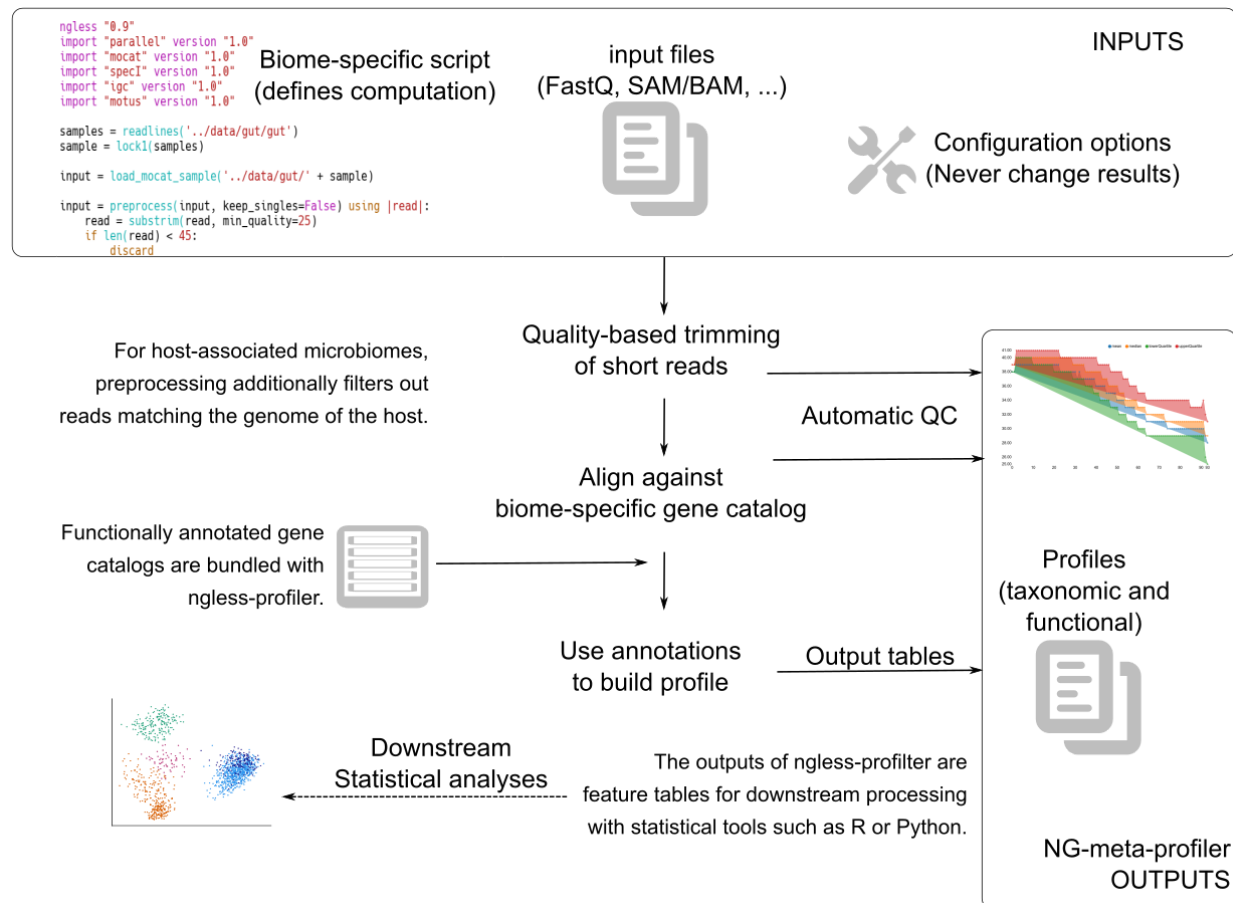


Figure 1 Schematic illustration of NG-meta-profiler.

We aim to describe the operation of the profiler while illustrating the NGLess language. These scripts provide a pre-defined pipeline, but they can also serve as a starting point for customization by the user.

