UCEasy: A software package for automating and simplifying the analysis of ultraconserved elements (UCEs)

Caio V. R. Ribeiro[‡], Lucas P. Oliveira[§], Romina Batista[|,¶], Marcos De Sousa[#,‡]

‡ Coordenação de Ciência da Computação, Centro Universitário do Estado do Pará (CESUPA), Belém, Brazil
§ Instituto de Computação, Universidade Estadual de Campinas (UNICAMP), Campinas, Brazil
| Instituto Nacional de Pesquisas da Amazônia (INPA), Manaus, Brazil
¶ Gothenburg Global Biodiversity Centre, Gothenburg, Sweden
# Museu Paraense Emílio Goeldi (MPEG), Belém, Brazil

Corresponding author: Marcos De Sousa (msousa@museu-goeldi.br)

Academic editor: Anne Ropiquet

Abstract

Background

The use of Ultraconserved Elements (UCEs) as genetic markers in phylogenomics has become popular and has provided promising results. Although UCE data can be easily obtained from targeted enriched sequencing, the protocol for *in silico* analysis of UCEs consist of the execution of heterogeneous and complex tools, a challenge for scientists without training in bioinformatics. Developing tools with the adoption of best practices in research software can lessen this problem by improving the execution of computational experiments, thus promoting better reproducibility.

New information

We present UCEasy, an easy-to-install and easy-to-use software package with a simple command line interface that facilitates the computational analysis of UCEs from sequencing samples, following the best practices of research software. UCEasy is a wrapper that standardises, automates and simplifies the quality control of raw reads, assembly and extraction and alignment of UCEs, generating at the end a data matrix with different levels of completeness that can be used to infer phylogenetic trees. We demonstrate the functionalities of UCEasy by reproducing the published results of phylogenomic studies of the bird genus *Turdus* (Aves) and of Adephaga families (Coleoptera) containing genomic datasets to efficiently extract UCEs.

Keywords

phylogenomics, ultraconserved elements (UCEs), research software, reproducibility, bioinformatics.

Introduction

In the last decade, new genome-subsampling methods have been developed as a cheaper and simpler alternative to complete genome sequencing, thus enabling the scientific community to better understand the evolutionary inter-relationships of species (Davey et al. 2011, Andermann et al. 2020). One of these methods concentrates the sequencing effort on sets of pre-selected genetic markers, a reduced set of the genome. Ultraconserved Elements (UCEs) are amongst the most commonly used capture baits to target highly-conserved regions spread across a genome. These genomic regions are strongly conserved in different species throughout multiple evolutionary timescales (Faircloth et al. 2012) and have been efficiently used as molecular markers for phylogenomic studies (Bejerano et al. 2004, Baca et al. 2017, Branstetter et al. 2017).

Although UCE data can be easily obtained from targeted enriched sequencing (McCormack et al. 2011, Faircloth et al. 2012), the protocols for *in silico* analysis of UCEs consists of the execution of many heterogeneous and complex tools. Currently, the most widely used bioinformatics pipeline for processing UCE data is the software package PHYLUCE (Faircloth 2015). Amongst the main tasks performed by this software package are: quality control, assembly and extraction and alignment of UCEs for downstream analysis. Other phylogenomic tools use the PHYLUCE pipeline, such as seqcap_pop (Harvey et al. 2016) to obtain phased alignments of UCEs and MitoFinder (Allio et al. 2020) to extract UCE and mitogenomic data. Although the PHYLUCE pipeline is robust and well-established, it requires the execution of many command line scripts from heterogeneous tools, which can be quite challenging for scientists without sufficient training in bioinformatics.

