DR. MAHSA MOUSAVI-DERAZMAHALLEH (Orcid ID : 0000-0002-2299-2050)

DR. MICHAEL BUNCE (Orcid ID : 0000-0002-7721-4790)

Article type : Resource Article

Title

eDNAFlow, an automated, reproducible and scalable workflow for analysis of environmental DNA (eDNA) sequences exploiting Nextflow and Singularity

Authors' list

Mahsa Mousavi-Derazmahalleh[1,2,3*], Audrey Stott[4], Rose Lines[1,2,3], Georgia Peverley[3], Georgia Nester[1], Tiffany Simpson[1,3], Michal Zawierta[5], Marco De La Pierre[4], Michael Bunce[1, 6], Claus T. Christophersen[1,2,7]

Affiliations

[1] Trace and Environmental DNA (TrEnD) Laboratory, School of Molecular and Life Sciences, Curtin University, Bentley, WA, 6102, Australia

[2] WA Human Microbiome Collaboration Centre, School of Molecular and Life Sciences, Curtin University, Bentley, WA, 6102, Australia

[3] eDNA frontiers, School of Molecular and Life Sciences, Curtin University, Bentley, WA, 6102, Australia

[4] Pawsey Supercomputing Centre, Kensington, WA, 6151, Australia

[5] Department of Electrical, Electronic and Computer Engineering, The University of Western Australia, Nedlands, WA 6009, Australia

[6] Environmental Protection Authority, 215 Lambton Quay, Wellington 6011, New Zealand

This article has been accepted for publication and undergone full peer review but has not been through the copyediting, typesetting, pagination and proofreading process, which may lead to differences between this version and the Version of Record. Please cite this article as doi: 10.1111/1755-0998.13356

[7] School of Medical & Health Sciences, Edith Cowan University, Joondalup, WA, 6027, Australia

Correspondence

*Mahsa Mousavi-Derazmahalleh

s.mousavid@curtin.edu.au

Running Title

Analysing environmental DNA data using eDNAFlow

Abstract

Metabarcoding of Environmental DNA (eDNA) when coupled with high throughput sequencing is revolutionising the way biodiversity can be monitored across a wide range of applications. However, the large number of tools deployed in downstream bioinformatic analyses often places a challenge in configuration and maintenance of a workflow, and consequently limits the research reproducibility. Furthermore, scalability needs to be considered to handle the growing amount of data due to increase in sequence output and the scale of project.

Here, we describe eDNAFlow, a fully automated workflow that employs a number of state-of-the-art applications to process eDNA data from raw sequences (single-end or paired-end) to generation of curated and non-curated zero-radius operational taxonomic units (ZOTUs) and their abundance tables. This pipeline is based on Nextflow and Singularity which enable a scalable, portable and reproducible workflow using software containers on a local computer, clouds and high-performance computing (HPC)

clusters. Finally, we present an in-house Python script to assign taxonomy to ZOTUs based on user specified thresholds for assigning Lowest Common Ancestor (LCA).

We demonstrate the utility and efficiency of the pipeline using an example of a published coral diversity biomonitoring study. Our results were congruent with the aforementioned study. The scalability of the pipeline is also demonstrated through analysis of a large data set containing 154 samples.

To our knowledge, this is the first automated bioinformatic pipeline for eDNA analysis using two powerful tools: Nextflow and Singularity. This pipeline addresses two major challenges in the analysis of eDNA data; scalability and reproducibility.

Key-words: eDNA, metabarcoding, Nextflow, Singularity

Introduction

Advances in high throughput sequencing (HTS) technologies accompanied by a drop in their cost, has provided an unprecedented opportunity for complementing conventional methods of biomonitoring with environmental DNA (eDNA) metabarcoding approaches (Rees, Maddison, Middleditch, Patmore, & Gough, 2014). eDNA metabarcoding methods have successfully been applied in biomonitoring and assessing the biodiversity of various ecosystems (Bista et al., 2017; Chain, Brown, MacIsaac, & Cristescu, 2016; Dejean et al., 2012; Stat et al., 2017; Valentini et al., 2016). Metabarcoding is a technology that allows rapid, simultaneous identification of multiple taxa from bulk environmental samples such as soil or water (Ficetola, Miaud, Pompanon, & Taberlet, 2008; Taberlet, Coissac, Hajibabaei, & Rieseberg, 2012) without the need to capture or morphologically identify individual organisms. DNA traces left in the environment through sloughed skin, mucus, faeces etc (Taberlet, Bonin, Zinger, & Coissac, 2018) are extracted, followed by PCR amplification and sequencing of target barcode regions (Ficetola et al., 2008). During PCR amplification, samples are labelled with individual index tags called multiplex identifier tags (MID-tags) which allows preparation of a multiplexed library where samples are sequenced in parallel (Sickel et al., 2015). Data generated from multiplexed libraries is processed using a bioinformatic pipeline to determine taxonomic assignments related to biological samples.