The first element in an NGLess script is the version declaration:

```
ngless "0.9"
```

The user can then import helper modules. Specifying versions is required for reproducibility. The first module imported is the `mocat` module. This module allows users to load data that is organized with a sample per directory, an approach that is also used in MOCAT/MOCAT2^{8,9}. Secondly, we import the `igc` module to be able to use the integrated gene catalog (IGC) of the human gut²⁰.

```
import "mocat" version "0.9"
import "igc" version "0.9"
```

The data is loaded using the `load_mocat_sample` function. The input directory is given as the first of the command line arguments (which are stored in an array named `ARGV`, following the convention from other programming languages):

```
input = load_mocat_sample(ARGV[1])
```

`preprocess` is a built-in construct to perform quality based trimming and filtering of short reads:

```
qc_reads = preprocess(input, keep_singles=False) using |read|:
  read = substrim(read, min_quality=25)
  if len(read) < 45:
    discard
```

After preprocessing, it is necessary to reads that match the human genomes. For this operation, we will first align the reads against the `hg19` built-in reference:

```
human_mapped = map(qc_reads, reference='hg19')
```

Spurious alignments (those covering fewer than 45bps or at a lower than 90% nucleotide identity) are now removed, followed by discarding any reads still aligned to the human genome:

```
non_human = select(human_mapped) using |mr|:
  mr = mr.filter(min_match_size=45, min_identity_pc=90, action={unmatch})
  if mr.flag({mapped}):
    discard
```

The `non_human` object contains a `MappedShortReadSet` (the `NGLess` representation of the information in a SAM file²¹). In order to extract only the sequences (effectively convert it back to a set of FastQ files), the `as_reads` function is used:

```
non_human_reads = as_reads(non_human)
```

Now, these sequences are mapped to the `igc` reference (the integrated gene catalogue, imported earlier), and the results are given to count function to obtain a functional profile:

```
igc_mapped = map(non_human_reads, reference='igc')
igc_counts = count(igc_mapped,
  features=['KEGG_ko', 'eggNOG_OG'],
  normalization={scaled})
```

Finally, `NG-meta-profiler` saves the result to the file `igc.profile.txt` within the output directory provided by the user:

```
write(igc_counts, ofile=ARGV[2] </> 'igc.profile.txt')
```

Bundled databases and modules

As part of the first release of NG-meta-profiler, we bundle several genomes (including human and mouse) as well as gene catalogs for the human gut microbiome²⁰, the marine microbiome⁷, and three non-human mammal microbiomes (pig,⁵ dog⁶ and mouse⁴). These gene catalogs have been functionally annotated with eggNOG-mapper²² so that functional profiles can be generated by NGLess (see Table 1). In the current version, NG-meta-profiler can produce eggNOG orthologous group²³, KEGG orthologous groups (KO)²⁴, SEED²⁵, and BiGG²⁶ abundance profiles. Additionally, genomes of several organisms (see Supplemental Text 1) are also provided. These are used for filtering out host reads in host-associated metagenomes in NG-meta-profiler.

All of these resources are automatically downloaded by NGLess the first time they are used so that the initial download is small (currently 15 MiB) and each user only downloads those resources they effectively use.

Database	Size (million genes)	Comment
igc	9.9	Integrated gene catalog for the human gut ²⁰
om-rgc	40	Ocean microbial gene catalog ⁷
mouse-gut	2.6	Gene catalog of the mouse gut ⁴
pig-gut	7.7	Gene catalog of the pig gut ⁴
dog-gut	1.2	Gene catalog of the dog gut ⁶

Table 1 Gene catalogs bundled with NG-meta-profiler

Several operations in NGLess are performed with bundled software. *De novo* assembly is performed using MEGAHIT²⁷, which has been found to perform well for metagenomics^{28,29}. Open reading frame (ORF) finding is performed with Prodigal³⁰. By default, mapping is performed using bwa³¹, but minimap2³² is also available. Additional built-in modules provide extra functionality as shown in Table 2.