Magee et al. (2014) showed that 60% of published phylogenetic analyses are not reproducible. Many types of software used in bioinformatics pipelines have been developed without the adoption of best practices in research software (Jiménez et al. 2017). Furthermore, making data and code available does not guarantee reproducibility (Beaulieu-Jones and Greene 2017, Ferenhof and Alves de Sousa 2021). A bioinformatics tool that adheres to recommendations for best practices in research software offers a greater guarantee of computational reproducibility (Sandve et al. 2013, Wilson et al. 2014, Leprevost et al. 2014, Piccolo and Frampton 2016, Jiménez et al. 2017). The main recommendations are: i) Use package managers in order to make installation easier and to ensure that software versions and dependencies are installed correctly. Difficult-to-install software can be frustrating, impact reliability and impair reproducibility (Georgeson et al. 2019); ii) Make your software source code and documentation available in public and indexed repositories through DOI as a way to promote accessibility; iii) Adopt an OSS (Open Source Software) licence, since unlicensed software discourages reusability and collaboration (Georgeson et al. 2019). The use of OSS licences improves accessibility, reusability and transparency and contributes to the reproducibility of the results generated by the software (Jiménez et al. 2017); iv) Use a version control system to maintain the history of changes made to the source code, allowing arbitrary versions to be retrieved and compared, thus providing provenance for the code; v) When it comes to command-line interface (CLI) tools, implement commands with consistent, distinct and meaningful names, besides clear output and error messages (Seemann 2013, Wilson et al. 2014); vi) When a bioinformatics analysis uses a pipeline containing heterogeneous tools, recording the execution progress of each tool in a log file is good practice. The metadata obtained in this manner should contain the commands executed, the generated output and the date and time of occurrence, all of which provide useful information to aid in debugging and provenance of the pipeline execution (Georgeson et al. 2019).

In this work, we present UCEasy, an open source software package that facilitates the analysis of UCEs from sequencing samples, following the best practices of research software. UCEasy is a Python wrapper that standardises, automates and simplifies the following PHYLUCE tasks: quality control of raw reads, assembly, alignment and UCE extraction. We demonstrate the functionalities of UCEasy by reproducing the published results from two phylogenomic studies (Baca et al. 2017, Batista et al. 2020) to efficiently extract UCEs.

Project description

Title: UCEasy

Design description: UCEasy automates and simplifies the analysis of UCE datasets from DNA sequence samples in FASTQ files (either single-ended or paired-ended), interacting with Python scripts adopted in the standard PHYLUCE 1.6 workflow (https://phyluce.readthedocs.io/en/latest/tutorials/tutorial-1.html), as shown in Fig. 1. The UCEasy CLI is organised into three modules: i) Trim: this module is responsible for quality control, removing adapters and low quality reads using the command **uceasy trim**. The user needs to specify the directory that contains the raw sequence files (.fastq) and also the CSV file that contains the barcode of the samples and sequence adapters; ii) Assemble: this module receives the clean reads by the Quality Control and then uses SPAdes (Bankevich et al. 2012) to perform the assembly of the contigs without the use of a reference genome using the command **uceasy assemble**; iii) Align: in this module, the extraction and alignment of UCEs is performed with the command **uceasy align**. The contigs matching UCE with a pre-selected UCE probe-set (https://github.com/faircloth-lab/uce-probe-sets) are identified to create a list of UCEs by sample. This list of UCEs is used to extract UCE contigs from de novo assemblies on a sample-by-sample basis, generating several FASTA files. Then, the data are aligned against all these FASTA files using MAFFT (Katoh and Standley 2013) or MUSCLE (Edgar 2004). The next step is the generation of alignment matrices with different levels of completeness: either all taxa may have data for all UCEs or some UCEs are presented for a certain percentage of taxa. Finally, these data matrices are concatenated to either NEXUS or PHYLIP file format, that can be used to infer phylogenetic trees using programmes such as RAxML or ExaBayes. More details about the UCEasy commands and their parameters, as well as a comparison between CLIs demonstrating the ease of use of UCEasy compared to PHYLUCE, can be found on our wiki (https://github.com/uceasy/uceasy/wiki).