Despite the numerous benefits of molecular techniques, such as being non-invasive, non-destructive and cost effective, subsequent downstream bioinformatic analysis presents a variety of challenges (Deiner et al., 2017; Zinger et al., 2019). Multiple bioinformatics tools/resources have been developed and optimised to assist with metabarcoding data analysis including; demultiplexing, quality filtering, chimera checking,

operational taxonomic unit (OTU) clustering and taxonomic assignment (Andrews, 2010; Bolyen et al., 2018; Callahan et al., 2016; Edgar, 2016; Rognes, Flouri, Nichols, Quince, & Mahé, 2016). Yet, the abundance of such tools required for the analysis can be daunting to install and combine into different steps of the workflow. In addition, updates and changes in dependencies can make tool maintenance challenging over the long term and consequently could limit research reproducibility. The growing amount of data produced by sequencing platforms has increased the demand for easily scalable workflows to take advantage of powerful computing resources including cloud and HPC cluster infrastructure (Porter & Hajibabaei, 2018; Taberlet, Coissac, Pompanon, Brochmann, & Willerslev, 2012; Zinger et al., 2019) and we set out to address some of these bioinformatic challenges with a resulting pipeline; eDNAflow.

There have been a few efforts to develop metabarcoding analysis pipelines, such as FACEPAI, Anacapa and SLIM (Curd et al., 2019; Dufresne, Lejzerowicz, Perret-Gentil, Pawlowski, & Cordier, 2019; Wahlberg, 2019). FACEPAI (Fast And Consistent Environmental DNA Processing AND Identification), an open source script provides a pipeline for analysis of paired-end sequences (Wahlberg, 2019), but the script is dependent on a number of bioinformatic tools that must be installed manually, and does not perform demultiplexing and/or post clustering curation. The Anacapa toolkit combines various software and can create reference libraries and generate custom reference databases, infer Amplicon Sequence Variants (ASVs) and assign taxonomy (Curd et al., 2019). Similar to FACEPAI, Anacapa also does not offer demultiplexing and post clustering curation steps, and can only accept paired-end sequences. SLIM, a web-based application was developed for user-friendly execution of eDNA metabarcoding analysis through a Graphic User Interact (GUI) (Dufresne et al., 2019). Even though both of the latter workflow approaches make use of containers, which are all-encompassing and ready-to-run software package bundles that are easily reproduced, deployed, and version-controlled, their pipelines still come with drawbacks that can impede efficiency.

One of the main challenges for many of the existing metabarcoding pipelines is scalability – it is often impractical to apply them on large datasets. Within the Anacapa toolkit, deployment is possible on various computing platforms, but adjusting parameters for each module to optimise use of computing resources is not demonstrated, such as for an HPC cluster. While SLIM incorporates containers for each module, they rely on the Docker container engine for building images, which is not permitted on HPC clusters. Furthermore, although a graphical user interface provides a favourable solution for non-experts, some computing knowledge such as the installation of Docker and use of the command line is still required for running scripts to launch the webserver.

Finally, none of the three workflows above support task execution caching which allows tasks which are completed successfully to be omitted and cached results to be deployed for downstream processes. A summary of these pipelines' characteristics is provided in Table 1.

The benefits, relevance and increasing need of employing workflow engines in bioinformatic pipelines have been raised in recent years (Perkel, 2019). Nextflow is a state-of-the-art workflow creation tool that seamlessly enables workload scalability (Di Tommaso et al., 2017). This is achieved within the same platform through its unified parallelism design, and cross-platform functionality by providing simple interfaces to a variety of schedulers and orchestrators, allowing workflow portability from local resources, to site workstations, cloud services and HPC clusters (Di Tommaso et al., 2017). Nextflow also ensures reproducibility by enabling container build on both HPC and non-HPC compute infrastructure through the integration of Singularity, which is a container engine that has been developed purposefully to facilitate HPC deployment (Kurtzer, Sochat, & Bauer, 2017). Nextflow also provides checkpoint-restart capabilities to resume execution from the last successful step.

Here, we thus use the aforementioned contemporary technologies, Nextflow and Singularity, to develop eDNAFlow, a reproducible and easily scalable bioinformatic workflow for analysing eDNA metabarcoding data, capable of using both single-end and paired-end sequence data. We use two marine environmental DNA studies as models to test the pipeline and demonstrate the scalability and the reproducibility of taxonomy results.