Module name	Comment
parallel	Process multiple samples in parallel
mocat	Compatibility with MOCAT/MOCAT2 ^{8,9}

specI	specI profiling (reference based metagenomics taxonomic profiling ³³)
motu	mOTU profiling (taxonomic profiling of metagenomes ³⁴)
minimap2	minimap2 mapper ³²

Table 2 NGLess builtin modules that add extra functionality.

Benchmarking

We compared the performance of NG-meta-profiler with both MOCAT2⁹ and a pipeline based on calling bwa without preprocessing data and htseq-count³⁵ for profiling human gut³⁶ and ocean metagenomes⁷. In all cases, we used the NG-meta-profiler databases and set parameters so that the results are identical, up to rounding errors. For this benchmark, 8 threads were used (except for the htseq-count software which only supports a single thread).

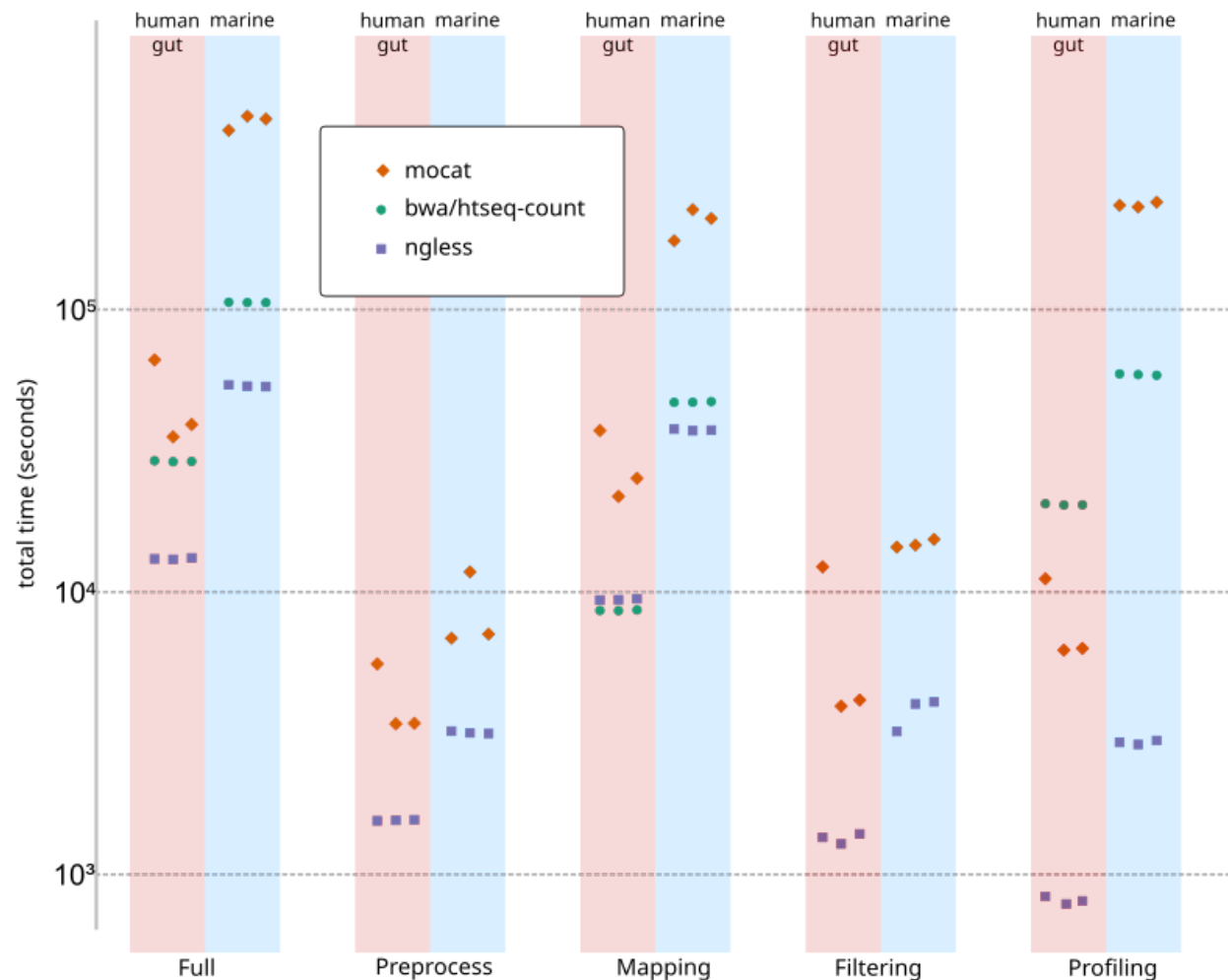


Figure 2 Timing comparison of NGLess and other tools. Three replicates are shown for each tool. The bwa/htseq-count pipeline does not include Preprocess and Filtering steps.

Results are presented in Fig. 2 (the Supplemental Software contains a preprocessed table with the full results). NG-meta-profiler clearly outperforms the other solutions in this task. When compared to MOCAT2, the NG-meta-profiler runs 11x faster in the marine benchmark, and 2.6x in the human gut metagenome benchmark.

The larger ratio in the larger ocean microbial gene catalog (40 million genes) compared to the integrated gene catalog used for profiling the human gut (10 million genes) is evidence that NGLess scales better to very large catalogs.

Note that running the full NG-meta-profiler pipeline takes less time than the sum of the individual steps as the pipeline is optimized as a whole (e.g., it avoids generating unnecessary intermediate files).

Pipelines developed with NGLess are reproducible

Unlike tools which are based on traditional programming languages, NGLess is designed from the ground up with reproducibility as a goal.