Web location (URIs)

Homepage: https://github.com/uceasy/uceasy

Wiki: https://github.com/uceasy/uceasy/wiki

Technical specification

Platform: Linux

Programming language: Python 3.7

Operational system: GNU/Linux; Hardware requirements (Minimum): 16 GB of RAM, 8 core CPU

Repository

Type: Github

Browse URI: https://github.com/uceasy/uceasy

Usage licence

Usage licence: Other

IP rights notes: MIT Licence

Implementation

Implements specification

UCEasy has an extensible software architecture that makes use of the Facade and Adapter design patterns (Gamma et al. 1994). The Facade pattern aims to provide a unified interface to a set of components in a subsystem, improving its usability. The idea is to add new modules in the future, as different types of UCE analysis emerge. The Adapter pattern is used when a new package needs to be integrated with an existing system, but the new package and the system have different structures without a direct interface. We applied this pattern to integrate UCEasy with PHYLUCE. The Facade and Adapter patterns allow the interaction between UCEasy and PHYLUCE without the need to change the source code of PHYLUCE. The UCEasy software architecture is shown in Fig. 2.

UCEasy was built based on best practices in scientific computing (Wilson et al. 2014) by adopting the following recommendations: a) Package manager: UCEasy uses pip (https://pip.pypa.io) as the package manager, easing the installation process. The UCEasy install command: pip install uceasy; b) Public software repositories: the source code and documentation of UCEasy is available in the public repositories of PyPI (https://pypi.org/project/uceasy) and Github (https://github.com/uceasy/uceasy). In addition, the package is indexed in the Zenodo repository with DOI: 10.5281/zenodo.5225152, in order to make it discoverable, searchable and referable to as many communities as possible; c) Software licence: UCEasy adopts the MIT licence (https://opensource.org/licenses/MIT), which gives full freedom of use, copying, modification and distribution of the source code, thereby enabling reuse and scientific collaboration; d) Version control system: the UCEasy source code and documentation are kept publicly on Github, a repository widely used by the scientific community (Perez-Riverol et al. 2016), allowing other developers to follow and contribute to the evolution of UCEasy; e) CLI standard: UCEasy follows main recommendations for making command lines usable (Seemann 2013, Georgeson et al. 2019), keeping commands associable, consistent and memorable. UCEasy commands adopt the POSIX standard as a way to help the user (Zlotnick 1991); f) UCEasy offers an optional progress-logging facility, using the uceasy --tracking-file (or -t) command. The log file contains the command line used to run the programme, the output generated (e.g. errors, warnings and messages) and the date and time the execution started and ended.

Audience

The target audience for this software package includes evolutionary biologists and conservation scientists with knowledge of basic Linux commands. We are open to discussing additional ideas or new features to expand the current functionality of this software package.

Additional information

Results and conclusion

To demonstrate the effectiveness of UCEasy, we reproduced the published results by Baca et al. (2017) and Batista et al. (2020) to extract UCE for downstream phylogenomics analyses. We ran UCEasy on an Ubuntu 20.04 server with 32-core CPU and 64 GB of RAM to compare the results.

Baca et al. (2017) used UCEs to reconstruct the phylogeny of Adephaga families (Coleoptera), which had proven to be a major challenge because of their exceptional species richness, complicated morphological characteristics and sparse molecular data. Baca et al. (2017) used a bait set to target 5k UCEs (Faircloth 2017). A genomic dataset containing 20 raw sequence reads (with a data volume of 8 Gbases and 3,407 Mbytes), available under BioProject accession PRJNA379181, was downloaded and processed.

The study of Batista et al. (2020) sequenced genomic data in order to solve controversies surrounding the early diversification and biogeography of the genus *Turdus* (Aves, Turdidae). Batista et al. (2020) used a bait set to target 2.5k UCEs (Faircloth et al. 2012), as well as specific probes for *Turdus*, based on 49 of the genetic markers described in Backström et al. (2007). We downloaded and processed a genomic dataset of 115 raw sequence reads (with a data volume of 37 Gbases and 23,333 Mbytes) used in Batista et al. (2020) available at NCBI under the BioProject accession PRJNA574741.