Workflow description

This workflow is a suite of applications and methods for defining zero-radius operational taxonomic units (ZOTUs) (Edgar, 2016). It performs the following tasks: Quality checking of raw single-end or paired-end sequences using FastQC (Andrews, 2010), quality filtering (merge if paired-end) using AdapterRemoval (Schubert, Lindgreen, & Orlando, 2016), followed by another round of FastQC quality checking, then demultiplexing with OBITools (Boyer et al., 2016) and removal of sequences smaller than a user specified minimum length. The indexing method used in the laboratory workflow will not affect the demultiplexing step, provided that the user is able to prepare a file detailing the indexes/tags related to each sample in the OBITools required format. Mismatches are not allowed in the index/tag sequences for demultiplexing. The resulting split sample files are then changed to a USEARCH (Edgar, 2016) suitable format by relabelling sequence headers. The pipeline also allows the user to skip the quality and demultiplexing steps by directly inputting demultiplexed data. The USEARCH unoise3 algorithm is employed to perform dereplication, create ZOTUs (default minimum abundance is 8, but can be adjusted by user for higher

sensitivity; see eDNAFlow GitHub page for details of how to adjust parameters), and make the ZOTU table. The ZOTU sequences are then queried against a local nucleotide database from GenBank (NCBI) using BLASTN and/or a custom database (Altschul, Gish, Miller, Myers, & Lipman, 1990). Finally LULU, a post clustering curation method, is applied for removal of spurious operational taxonomic units that are identified by relating sequence similarity and co-occurrence patterns (Frøslev et al., 2017). Once the first part of the analysis is finished, the user can run eDNAFlow to assign taxonomy to curated or un-curated ZOTUs. The script supporting this part is described in the next section: "Assigning ZOTUs to lowest common ancestor (LCA)". A schematic outline of these tasks is illustrated in Fig 1.

All parameters of different tasks and platforms are set from adaptable configuration files and can be adjusted via setting parameters at command line. For instance, while the default minimum abundance size of unoise3 is eight sequence reads to form a ZOTU cluster, a user may choose to reduce the size to four for higher sensitivity. Replacing the default settings of the configuration file should be achieved by adding the relevant parameters' flag when running the script (e.g. nextflow run eDNAFlow.nf --reads 'test_30000reads.fastq' --barcode 'se_bc*' --blast_db 'Path2TestBlastDataset/db' --minsize '4'). Similarly, if for example a user has access to computing resources with higher memory, then they can run eDNAFlow with setting '-profile nimbus'.

To see a detailed description of how the pipeline can be run and how the parameters and/or configuration file can be adjusted see https://github.com/mahsa-mousavi/eDNAFlow.

Assigning ZOTUs to lowest common ancestor (LCA)

A custom Python script was created to filter multiple taxonomic assignments (hits) from the BLAST results, download the latest taxonomy information from NCBI website, assign ZOTUs to their lowest common ancestor (LCA), and link this information to the ZOTU abundance table. The filtering applied in this script is based on a set of user specified thresholds, including query coverage (qCov), percentage identity (% identity) and the difference (Diff) between % identities of two hits when their qCov is equal. Setting qCov and % identity thresholds ensures that only BLAST hits >= to those thresholds will progress to the Diff comparison step. Setting Diff means that if the absolute value for the difference between % identity of hit1 and hit2 is > Diff, then a species level taxonomy will be returned, otherwise taxonomy of that ZOTU will be dropped to the lowest common ancestor. This script produces two files, a file in which the taxonomy is assigned to LCA (the final result), and an intermediate file which includes the blast result linked with taxonomy information, where sequences have passed initial filtering thresholds (i.e. qCov and %identity), but have not yet been compared for LCA assignment nor have been linked with the abundance

table. This file will give the user an idea of why some ZOTUs may have been assigned to the lowest common ancestor. For instance, if user was expecting a particular species which was not found in their final LCA result, they may be able to find it in the intermediate file and check what other close blast hits existed that caused the final assignment to drop to LCA level.

Case study 1 - Accuracy

To demonstrate the relevance of the pipeline to a real-world study and the influence of user-defined classification parameters, we analysed the dataset reported by Alexander et al. (2020). This dataset is comprised of 90 surface seawater samples, three negative controls (bleach and tap water) and a positive control tested using the CoralITS2 assay (paired-end) primers (Brian, Davy, & Wilkinson, 2019). The Alexander et al. (2020) study provided an opportunity to validate the result of our pipeline as it reported traditional diver-based visual survey results and eDNA data that was produced using different bioinformatic tools: the R package Insect (Wilkinson, Davy, Bunce, & Stat, 2018) for demultiplexing, DADA2 (Callahan et al., 2016) for detection of amplicon sequence variance (ASV), and taxonomy assignment using a 95% threshold with MEtaGenome Analyser (MEGAN) (Huson et al., 2016).

eDNAFlow was run on a high-throughput HPC Linux cluster called Zeus, at the Pawsey Supercomputing Centre. A general purpose work queue node on Zeus allows jobs to run up to 24 hours on 28 CPUs and 128GB of RAM. Multiple nodes can be used for each job to make use of more CPUs and RAM, and/or to parallelise workflows.