As seen above, every pipeline defined with NGLess includes a version declaration and every imported module specifies the particular version which is being imported. Documenting the version of all the tools and dependencies used for a given analysis is considered a best practice^{37,38}, but is not always followed. By making it a requirement within the script, NGLess ensures that this best practice is adopted. At the same time, NGLess lowers the necessary effort when compared to having to record the version of all tools and dependencies manually. For example, the versions of samtools²¹ and bwa³¹ used internally (and shipped with NGLess) are implicitly fixed by specifying the NGLess version. To encourage that credit be given to the original authors, NGLess will print out references for any tools that are used, asking the user to cite them in publications.

Furthermore, although external configuration and command line options may change *how* results are computed (e.g., how many threads to use, where to store temporary files), the results do not depend on any information outside the script. This separation of implementation details from the data processing specification has the added potential of making the resulting code easier to port between systems³⁹.

Extensibility and integration into the wider ecosystem of bioinformatics tools

Pipelines defined with NGLess are easily extensible. We encourage users of NG-meta-profiler to customize these pipelines to their specific problems and to extend them as desired. Functions can be added to the language based on external software by specifying the interface in a text format and importing it from the main script.

In May 2018, we opened up to the community a repository of external modules (<https://github.com/ngless-toolkit/ngless-contrib>) where contributions of new integrations are accepted. At the moment, integration of MetaPhlAn2 (a tool for profiling microbial species based on taxa-specific marker genes⁴⁰) and salmon (a tool which generates abundance profiles based on k-mers⁴¹) are available.

Alternatively, NGLess based analyses can be integrated into larger pipelines. Most existing bioinformatics pipelines are performed with command-line based software. To facilitate integration with existing tools, we provide several tools based on NGLess with a command line interface. In addition, we provide Common Workflow Language (CWL) descriptions of these tools which enable their use as part of CWL workflows⁴².

