

¹ Barcode Validator: A Python toolkit for structural and ² taxonomic validation of DNA barcode sequences

³ **D. S. J. Groenenberg**  ¹ and **Rutger A. Vos**  ¹¶

⁴ ¹ Naturalis Biodiversity Center, Leiden, The Netherlands ¶ Corresponding author

DOI: [10.xxxxxx/draft](https://doi.org/10.xxxxxx/draft)

Software

- [Review](#) ↗
- [Repository](#) ↗
- [Archive](#) ↗

Editor: [Open Journals](#) ↗

Reviewers:

- [@openjournals](#)

Submitted: 01 January 1970

Published: unpublished

License

Authors of papers retain copyright and release the work under a Creative Commons Attribution 4.0 International License ([CC BY 4.0](#)).
¹⁶
¹⁷
¹⁸
¹⁹

⁵ Summary

⁶ DNA barcoding has become a cornerstone technique in molecular biodiversity research, ⁷ enabling rapid species identification and discovery through standardized genetic markers. ⁸ The Barcode Validator is a Python toolkit designed to ensure the quality and accuracy of ⁹ DNA barcode sequences before submission to public databases such as the Barcode of Life ¹⁰ Data System (BOLD) and institutional repositories. The software performs both structural ¹¹ validation (assessing sequence quality, length, ambiguous bases, and marker-specific features ¹² like stop codons) and taxonomic validation (verifying specimen identifications through reverse ¹³ taxonomy using BLAST-based identification services). Developed to support large-scale ¹⁴ biodiversity genomics initiatives, particularly the Biodiversity Genomics Europe (BGE) and ¹⁵ ARISE projects, the toolkit provides automated workflows for processing thousands of sequences ¹⁶ with flexible configuration options and comprehensive reporting. The source code is available ¹⁷ on GitHub at https://github.com/naturalis/barcode_validator under the Apache License 2.0, ¹⁸ with distribution via PyPI and Bioconda, and a Galaxy tool wrapper for web-based access.
¹⁹

¹⁹ Statement of need

²⁰ DNA barcoding projects generate large volumes of sequence data that must meet stringent ²¹ quality standards before deposition in public databases. Manual validation of sequences is ²² time-consuming, error-prone, and impractical for projects processing hundreds or thousands of ²³ specimens. Furthermore, the increasing adoption of genome skimming and high-throughput ²⁴ sequencing technologies produces multiple assembly attempts per specimen, requiring intelligent ²⁵ selection of the best valid sequence among alternatives.

²⁶ The Barcode Validator addresses these needs by providing:

- ²⁷ 1. **Integrated validation:** Combined structural and taxonomic validation in a single workflow, ²⁸ with support for marker-specific requirements (e.g., stop codon detection for protein-²⁹ coding genes via translation, GC content assessment for non-coding markers).
- ³⁰ 2. **Assembly triage:** Automatic selection of the best valid sequence when multiple assembly ³¹ attempts exist per specimen, using configurable criteria including validation results and ³² optional assembly quality metrics.
- ³³ 3. **Flexible taxonomic validation:** Support for multiple identification backends (BOLD API, ³⁴ local BLAST, Galaxy web services) and taxonomic backbones (BOLD, NCBI, Netherlands ³⁵ Species Register), enabling validation against expected specimen identifications at ³⁶ configurable taxonomic ranks.
- ³⁷ 4. **Batch processing:** Efficient handling of large datasets through batched API calls and ³⁸ parallel processing where appropriate.

39 5. **Workflow integration:** Command-line interface suitable for automated pipelines, with
40 Galaxy tool integration for web-based access.

41 The software has been deployed in production workflows at Naturalis Biodiversity Center
42 (the Netherlands) and the Natural History Museum (United Kingdom) for the BGE project,
43 processing thousands of arthropod COI sequences from genome skimming experiments, and
44 the ARISE project for the validation of thousands of freshly sequenced vertebrate and marine
45 and terrestrial invertebrate specimens. Its design supports the quality assurance requirements
46 of modern DNA barcoding initiatives while remaining flexible enough to accommodate diverse
47 project-specific workflows.