Analysing the raw sequence file that was published by Alexander et al. (2020), eDNAFlow generated 7,466 ZOTUs, out of which 4,103 ZOTUs remained after post-clustering curation with LULU. Taxonomies were assigned by running the LCA script on the curated ZOTU table specifying a range of taxonomy classification parameters: 1) qCov 100, %id 95, Diff 0.5; 2) qCov 100, %id 95, Diff 1; 3) qCov 98, %id 93, Diff 0.5. To compare and provide an accuracy estimation of our classification results, we used the heat map of the CKI Scleractinia visual survey and molecular OTUs detected per genus as demonstrated in Alexander et al. (2020). We also conducted an *in silico* analysis of all taxa reported in the heat map that were not detected by the CoralITS2 assay by Alexander et al. (2020) to determine whether there are available reference sequences for these taxa and that the assay is theoretically capable of detecting them.

In general, our taxonomy results were highly congruent with Alexander et al. (2020). Some differences could be attributed to altered classification parameters. Using the first setting parameters listed above, we were able to detect *Lithophyllon* and *Echinopora* genera in seawater samples, both of which were

recorded in their visual survey. These genera were not found by Alexander et al. (2020), with the exception of one *Echinopora* OTU found in high abundance in their control sample and subsequently removed. In contrast to Alexander et al. (2020) we did not find the genus *Goniastrea* in our final LCA result, though it was detected in our intermediate file. That is because one of the representing ZOTUs was removed after LULU curation and the other returned the first hit among the two (%id 96.98 for *Favia stelligera* versus 96.25 for *Goniastrea stelligera*) as Diff was set to > 0.5.

When the second parameter thresholds were applied, *Lithophyllon* did not appear in our final result, because multiple ZOTUs had multiple blast hits, but all of them had %id difference of below 1 (i.e. Diff 1). Consequently, the taxonomy assignment was dropped to the lowest common ancestor as it no longer could be resolved at genus level with Diff 1. Also, we did not find *Oxypora* or *Lobophyllia* with either setting 1 or 2, because the ZOTU representing *Oxypora* had 2 very close hits - i.e. %id of 98.944 for *Oxypora lacera* and 98.947 for *Echinophyllia echinate*. These could not be further resolved at either Diff 1 or Diff 0.5 and so the taxonomy was dropped to the LCA. For *Lobophyllia*, the representing ZOTU was absent after LULU curation. Furthermore, despite visual support for the genera *Favites* and *Cyphastrea*, they were not detected by Alexander et al. (2020) or eDNAFlow until qCov threshold was relaxed from 100 to 98 using the third parameter settings.

To further benchmark eDNAFlow, we conducted additional analysis of the Alexander et al. (2020) dataset using SLIM (Dufresne et al., 2019). The raw sequences were demultiplexed using Double Tag Demultiplexing module of SLIM. The demultiplexed files were analysed by module DADA2 (Callahan et al., 2016) for ASV inferences with the following setting: making an error model for each sample. This generated a total of 6,875 ASVs. We used these ASV sequences to query against the GenBank nucleotide database. The eDNAFlow LCA script with setting 1 was used for taxonomy assignment.

The comparison of these results with those already discussed above shows a high similarity among all three methods with some differences (Fig 2). For instance, while both Alexander et al. (2020) and eDNAFlow identified the genera *Tubastraea* and *Anacropora,* neither of these were found by SLIM. The genera *Danafungia* and *Favites* were detected in SLIM results, but were only seen in the intermediate result file with eDNAFlow and were missed in Alexander et al. (2020).

The observed differences could be ascribed to utilization of different algorithms or choosing different parameters of the same algorithm (e.g. differences between Alexander et al (2020) vs SLIM, both of which have used DADA2).

Our *in silico* CoralITS2 analysis for 35 genera listed in the heat map presented in Alexander et al. (2020) revealed that the CoralITS2 assay is theoretically capable of detecting 26 of those genera, out of which 19 were also reported by the visual survey. The comparison of the *in silico* analysis, visual survey filtered by *in silico* outcomes, and the results of three pipeline analyses is provided in Table 2. All of the intermediate, final LCA taxonomy results and comparisons can be found in Table S1.