We captured significantly more UCEs than Baca et al. 2017 (305 at 50% vs. 334 at 50% in this study) and Batista et al. (2020) (2,312 of total UCEs, 854 at 85%, 1,931 at 75% vs. 2,436 of total UCEs, 1,708 at 85%, 2,244 at 75% in this study). The study by Baca et al. (2017) was reproduced in a total time of 16 hours and 15 minutes and by Batista et al. (2020) in 63 hours and 47 minutes. In both published studies, the Trinity assembler was used. UCEasy used Spades which gave us better results.

UCEasy successfully reproduced the pipeline of the studies mentioned and met the best practices recommended in the literature for scientific computing. A standardised package, such as that presented here, can help evolutionary biologists by automating laborious tasks and facilitating the reproducibility of computational experiments. Finally, UCEasy architecture is sufficiently robust to support new updates from PHYLUCE without hassle. As future work, we plan to extend UCEasy to include the new PHYLUCE 1.7 version and incorporate new phylogenetic software packages from other developers.

Acknowledgements

The authors would like to thank the Conselho Nacional de Desenvolvimento Científico e Tecnológico-CNPq (grants 149985/2018-5; 129954/2018-7) and Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES-INPA proc. 88887477562/2020-00).

References

- Allio R, Schomaker-Bastos A, Romiguier J, Prosdocimi F, Nabholz B, Delsuc F (2020) MitoFinder: efficient automated large-scale extraction of mitogenomic data in target enrichment phylogenomics. Molecular Ecology Resources 20 (4): 892-905. https://doi.org/10.1111/1755-0998.13160
- Andermann T, Torres Jiménez MF, Matos-Maraví P, Batista R, Blanco-Pastor J, Gustafsson ALS, Kistler L, Liberal I, Oxelman B, Bacon C, Antonelli A (2020) A guide to carrying out a phylogenomic target sequence capture project. Frontiers in Genetics 10: 1407. https://doi.org/10.3389/fgene.2019.01407
- Baca S, Alexandre A, Gustafson G, Short AZ (2017) Ultraconserved elements show utility in phylogenetic inference of Adephaga (Coleoptera) and suggest paraphyly of

‘Hydradephaga’. Systematic Entomology 42 (4): 786-795. https://doi.org/10.1111/syen.12244

- Backström N, Fagerberg S, Ellegren H (2007) Genomics of natural bird populations: a gene-based set of reference markers evenly spread across the avian genome. Molecular Ecology 17 (4): 964-980. https://doi.org/10.1111/j.1365-294x.2007.03551.x
- Bankevich A, Nurk S, Antipov D, Gurevich A, Dvorkin M, Kulikov A, Lesin V, Nikolenko S, Pham S, Prjibelski A, Pyshkin A, Sirotkin A, Vyahhi N, Tesler G, Alekseyev M, Pevzner P (2012) SPAdes: a new genome assembly algorithm and its applications to single-cell sequencing. Journal of Computational Biology 19 (5): 455-477. https://doi.org/10.1089/cmb.2012.0021
- Batista R, Olsson U, Andermann T, Aleixo A, Ribas CC, Antonelli A (2020) Phylogenomics and biogeography of the world's thrushes (Aves,Turdus): new evidence for a more parsimonious evolutionary history. Proceedings of the Royal Society B: Biological Sciences 287 (1919): 20192400. https://doi.org/10.1098/rspb.2019.2400
- Beaulieu-Jones BK, Greene CS (2017) Reproducibility of computational workflows is automated using continuous analysis. Nature Biotechnology 35 (4): 342-346. https://doi.org/10.1038/nbt.3780
- Bejerano G, Pheasant M, Makunin I, Stephen S, Kent WJ, Mattick J, Haussler D (2004) Ultraconserved elements in the human genome. Science 304 (5675): 1321-1325. https://doi.org/10.1126/science.1098119
- Branstetter M, Longino J, Ward P, Faircloth B (2017) Enriching the ant tree of life: enhanced UCE bait set for genome-scale phylogenetics of ants and other Hymenoptera. Methods in Ecology and Evolution 8 (6): 768-776. https://doi.org/10.1111/2041-210x.12742
- Davey J, Hohenlohe P, Etter P, Boone J, Catchen J, Blaxter M (2011) Genome-wide genetic marker discovery and genotyping using next-generation sequencing. Nature Reviews Genetics 12 (7): 499-510. https://doi.org/10.1038/nrg3012
- Edgar RC (2004) MUSCLE: multiple sequence alignment with high accuracy and high throughput. Nucleic Acids Research 32 (5): 1792-1797. https://doi.org/10.1093/nar/gkh340
- Faircloth B, McCormack J, Crawford N, Harvey M, Brumfield R, Glenn T (2012) Ultraconserved elements anchor thousands of genetic markers spanning multiple evolutionary timescales. Systematic Biology 61 (5): 717-726. https://doi.org/10.1093/sysbio/sys004
- Faircloth B (2015) PHYLUCE is a software package for the analysis of conserved genomic loci. Bioinformatics 32 (5): 786-788. https://doi.org/10.1093/bioinformatics/btv646
- Faircloth B (2017) Identifying conserved genomic elements and designing universal bait sets to enrich them. Methods in Ecology and Evolution 8 (9): 1103-1112. https://doi.org/10.1111/2041-210x.12754
- Ferenhof HA, Alves de Sousa MP (2021) Scientific community-driven ecosystem as a supporter to co-create and co-evolute science. Emerging Ecosystem-Centric Business Models for Sustainable Value Creation. pp.53-66. https://doi.org/10.4018/978-1-7998-4843-1.ch003
- Gamma E, Helm R, Johnson R, Vlissides J, Booch G (1994) Design patterns: elements of reusable object-oriented software. Addison-Wesley Professional [ISBN 978-0201633610]