48 State of the field

Existing tools for DNA barcode quality assurance are typically limited in scope. Quality control tools like FastQC ([Andrews, 2010](#)) assess raw read quality rather than assembled barcode sequences. Taxonomic identification tools like BOLD's identification engine ([Ratnasingham & Hebert, 2007](#)) or standalone BLAST ([Altschul et al., 1990](#)) provide species identification but lack integration with structural quality metrics. Biopython ([Cock et al., 2009](#)) provides sequence manipulation and translation capabilities but not marker-specific HMM alignment for reading frame detection. Profile HMM tools such as HMMER ([Eddy, 2011](#)) enable sequence alignment but do not incorporate downstream validation logic.

No comprehensive solution exists that integrates structural validation, taxonomic verification, and assembly triage for the specific workflow requirements of modern barcoding projects. The Barcode Validator's contribution lies in this integration: combining HMM-based codon phase detection, taxon-aware translation table selection, multi-backend taxonomic validation, and assembly triage into a single configurable pipeline. No existing package provided extension points suitable for adding this combined functionality; the unique combination of requirements necessitated new software.

64 Software design

Barcode Validator is implemented in Python (3.9+) with a modular, extensible architecture built around several key design patterns:

- **Strategy pattern** for validators: An abstract Validator base class defines the validation interface, with concrete implementations for structural validation (StructuralValidator with subclasses ProteinCodingValidator and NonCodingValidator) and taxonomic validation (TaxonomicValidator).
 - **Factory pattern** for services: Pluggable identification services (IDService hierarchy supporting BOLD, BLAST, and Galaxy backends) and taxonomic resolvers (TaxonResolver supporting BOLD, NCBI, and NSR taxonomies) enable flexible backend selection.
 - **Orchestration pattern**: A ValidationOrchestrator coordinates the validation pipeline, managing validator initialization, batch processing, result aggregation, and output generation.

The central design decision was to separate validation logic (what measurements to collect) from validity adjudication (what thresholds constitute pass/fail). This separation enables the same codebase to serve diverse projects with different quality requirements—genome skimming workflows demanding zero ambiguous bases versus Sanger sequencing tolerating several—without code modifications. The Strategy pattern for validators and Factory pattern for services enable runtime selection of validation approaches and identification backends. While this introduces abstraction overhead, it proved essential for accommodating the consortium’s

85 heterogeneous infrastructure: some partners operate local BLAST databases, others rely on
86 Galaxy web services, and still others use BOLD's identification API directly.

87 The software integrates with established bioinformatics tools including BLAST+ (Camacho et
88 al., 2009) for sequence similarity searches, HMMER (Eddy, 2011) for profile Hidden Markov
89 Model-based alignment and codon phase detection, and Biopython (Cock et al., 2009) for
90 sequence manipulation and translation. External validation is performed through REST API
91 calls to BOLD (Ratnasingham & Hebert, 2007) and Galaxy (The Galaxy Community, 2022)
92 identification services.

93 Input data can be provided as FASTA files with optional CSV metadata and BOLD Excel
94 spreadsheets containing specimen and taxonomic information. Validation results are output in
95 both human-readable TSV format (with detailed pass/fail status for each validation criterion)
96 and filtered FASTA format (containing only sequences meeting all validation requirements).

97 Research impact statement

98 The Barcode Validator has demonstrated substantial realized impact through its deployment in
99 the Biodiversity Genomics Europe (BGE) project. As documented in BGE Deliverable D8.4, the
100 toolkit processed sequences from over 18,500 specimens across 68 taxonomic orders, enabling
101 the submission of more than 47,000 validated DNA barcode sequences to BOLD and the
102 European Nucleotide Archive by October 2025. The validation framework identified systematic
103 issues including plate-swap errors that would otherwise have corrupted database submissions,
104 and revealed taxonomic patterns in validation success rates ranging from 0% to 100% across
105 orders—insights that directly informed protocol optimizations.