Based on these findings, we believe that eDNAflow provides a high level of accuracy in data filtering and taxonomic assignment. However, we note that the LCA parameter settings should be adjusted according to the research question and how stringent the user wants the taxonomy assignments to be. We suggest it is worthwhile to run the LCA script with different settings to inform researchers of the impact of these parameters.

Case study 2 - Scalability

To demonstrate the scalability of our pipeline, we combined sequence results of single-end libraries generated from seawater samples collected from the east and west coasts of Australia. A total of 154 samples (including negative controls) were analysed with the 16S_FishSyn_Short (Nester et al., 2019) assay from multiple single-end libraries (a total of 49Gb of raw sequences). There were nine Multiplex Identifier (MID) tag files used in the demultiplexing step. Laboratory preparation details are provided in Appendix S1. eDNAFlow was successfully run on the Zeus long queue node, which allows jobs to run for up to 4 days, generating 3,156 ZOTUs, which were filtered to 2,841 ZOTUs after curation with LULU. The curated result was then used for assigning taxonomy with the LCA script using the same parameters as setting one in Case study 1 only to demonstrate the scalability of the script. ZOTUs were assigned to 411 taxa, out of which 20 were assigned to class level, 76 to order level, 150 to family level, 230 to genus level and 242 to species level. A list of identified taxa is provided in Table S2.

All scripts and parameters used to run eDNAFlow on these single-end and paired-end datasets can be accessed at the following GitHub page: https://github.com/mahsa-mousavi/eDNAFlow.

Future direction

Currently eDNAFlow supports using both USEARCH 32-bit (by default) and USEARCH 64-bit (if a path to its executable is provided). We acknowledge the memory limitation of USEARCH 32-bit version. Even though in our experience and as shown per our “Case study” sections, USEARCH 32-bit works very well with

moderate to large datasets, for very large datasets the 64-bit version is preferred (Edgar, 2020). Therefore, in future we aim to add the DADA2 workflow to this pipeline, to both address the aforementioned limitation and give users more flexibility in choosing the classifier to identify zero-radius operational taxonomic units and/or ASVs.

Conclusions

The rate at which metabarcoding workflows and associated tools are growing places a challenge on researchers to generate high fidelity and reproducible results which are easily scalable. Along with two effective tools, Nextflow and Singularity, eDNAFlow enables processing of eDNA metabarcoding datasets with a pipeline that is customizable, scalable, portable and reproducible. Additionally, thanks to Nextflow's caching mechanism, eDNAFlow allows for task execution caching, meaning, upon resuming the pipeline, tasks which are completed successfully are skipped and cached results are used for downstream processes. This minimizes the repetition of efforts and enables users to easily test how choosing different parameters affects the sensitivity and fidelity of their results. We believe this workflow is a step forward to generate reproducible, comparative and transparent eDNA metabarcoding analysis.

Acknowledgment

This work was supported by resources provided by the Pawsey Supercomputing Centre with funding from the Australian Government and the Government of Western Australia. We would like to thank Dr Tina Berry for her comments on the manuscript. The authors declare no conflict of interest.

Authors' Contributions

MMD conceived the ideas and designed eDNAFlow; MMD and AS benchmarked the workflow; RL and GP performed lab works; MMD and GN made LULU container; AS and MDLP debugged the workflow; MMD designed the LCA script; MZ debugged the LCA script; MMD led the writing of the manuscript; RL wrote the lab work section; TS performed the *in silico* test; MMD, RL, AS, MDLP, TS, MB and CC designed the laboratory workflow, wrote the manuscript and provided critical feedback. All authors contributed critically to the drafts and gave final approval for publication.

DATA Accessibility

The eDNAFlow is available from GitHub (https://github.com/mahsa-mousavi/eDNAFlow).

References

Alexander, J. B., Bunce, M., White, N., Wilkinson, S. P., Adam, A. A., Berry, T., . . . Dugal, L. (2020). Development of a multi-assay approach for monitoring coral diversity using eDNA metabarcoding. *Coral Reefs, 39*(1), 159-171.

Altschul, S. F., Gish, W., Miller, W., Myers, E. W., & Lipman, D. J. (1990). Basic local alignment search tool. *Journal of Molecular Biology, 215*(3), 403-410.

Andrews, S. (2010). FastQC: a quality control tool for high throughput sequence data. In: Babraham Bioinformatics, Babraham Institute, Cambridge, United Kingdom.

Bista, I., Carvalho, G. R., Walsh, K., Seymour, M., Hajibabaei, M., Lallias, D., . . . Creer, S. (2017). Annual time-series analysis of aqueous eDNA reveals ecologically relevant dynamics of lake ecosystem biodiversity. *Nature Communications, 8*, 14087.

