Bioinformatics, 38(20), 2022, 4817–4819
https://doi.org/10.1093/bioinformatics/btac588
Advance Access Publication Date: 27 August 2022
Applications Note

Sequence analysis

APSCALE: advanced pipeline for simple yet comprehensive analyses of DNA metabarcoding data

Dominik Buchner[1,*,†], **Till-Hendrik Macher**[1,*,†], **and Florian Leese**[1,2]

[1]University of Duisburg-Essen, Faculty of Biology, Aquatic Ecosystem Research, Essen 45141, Germany and [2]Univeresity of Duisburg-Essen, Centre for Water and Environmental Research (ZWU), Essen 45141, Germany

*To whom correspondence should be addressed.
†The authors wish it to be known that, in their opinion, the first two authors should be regarded as Joint First Authors.

Associate Editor: Can Alkan

Received on February 15, 2022; revised on August 19, 2022; editorial decision on August 19, 2022; accepted on August 26, 2022

Abstract

Summary: DNA metabarcoding is an emerging approach to assess and monitor biodiversity worldwide and consequently the number and size of data sets increases exponentially. To date, no published DNA metabarcoding data processing pipeline exists that is (i) platform independent, (ii) easy to use [incl. graphical user interface (GUI)], (iii) fast (does scale well with dataset size) and (iv) complies with data protection regulations of e.g. environmental agencies. The presented pipeline APSCALE meets these requirements and handles the most common tasks of sequence data processing, such as paired-end merging, primer trimming, quality filtering, clustering and denoising of any popular metabarcoding marker, such as internal transcribed spacer, 16S or cytochrome c oxidase subunit I. APSCALE comes in a command line and a GUI version. The latter provides the user with additional summary statistics options and links to GUI-based downstream applications.

Availability and implementation: APSCALE is written in Python, a platform-independent language, and integrates functions of the open-source tools, VSEARCH (Rognes *et al.*, 2016), cutadapt (Martin, 2011) and LULU (Frøslev *et al.*, 2017). All modules support multithreading to allow fast processing of larger DNA metabarcoding datasets. Further information and troubleshooting are provided on the respective GitHub pages for the command-line version (https://github.com/DominikBuchner/apscale) and the GUI-based version (https://github.com/TillMacher/apscale_gui), including a detailed tutorial.

Contact: till-hendrik.macher@uni-due.de or dominik.buchner@uni-due.de

Supplementary information: Supplementary data are available at *Bioinformatics* online.

1. Introduction

DNA metabarcoding is becoming an established and efficient method to identify the taxonomic composition of biological samples and is increasingly applied worldwide as a comprehensive biodiversity monitoring approach for terrestrial, freshwater and marine habitats (Compson *et al.*, 2020). Coupled with technical advancements in sequencers, data availability increases exponentially. Software to process DNA metabarcoding sequencing data, like USEARCH (Edgar, 2010) and VSEARCH (Rognes *et al.*, 2016) or pipelines like QIIME2 (Bolyen *et al.*, 2019), DADA2 (Callahan *et al.*, 2016) or slim (Dufresne *et al.*, 2019) exists. Typically, similar analytical steps are implemented in all those pipelines in order to produce a final read table. However, performing the analysis of sequencing data typically requires advanced bioinformatic user skills. Our aim was to bridge the gap between metabarcoding data and end users (biodiversity researchers and environmental managers) further by developing a DNA metabarcoding pipeline that is (i) platform-independent, (ii) easy to use [graphical user interface (GUI)], (iii) fast (i.e. does scale well with dataset size) and (iv) runs locally to be compliant with data protection regulations at environmental agencies, where data cannot be transferred to external servers. Therefore, we developed APSCALE, a software specifically targeting biologists without bioinformatics background to reliably analyze DNA metabarcoding data.

2. APSCALE

The software comes in the command-line version APSCALE (version 1.5.5) and the GUI version APSCALE-GUI (version 1.1.2), which are both available from the Python Package Index (PyPI) at https://pypi.org/project/apscale/ and https://pypi.org/project/apscale_gui/. Both versions can be easily installed using the Python package installer (pip). APSCALE runs on all major operating systems (Linux, MacOS and Windows) and requires Python3.7 (or higher) and VSEARCH (version 2.19.0 or higher). Additionally, the installation of the zlib library is mandatory for the support of gzip-compressed files. For Unix-based systems, zlib is shipped with VSEARCH, while Windows users must install zlib manually. All further dependencies are automatically installed. Updates for APSCALE can be downloaded via pip and users are automatically informed when a new version is available.