- Georgeson P, Syme A, Sloggett C, Chung J, Dashnow H, Milton M, Lonsdale A, Powell D, Seemann T, Pope B (2019) Bionitio: demonstrating and facilitating best practices for bioinformatics command-line software. GigaScience 8 (9): giz109. https://doi.org/10.1093/gigascience/giz109
- Harvey M, Smith BT, Glenn T, Faircloth B, Brumfield R (2016) Sequence capture versus restriction site associated DNA sequencing for shallow systematics. Systematic Biology 65 (5): 910-924. https://doi.org/10.1093/sysbio/syw036
- Jiménez R, Kuzak M, Alhamdoosh M, Barker M, Batut B, Borg M, Capella-Gutierrez S, Chue Hong N, Cook M, Corpas M, Flannery M, Garcia L, Gelpí J, Gladman S, Goble C, González Ferreiro M, Gonzalez-Beltran A, Griffin P, Grüning B, Hagberg J, Holub P, Hooft R, Ison J, Katz D, Leskošek B, López Gómez F, Oliveira L, Mellor D, Mosbergen R, Mulder N, Perez-Riverol Y, Pergl R, Pichler H, Pope B, Sanz F, Schneider M, Stodden V, Suchecki R, Svobodová Vařeková R, Talvik H, Todorov I, Treloar A, Tyagi S, van Gompel M, Vaughan D, Via A, Wang X, Watson-Haigh N, Crouch S (2017) Four simple recommendations to encourage best practices in research software. F1000Research 6: 876. https://doi.org/10.12688/f1000research.11407.1
- Katoh K, Standley DM (2013) MAFFT multiple sequence alignment software version 7: improvements in performance and usability. Molecular Biology and Evolution 30 (4): 772-780. https://doi.org/10.1093/molbev/mst010
- Leprevost FdV, Barbosa V, Francisco E, Perez-Riverol Y, Carvalho P (2014) On best practices in the development of bioinformatics software. Frontiers in Genetics 5: 199. https://doi.org/10.3389/fgene.2014.00199
- Magee A, May M, Moore B (2014) The dawn of open access to phylogenetic data. PLOS One 9 (10): e110268. https://doi.org/10.1371/journal.pone.0110268
- McCormack JE, Faircloth BC, Crawford NG, Gowaty PA, Brumfield RT, Glenn TC (2011) Ultraconserved elements are novel phylogenomic markers that resolve placental mammal phylogeny when combined with species-tree analysis. Genome Research 22 (4): 746-754. https://doi.org/10.1101/gr.125864.111
- Perez-Riverol Y, Gatto L, Wang R, Sachsenberg T, Uszkoreit J, Leprevost FdV, Fufezan C, Ternent T, Eglen S, Katz D, Pollard T, Konovalov A, Flight R, Blin K, Vizcaíno JA (2016) Ten simple rules for taking advantage of git and gitHub. PLOS Computational Biology 12 (7): e1004947. https://doi.org/10.1371/journal.pcbi.1004947
- Piccolo S, Frampton M (2016) Tools and techniques for computational reproducibility. GigaScience 5 (1): s13742-016. https://doi.org/10.1186/s13742-016-0135-4
- Sandve GK, Nekrutenko A, Taylor J, Hovig E (2013) Ten simple rules for reproducible computational research. PLOS Computational Biology 9 (10): e1003285. https://doi.org/10.1371/journal.pcbi.1003285
- Seemann T (2013) Ten recommendations for creating usable bioinformatics command line software. GigaScience 2 (1): 2047-217X. https://doi.org/10.1186/2047-217x-2-15
- Wilson G, Aruliah DA, Brown CT, Chue Hong N, Davis M, Guy R, Haddock SD, Huff K, Mitchell I, Plumbley M, Waugh B, White E, Wilson P (2014) Best practices for scientific computing. PLOS Biology 12 (1): e1001745. https://doi.org/10.1371/journal.pbio.1001745
- Zlotnick F (1991) The posix.1 standard: a programmer's guide. Addison-Wesley [ISBN 978-0805396058]