NGLess scripts can (with some limitations) be automatically exported as CWL tools in order to be embedded in larger projects. When passed the `--export-cwl` option, NGLess will output a CWL description of a given script which can then be embedded in a larger CWL-based pipeline.

Finally, NGLess can also be used as an embedded language within Python, a programming language that is widely used for scientific computing¹⁵. With this interface, NGLess-based pipelines can be defined by Python-based scripts.

Discussion

NGLess puts forward a different approach for defining data analysis pipelines: the use of a domain-specific language for sequence analysis. Several tools had already used a domain-specific language to define a computational pipeline. The classical Make tool was originally designed for compiling software, but has been used as the basis of a scientific pipeline system⁴³ uses its own internal language, as do the more modern pipelining tool Snakemake⁴⁴ and nextflow⁴⁵. These tools operate by organizing the computation around calls to command-line software. As such, they are fully generic and can be used in a wide range of problems. NGLess trades off this generality to achieve higher usability within its problem domain.

Using this framework, we developed NG-meta-profiler which generates taxonomic and functional profiles from metagenomes based on prebuilt gene catalogs that are provided with the tool. When compared with other alternatives, NG-meta-profiler performs reference-based functional and taxonomic profiling much faster. Furthermore, this collection of scripts can be easily adapted and extended by the users within the NGLess framework to perform novel functions.

Software and data availability

NGLess is available as open source software at <http://github.com/ngless-toolkit/ngless>. Additionally, NGLess is available as a bioconda package⁴⁶ and in container form (through biocontainers⁴⁷). Documentation and tutorials can be found at <http://ngless.embl.de>. All the databases mentioned can be downloaded automatically with the NGLess tool or accessed using the DOI 10.5281/zenodo.1299267.

Jug⁴⁸ based scripts to download the data and run the benchmarks are available at <https://github.com/ngless-toolkit/NGLess2018benchmark>. This repository also contains the results of running the benchmarks on our servers and all downstream processing. The benchmarks were run on an Intel Xeon (CPU model “E7- 4830 @ 2.13GHz”) with 32 cores (64 virtual cores).

The sequence datasets used for the benchmark are available from the European Nucleotide Archive (ENA) (accession numbers: SAMEA2621229, SAMEA2621155, SAMEA2621033, SAMEA2467039, SAMEA2466896, and SAMEA2466965).

Acknowledgements

The authors thank João Carriço (Instituto de Medicina Molecular, University of Lisbon) as well as the members of the Bork lab for helpful comments. We thank Anna Głazek (EMBL) for several patches to the code. Beta users of NGLess are thanked for their feedback and bug reports.

Funding was provided by the European Union's Horizon 2020 Research and Innovation Programme (grant #686070; DD-DeCaF), the European Research Council (ERC) MicrobioS (ERC-AdG-669830), the Fundação para a Ciência e a Tecnologia (grant EXCL/EEI-ESS/0257/2012; DataStorm: Large-Scale Data management in cloud environments), and the European Molecular Biology Laboratory (EMBL).

Author contribution

LPC conceived of the project and wrote the first draft of the manuscript. LPC, ATF, and PB coordinated and oversaw the study. LPC, RA, JHC, and PM developed the software and associated databases. LPC, RA, PM, JHC, and PB drafted the manuscript and supplemental text. LPC and RA designed and performed the benchmark study. All authors contributed to the review of the manuscript before submission for publication. All authors read and approved the final manuscript.