106 The software is deployed in production workflows at Naturalis Biodiversity Center (Netherlands)
107 and the Natural History Museum (United Kingdom), with the ARISE project using it for
108 validation of freshly sequenced vertebrate and invertebrate specimens. Community readiness is
109 evidenced by: distribution through PyPI and Bioconda channels; availability as a Galaxy tool
110 wrapper enabling web-based access for non-technical users; comprehensive documentation
111 including architecture diagrams and use-case examples; an Apache 2.0 license; and a public
112 GitHub repository with contribution guidelines. The toolkit's analytical outputs informed
113 the BGE consortium's understanding of genome skimming assembly parameter optimization,
114 demonstrating that the combination of specific preprocessing steps (*fcleaner*=TRUE,
115 *merge*=FALSE) with alignment thresholds (*r*=1.0, *s*=50) maximizes barcode recovery while
116 maintaining stringent quality standards.

117 AI usage disclosure

118 The overall software architecture—including the Strategy pattern for validators, Factory pattern
119 for services, and the separation of validation logic from criteria-based adjudication—was
120 conceived by the author prior to the widespread availability of usable large language models,
121 drawing on established object-oriented design principles. The parameterization of validation
122 logic, including marker-specific thresholds, taxonomic validation levels, and quality criteria, was
123 determined through iterative discussions among consortium users based on empirical analysis
124 of validation outcomes.

125 However, portions of the implementation benefited from generative AI assistance. Specifically,
126 Claude (Anthropic) and ChatGPT (OpenAI) were used to accelerate code syntax generation
127 for routine operations, produce initial drafts of docstrings and inline documentation, and refine
128 error handling patterns. The author reviewed, tested, and modified all AI-generated content
129 before incorporation. This manuscript was drafted by the author with AI assistance for prose
130 refinement and structural suggestions.

131 Acknowledgements

132 This work was supported by the Biodiversity Genomics Europe (BGE) project, which has received
133 funding from the European Union's Horizon Europe Research and Innovation Programme
134 under grant agreement No. 101059492, and the ARISE project. The author acknowledges the
135 Naturalis Biodiversity Center for institutional support and infrastructure, and Dick Groenenberg,
136 Dan Parsons and Ben Price for extensive testing, feedback, and improvement suggestions over
137 the course of this project.

138 References

- 139 Altschul, S. F., Gish, W., Miller, W., Myers, E. W., & Lipman, D. J. (1990). Basic local
140 alignment search tool. *Journal of Molecular Biology*, 215(3), 403–410. [https://doi.org/10.1016/S0022-2836\(05\)80360-2](https://doi.org/10.1016/S0022-2836(05)80360-2)
- 141 Andrews, S. (2010). *FastQC: A quality control tool for high throughput sequence data*.
<https://www.bioinformatics.babraham.ac.uk/projects/fastqc/>
- 142 Camacho, C., Coulouris, G., Avagyan, V., Ma, N., Papadopoulos, J., Bealer, K., & Madden,
143 T. L. (2009). BLAST+: Architecture and applications. *BMC Bioinformatics*, 10(1), 421.
<https://doi.org/10.1186/1471-2105-10-421>
- 144 Cock, P. J., Antao, T., Chang, J. T., Chapman, B. A., Cox, C. J., Dalke, A., Friedberg,
145 I., Hamelryck, T., Kauff, F., Wilczynski, B., & Hoon, M. J. de. (2009). Biopython:
146 Freely available Python tools for computational molecular biology and bioinformatics.
Bioinformatics, 25(11), 1422–1423. <https://doi.org/10.1093/bioinformatics/btp163>
- 147 Eddy, S. R. (2011). Accelerated profile HMM searches. *PLoS Computational Biology*, 7(10),
148 e1002195. <https://doi.org/10.1371/journal.pcbi.1002195>
- 149 Ratnasingham, S., & Hebert, P. D. (2007). BOLD: The barcode of life data system. *Molecular
150 Ecology Notes*, 7(3), 355–364. <https://doi.org/10.1111/j.1471-8286.2007.01678.x>
- 151 The Galaxy Community. (2022). The Galaxy platform for accessible, reproducible and
152 collaborative biomedical analyses: 2022 update. *Nucleic Acids Research*, 50(W1),
153 W345–W351. <https://doi.org/10.1093/nar/gkac247>