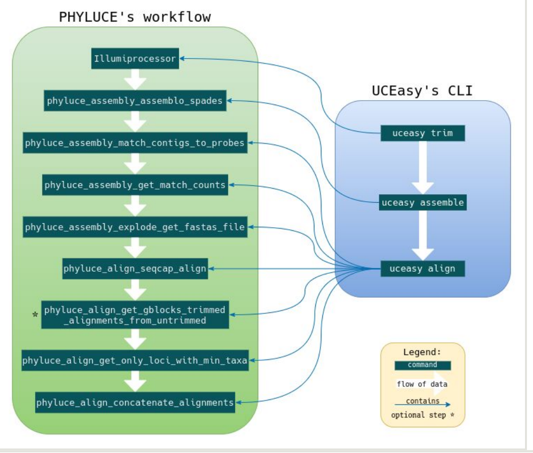
PHYLUCE's workflow
Illumiprocessor
phyluce_assembly_assemblo_spades
phyluce_assembly_match_contigs_to_probes
phyluce_assembly_get_match_counts
phyluce_assembly_explode_get_fastas_file
phyluce_align_seqcap_align
* phyluce_align_get_gblocks_trimmed _alignments_from_untrimmed
phyluce_align_get_only_loci_with_min_taxa
phyluce_align_concatenate_alignments
UCEasy's CLI
uceasy trim
uceasy assemble
uceasy align
Legend:
command
flow of data
contains
optional step *

Figure 1.

UCEasy's CLIs interacting with PHYLUCE's workflow.

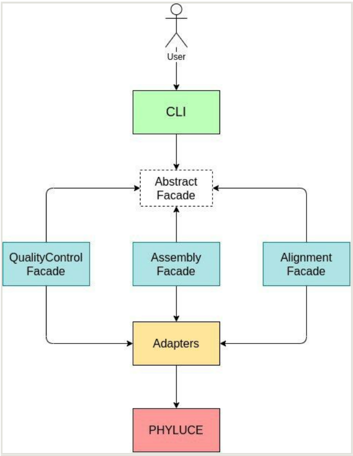

Figure 2.

UCEasy software architecture. The coloured boxes represent components and arrows the dependencies between them. The Abstract Facade implements a common interface with which the other facades have to comply.