References

1. Grice, E. A. *et al.* Topographical and temporal diversity of the human skin microbiome. *Science* **324**, 1190–1192 (2009).
2. Schmidt, T. S. B., Raes, J. & Bork, P. The human gut microbiome: From association to modulation. *Cell* **172**, 1198–1215 (2018).
3. Gilbert, J. A. *et al.* Current understanding of the human microbiome. *Nat. Med.* **24**, 392–400 (2018).
4. Xiao, L. *et al.* A catalog of the mouse gut metagenome. *Nature Biotechnology* **33**, 1103–1108 (2015).
5. Xiao, L. *et al.* A reference gene catalogue of the pig gut microbiome. *Nature Microbiology* **1**, 16161 (2016).
6. Coelho, L. P. *et al.* Similarity of the dog and human gut microbiomes in gene content and response to diet. *Microbiome* **6**, 72 (2018).
7. Sunagawa, S. *et al.* Structure and function of the global ocean microbiome. *Science* **348**, 1261359 (2015).
8. Kultima, J. R. *et al.* MOCAT: A metagenomics assembly and gene prediction toolkit. *PLoS ONE* **7**, e47656 (2012).
9. Kultima, J. R. *et al.* MOCAT2: A metagenomic assembly, annotation and profiling framework. *Bioinformatics (Oxford, England)* **32**, 2520–2523 (2016).
10. Treangen, T. J. *et al.* MetAMOS: A modular and open source metagenomic assembly and analysis pipeline. *Genome biology* **14**, R2 (2013).

11. Narayanasamy, S. *et al.* IMP: A pipeline for reproducible reference-independent integrated metagenomic and metatranscriptomic analyses. *Genome Biol.* **17**, 260 (2016).
12. McMurdie, P. J. & Holmes, S. Phyloseq: An R package for reproducible interactive analysis and graphics of microbiome census data. *PLoS One* **8**, e61217 (2013).
13. Oksanen, J. *et al.* The vegan package. *Community ecology package* **10**, 631–637 (2007).
14. Segata, N. *et al.* Metagenomic biomarker discovery and explanation. *Genome Biol.* **12**, R60 (2011).
15. Prabhu, P. *et al.* A survey of the practice of computational science. 19 (2011). doi:[10.1145/2063348.2063374](https://doi.org/10.1145/2063348.2063374)
16. Johanson, A. N. & Hasselbring, W. Effectiveness and efficiency of a domain-specific language for high-performance marine ecosystem simulation: A controlled experiment. *Empirical Software Engineering* **22**, 2206–2236 (2017).
17. Donoho, D. L., Maleki, A., Rahman, I. U., Shahram, M. & Stodden, V. Reproducible research in computational harmonic analysis. *Computing in Science & Engineering* **11**, 8–18 (2009).
18. Vandewalle, P., Kovacevic, J. & Vetterli, M. Reproducible research in signal processing. *Signal Processing Magazine, IEEE* **26**, 37–47 (2009).
19. Fomel, S. Reproducible research as a community effort: Lessons from the madagascar project. *Computing in Science & Engineering* **17**, 20–26 (2015).
20. Li, J. *et al.* An integrated catalog of reference genes in the human gut microbiome. *Nature Biotechnology* **32**, 834–841 (2014).
21. Li, H. *et al.* The sequence alignment/map format and samtools. *Bioinformatics* **25**, 2078–2079 (2009).
22. Huerta-Cepas, J. *et al.* Fast genome-wide functional annotation through orthology assignment by eggNOG-mapper. *Molecular Biology and Evolution* **34**, 2115–2122 (2017).
23. Huerta-Cepas, J. *et al.* eggNOG 4.5: A hierarchical orthology framework with improved functional annotations for eukaryotic, prokaryotic and viral sequences. *Nucleic Acids Res.* **44**, D286–93 (2016).
24. Kanehisa, M. *et al.* Data, information, knowledge and principle: Back to metabolism in KEGG. *Nucleic Acids Res.* **42**, D199–205 (2014).
25. Overbeek, R. *et al.* The subsystems approach to genome annotation and its use in the project to annotate 1000 genomes. *Nucleic Acids Res.* **33**, 5691–5702 (2005).