Bolyen, E., Rideout, J. R., Dillon, M. R., Bokulich, N. A., Abnet, C., Al-Ghalith, G. A., . . . Asnicar, F. (2018). *QIIME 2: Reproducible, interactive, scalable, and extensible microbiome data science* (2167-9843). Retrieved from

Boyer, F., Mercier, C., Bonin, A., Le Bras, Y., Taberlet, P., & Coissac, E. (2016). obitools: A unix-inspired software package for DNA metabarcoding. *Molecular Ecology Resources, 16*(1), 176-182.

Brian, J. I., Davy, S. K., & Wilkinson, S. P. (2019). Elevated Symbiodiniaceae richness at Atauro Island (Timor-Leste): a highly biodiverse reef system. *Coral Reefs, 38*(1), 123-136.

Callahan, B. J., McMurdie, P. J., Rosen, M. J., Han, A. W., Johnson, A. J. A., & Holmes, S. P. (2016). DADA2: high-resolution sample inference from Illumina amplicon data. *Nature Methods, 13*(7), 581.

Chain, F. J., Brown, E. A., MacIsaac, H. J., & Cristescu, M. E. (2016). Metabarcoding reveals strong spatial structure and temporal turnover of zooplankton communities among marine and freshwater ports. *Diversity and Distributions, 22*(5), 493-504.

Curd, E. E., Gold, Z., Kandlikar, G. S., Gomer, J., Ogden, M., O'Connell, T., . . . Lin, M. (2019). Anacapa Toolkit: an environmental DNA toolkit for processing multilocus metabarcode datasets. *Methods in Ecology and Evolution, 10*(9), 1469-1475.

Deiner, K., Bik, H. M., Mächler, E., Seymour, M., Lacoursière-Roussel, A., Altermatt, F., . . . De Vere, N. (2017). Environmental DNA metabarcoding: Transforming how we survey animal and plant communities. *Molecular Ecology, 26*(21), 5872-5895.

Dejean, T., Valentini, A., Miquel, C., Taberlet, P., Bellemain, E., & Miaud, C. (2012). Improved detection of an alien invasive species through environmental DNA barcoding: the example of the American bullfrog *Lithobates catesbeianus*. *Journal of Applied Ecology, 49*(4), 953-959.

Di Tommaso, P., Chatzou, M., Floden, E. W., Barja, P. P., Palumbo, E., & Notredame, C. (2017). Nextflow enables reproducible computational workflows. *Nature Biotechnology, 35*(4), 316-319.

Dufresne, Y., Lejzerowicz, F., Perret-Gentil, L. A., Pawlowski, J., & Cordier, T. (2019). SLIM: a flexible web application for the reproducible processing of environmental DNA metabarcoding data. *BMC Bioinformatics, 20*(1), 1-6.

Edgar, R. C. (2016). UNOISE2: improved error-correction for Illumina 16S and ITS amplicon sequencing. *bioRxiv*.

Edgar, R. C. (2020). USEARCH download. Retrieved from https://drive5.com/usearch/download.html

Ficetola, G. F., Miaud, C., Pompanon, F., & Taberlet, P. (2008). Species detection using environmental DNA from water samples. *Biology letters, 4*(4), 423-425.

Frøslev, T. G., Kjøller, R., Bruun, H. H., Ejrnæs, R., Brunbjerg, A. K., Pietroni, C., & Hansen, A. J. (2017). Algorithm for post-clustering curation of DNA amplicon data yields reliable biodiversity estimates. *Nature Communications, 8*(1), 1188.

Huson, D. H., Beier, S., Flade, I., Górska, A., El-Hadidi, M., Mitra, S., . . . Tappu, R. (2016). MEGAN community edition-interactive exploration and analysis of large-scale microbiome sequencing data. *PLoS computational biology, 12*(6), e1004957.

Kurtzer, G. M., Sochat, V., & Bauer, M. W. (2017). Singularity: Scientific containers for mobility of compute. *PLoS One, 12*(5), e0177459.

Nester, G. M., De Brauwer, M., Koziol, A., West, K. M., DiBattista, J. D., White, N. E., . . . Bunce, M. (2019). Development and evaluation of fish eDNA metabarcoding assays facilitate the detection of cryptic seahorse taxa (family: Syngnathidae). *Environmental DNA*.

Perkel, J. M. (2019). Workflow systems turn raw data into scientific knowledge. *Nature, 573*(7772), 149-150.

Porter, T. M., & Hajibabaei, M. (2018). Scaling up: A guide to high-throughput genomic approaches for biodiversity analysis. *Molecular Ecology, 27*(2), 313-338.