APSCALE uses dedicated project folders to enhance file management during the data processing. New projects are automatically created from within the pipeline. Within these projects, all output data will be saved into a consistent folder structure. All output files (.fastq and .fasta format) are saved in a compressed gzip format.

Since each DNA metabarcoding dataset requires specific settings for processing, the user can easily adjust the running conditions via an excel spreadsheet that is automatically loaded by APSCALE. Changes can either be applied directly to the spreadsheet or through the GUI. Each module offers the user to apply changes to the default values. In the general settings, the number of used cores (default = available cores * 0.75) and the gzip compression level (default = 6) can be adjusted. In the paired-end merging module, the maximum difference percentage (default = 25), the number of maximum differences (default = 199) and the minimum overlap length (default = 5) can be set. The primer trimming module requires sequences of both the forward and reverse metabarcoding primers (5′-3′ orientation). Additionally, anchoring can be enabled to only pass sequences where the primer sequence is the prefix of the 5′-end. The quality filtering module requires a maximum expected error threshold (default = 1) for quality filtering and the minimum and maximum expected length of the target fragment (user input required). Individual samples are dereplicated and singletons are kept for mapping them in the OTU clustering and denoising modules. Before pooling all samples for global dereplication, only reads that are present with an abundance above a minimum size are kept (default = 4) as an additional quality filtering step. Consequently, only operational taxonomic units (OTUs) represented by at least four reads (default) in one sample are kept. The OTU clustering algorithm implemented in VSEARCH is based on an adjustable sequence similarity threshold (default = 97). Sequence denoising is based on the alpha value (default = 2), which corresponds to the number of allowed sequence differences, described in (Edgar, 2016). The minimum size of an exact sequence variant (ESV) to be kept can be adjusted (default = 8). Additionally, APSCALE offers an internal python-based version of the r-package LULU (Frøslev *et al.*, 2017), an algorithm for post-clustering curation that aims to provide more reliable biodiversity estimates. Here, the minimum similarity (default = 84), minimum relative co-occurrence (default = 95) and minimum ratio (default = 1) can be adjusted.

To start the data processing workflow, sequencing raw data can be copied to a raw data folder. Demultiplexing is not handled by APSCALE directly, since different tagging schemes exist (e.g. combinatorial indexing or mixed orientation indexing), which can hardly be covered all in a single pipeline. However, a separate package dedicated to demultiplexing (https://github.com/DominikBuchner/demultiplexer) can directly be called via the GUI. APSCALE uses demultiplexed data as input and automatically loads all files from the demultiplexing folder detecting corresponding files via file extension.

The data analysis is composed of the following steps: (i) demultiplexed paired-end reads are merged using the vsearch –fastq_mergepairs command, (ii) primers are removed using the adapter trimming command in cutadapt, (iii) reads are filtered based on per-base quality (maximum expected error) and read length thresholds using the vsearch –fastq_filter command and (iv) reads are dereplicated using the vsearch –fastx_uniques command, followed by the abundance filtering step. Subsequently, all individually dereplicated and filtered files are pooled into a single file, which is globally dereplicated. The pooled reads can then be (v) clustered into OTUs using the vsearch –cluster-size command and (vi) denoised into ESVs using the vsearch –cluster_unoise command. Chimeras are automatically detected and removed from both the OTUs and ESVs using the vsearch –uchime_denovo command. OTUs and ESVs are mapped against the dereplicated files prior to the read abundance filter using the vsearch –usearch_global (OTUs) and vsearch –search_exact (ESVs) functions to produce OTU and ESV tables. Lastly, (vii) both OTUs and ESVs are filtered using the LULU algorithm to reduce the number of erroneous OTUs and ESVs.