26. King, Z. A. *et al.* BiGG models: A platform for integrating, standardizing and sharing genome-scale models. *Nucleic Acids Res.* **44**, D515–22 (2016).
27. Li, D., Liu, C.-M., Luo, R., Sadakane, K. & Lam, T.-W. MEGAHIT: An ultra-fast single-node solution for large and complex metagenomics assembly via succinct de bruijn graph. *Bioinformatics* **31**, 1674–1676 (2015).
28. Darling, A. E. *et al.* Critical assessment of metagenome interpretation—a benchmark of metagenomics software. *Nature Methods* **14**, 1063 (2017).
29. Awad, S., Irber, L. & Titus Brown, C. Evaluating metagenome assembly on a simple defined community with many strain variants. *bioRxiv* 155358 (2017).
30. Hyatt, D. *et al.* Prodigal: Prokaryotic gene recognition and translation initiation site identification. *BMC Bioinformatics* **11**, 1–11 (2010).
31. Li, H. Aligning sequence reads, clone sequences and assembly contigs with bwa-mem. (2013).
32. Li, H. Minimap2: Pairwise alignment for nucleotide sequences. *Bioinformatics* **34**, 171–178 (2018). doi:[10.1093/bioinformatics/bty191](https://doi.org/10.1093/bioinformatics/bty191)
33. Mende, D. R., Sunagawa, S., Zeller, G. & Bork, P. Accurate and universal delineation of prokaryotic species. *Nature Methods* **10**, 881–884 (2013).
34. Sunagawa, S. *et al.* Metagenomic species profiling using universal phylogenetic marker genes. *Nature Methods* **10**, 1196–1199 (2013).
35. Anders, S., Pyl, P. T. & Huber, W. HTSeq—a python framework to work with high-throughput sequencing data. *Bioinformatics* **31**, 166–169 (2015).
36. Zeller, G. *et al.* Potential of fecal microbiota for early-stage detection of colorectal cancer. *Molecular Systems Biology* **10**, (2014).
37. Wilson, G. *et al.* Best practices for scientific computing. *PLoS Biology* **12**, e1001745 (2014).
38. Smith, A. M., Katz, D. S. & Niemeyer, K. E. Software citation principles. *PeerJ Comput. Sci.* **2**, e86 (2016).
39. Johanson, A. & Hasselbring, W. Software engineering for computational science: Past, present, future. *Computing in Science Engineering* 1–1 (2018).
40. Truong, D. T. *et al.* MetaPhlAn2 for enhanced metagenomic taxonomic profiling. *Nat. Methods* **12**, 902–903 (2015).
41. Patro, R., Duggal, G., Love, M. I., Irizarry, R. A. & Kingsford, C. Salmon provides fast and bias-aware quantification of transcript expression. *Nat. Methods* **14**, 417–419 (2017).

42. Stojanovic, P. A. M. R. C. N. T. B. C. J. C. M. H. A. K. J. K. D. L. H. M. M. N. M. S. S. S.-R. L. Common workflow language, v1.0. (2016). doi:[10.6084/m9.figshare.3115156.v2](https://doi.org/10.6084/m9.figshare.3115156.v2)
43. Schwab, M., Karrenbach, M. & Claerbout, J. Making scientific computations reproducible. *Computing in Science & Engineering* **2**, 61–67 (2000).
44. Köster, J. & Rahmann, S. Snakemake—a scalable bioinformatics workflow engine. *Bioinformatics (Oxford, England)* **28**, 2520–2 (2012).
45. Di Tommaso, P. *et al.* Nextflow enables reproducible computational workflows. *Nat. Biotechnol.* **35**, 316–319 (2017).
46. Grüning, B. *et al.* Bioconda: A sustainable and comprehensive software distribution for the life sciences. *bioRxiv* 207092 (2017). doi:[10.1101/207092](https://doi.org/10.1101/207092)
47. Veiga Leprevost, F. *da et al.* BioContainers: An open-source and community-driven framework for software standardization. *Bioinformatics* **33**, 2580–2582 (2017).
48. Coelho, L. P. Jug: Software for parallel reproducible computation in python. *Journal of Open Research Software* **5**, (2017).