Rees, H. C., Maddison, B. C., Middleditch, D. J., Patmore, J. R., & Gough, K. C. (2014). The detection of aquatic animal species using environmental DNA–a review of eDNA as a survey tool in ecology. *Journal of Applied Ecology, 51*(5), 1450-1459.

Rognes, T., Flouri, T., Nichols, B., Quince, C., & Mahé, F. (2016). VSEARCH: a versatile open source tool for metagenomics. *PeerJ, 4*, e2584.

Schubert, M., Lindgreen, S., & Orlando, L. (2016). AdapterRemoval v2: rapid adapter trimming, identification, and read merging. *BMC Research Notes, 9*(1), 88.

Sickel, W., Ankenbrand, M. J., Grimmer, G., Holzschuh, A., Härtel, S., Lanzen, J., . . . Keller, A. (2015). Increased efficiency in identifying mixed pollen samples by meta-barcoding with a dual-indexing approach. *BMC Ecology, 15*(1), 1-9.

Stat, M., Huggett, M. J., Bernasconi, R., DiBattista, J. D., Berry, T. E., Newman, S. J., . . . Bunce, M. (2017). Ecosystem biomonitoring with eDNA: metabarcoding across the tree of life in a tropical marine environment. *Scientific Reports, 7*(1), 1-11.

Taberlet, P., Bonin, A., Zinger, L., & Coissac, E. (2018). *Environmental DNA: For biodiversity research and monitoring*: Oxford University Press.

Taberlet, P., Coissac, E., Hajibabaei, M., & Rieseberg, L. H. (2012). Environmental dna. In: Wiley Online Library.

Taberlet, P., Coissac, E., Pompanon, F., Brochmann, C., & Willerslev, E. (2012). Towards next-generation biodiversity assessment using DNA metabarcoding. *Molecular Ecology, 21*(8), 2045-2050.

Valentini, A., Taberlet, P., Miaud, C., Civade, R., Herder, J., Thomsen, P. F., . . . Boyer, F. (2016). Next-generation monitoring of aquatic biodiversity using environmental DNA metabarcoding. *Molecular Ecology, 25*(4), 929-942.

Wahlberg, E. (2019). FACEPAI: a script for fast and consistent environmental DNA processing and identification. *BMC Ecology, 19*(1), 1-6.

Wilkinson, S. P., Davy, S. K., Bunce, M., & Stat, M. (2018). Taxonomic identification of environmental DNA with informatic sequence classification trees. *PeerJ*.

Zinger, L., Bonin, A., Alsos, I. G., Bálint, M., Bik, H., Boyer, F., . . . Deagle, B. E. (2019). DNA metabarcoding—Need for robust experimental designs to draw sound ecological conclusions. *Molecular Ecology, 28*(8), 1857-1862.

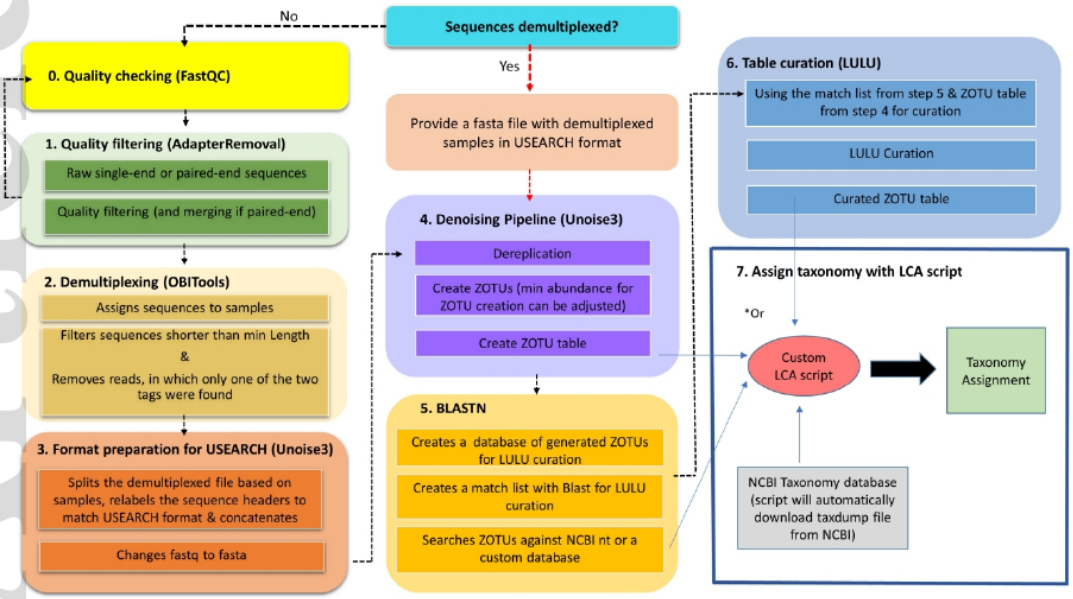

Fig 1. Flowchart of the eDNAFlow pipeline. * Non-curated or curated ZOTU tables can be used with the script.