Taxonomic assignment can be conducted from the generated .fasta files (containing the OTUs or ESVs) using the local BLAST (based on (Camacho, 2009)) and NCBI BLAST modules or the software BOLDigger (Buchner and Leese, 2020), which are all integrated in the GUI. Furthermore, the APSCALE output files are compatible with downstream analyses using TaxonTableTools (Macher *et al.*, 2021). Additionally, various raw data and summary statistics can be visualized from within APSCALE-GUI and saved as .html and .pdf plots, using the python package plotly (https://plot.ly). All progress of the data processing is written to a log file.

3. Benchmark

A benchmark with two different datasets, one set of 10 aquatic invertebrate mock communities (single MiSeq run 23 M reads) and one set of 256 insect monitoring Malaise trap samples (two HiSeq runs; 729 M reads), was conducted to evaluate the accuracy and runtime performance of APSCALE compared to the established DNA metabarcoding pipelines DADA2 (generating ESVs) and QIIME2 (generating OTUs). A detailed description of the benchmark can be found in Supplementary material S1. The datasets were produced according to the workflow presented in Buchner *et al.* (2021). In short, APSCALE showed significantly lower runtimes while recovering similar communities (Supplementary Figs S1 and S2). The total number of sequence clusters (ESVs/OTUs), however, varied depending on the pipeline.

Acknowledgements

We thank the NFDI4Biodiversity community and Leese Lab Journal Club for input on the software.

Authors' contributions

D.B. and T.M. conceived and designed the software. D.B. and T.M. wrote the Python packages. F.L. provided input to the packages and supervised the project. T.M. and D.B. performed the benchmark. T.M., D.B. and F.L. wrote the article. All authors read and approved the final manuscript.

Funding

The software was developed as part of the GeDNA project, funded by the German Federal Environment Agency [Umweltbundesamt FKZ 3719242040] and a project funded by Deutsche Forschungsgemeinschaft [LE2323/9-1].

Conflict of Interest: none declared.

Data availability

Data for the benchmark is available in the supplementary files. APSCALE is open source and free for all use, including commercial use. It is licensed under MIT license. The source code is available at https://github.com/DominikBuchner/apscale and https://github.com/TillMacher/apscale_gui.

References

Bolyen,E. *et al.* (2019) Reproducible, interactive, scalable and extensible microbiome data science using QIIME 2. *Nat. Biotechnol.*, **37**, 852–857.

Buchner,D. and Leese,F. (2020) BOLDigger – a python package to identify and organise sequences with the barcode of life data systems. *MBMG*, **4**, e53535.

Buchner,D. *et al.* (2021) Standardized high-throughput biomonitoring using DNA metabarcoding: strategies for the adoption of automated liquid handlers. *Environ. Sci. Ecotechnol.*, **8**, 100122.

Callahan,B.J. *et al.* (2016) DADA2: high-resolution sample inference from illumina amplicon data. *Nat. Methods.*, **13**, 581–583.

Camacho,C. *et al.* (2009) BLAST: architecture and applications. BMC Bioinformatics, **10**, 421.

Compson,Z.G. *et al.* (2020) Metabarcoding from microbes to mammals: comprehensive bioassessment on a global scale. *Front. Ecol. Evol.*, **8**, 581835. https://doi.org/10.3389/fevo.2020.581835.

Dufresne,Y. *et al.* (2019) SLIM: a flexible web application for the reproducible processing of environmental DNA metabarcoding data. *BMC Bioinformatics*, **20**, 88.

Edgar,R.C. (2016) UNOISE2: improved error-correction for illumina 16S AND ITS amplicon sequencing. *bioRxiv*, https://doi.org/10.1101/081257.

Edgar,R.C. (2010) Search and clustering orders of magnitude faster than BLAST. *Bioinformatics*, **26**, 2460–2461.

Frøslev,T.G. *et al.* (2017) Algorithm for post-clustering curation of DNA amplicon data yields reliable biodiversity estimates. *Nat. Commun.*, **8**, 1188.

Macher,T.-H. *et al.* (2021) TaxonTableTools: a comprehensive, platform-independent graphical user interface software to explore and visualise DNA metabarcoding data. *Mol. Ecol. Resour.*, **21**, 1705–1714.

Martin,M. (2011) Cutadapt removes adapter sequences from high-throughput sequencing reads. *EMBnet.J.*, **17**, 10.

Rognes,T. *et al.* (2016) VSEARCH: a versatile open source tool for metagenomics. *PeerJ*, **4**, e2584.