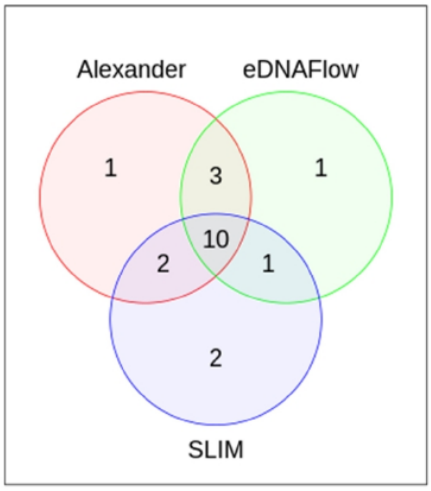

Fig 2. Venn diagram depicting number of genera from the Scleractinia order detected using Alexander et al. 2020, SLIM and eDNAFlow methods

Table 1. Summ
eDNAFlow and
workflows

	eDNAFlow	FACEPAI	Anacapa	SLIM
Reproducibility	Dependencies are automatically installed by Singularity container management applied in the workflow. Workflow configuration files are recorded and reusable. Each execution automatically creates various log files that records the run details and can be rerun or used for debugging.	No container management is used; correct dependency version of tools must be installed.	Anacapa and its dependencies must be installed, or it can be downloaded in a Singularity container.	A bash script fetches the application dependencies. Each execution can be saved and stored as a configuration file.
Flexibility	Workflow parameters and the choice of computational platforms can be easily specified by adding the relevant parameters' flag when running the script. If needed, modification of computational resources can be achieved by adjusting the configuration file.	Only a few options are adjustable through a config file, while the majority have been hard-coded in the facepai.sh script. Not easily expandable to other computational platforms.	Being a modular tool makes it flexible in choosing run parameters; however setting it up in various computational platforms and adjusting computational resources will be more difficult.	A module-centered application makes it flexible in choosing run parameters; however setting it up in various computational platforms and adjusting computational resources will be more difficult.
Scalability	A config file sets the computational resources necessary for different platforms. It enables the workflow to be run on local machine, cloud and HPC. If run on cloud or HPC, minimal changes in the configuration file may be necessary depending on users' cluster/cloud infrastructure.	Not available	It can be downloaded in a Singularity container and can be run on HPC cluster; however it doesn't allow easy configuration of computational resources.	The default settings are not suitable for scaling to bigger datasets. More knowledge on changing the default settings is required through Docker configurations. Scaling to HPC resources is not directly possible.
Interface Type	Command line	Command line	Command line	Web-based (with some command line)
Workflow versioning	Yes	No	No	Yes
Container management	It can manage multi scale containers.	Not available	It can be downloaded in a Singularity container.	It uses a docker container.
Task execution caching	Yes	No	No	No
Automated job parallelization	Yes	No	In case of using a cluster, RcppParallel has to be installed.	Yes
Read types	Single-end & paired-end	Paired-end	Paired-end	Single-end & paired-end
Demultiplexing	Yes	No	No	Yes
Post-clustering curation	Yes	No	No	Yes

Table 2. Number of Scleractinia genera detected using the CoralITS2 assay from Alexander et al. (2020), SLIM analysis and the eDNAFlow intermediate and Lowest Com determined using different stringency parameters compared with *In silico* analysis of genera listed in that study heatmap

			SLIM analysis (Un-curated)		eDNAFlow analysis (LULU curated)				
	Alexander et al., (2020)		Parameter setting 1		parameter settings 1		parameter settings 2		p
***In silico* CoralITS2 analysis – genera listed in heatmap**	**Visual survey (filtered by *in silico* results)**	**CoralITS2 eDNA results**	**Intermediate classification** **qCov100 id95 Diff0.5**	**LCA classification** **qCov100 id95 Diff0.5**	**Intermediate classification** **qCov100 id95 Diff0.5**	**LCA classification** **qCov100 id95 Diff0.5**	**Intermediate classification** **qCov100 id95 Diff1**	**LCA classification** **qCov100 id95 Diff1**	**Interm classi** **qC id Di**
26	19	16	17	15	22	15	22	14	

Legends of the Supplementary Tables

Table S1. The list of intermediate, final identified taxa and comparison of different taxonomy classification parameters in case study 1

Table S2. The list of intermediate and final identified taxa in case study 2