The Journal of Open Source Software

Antique.jl: A Julia package on analytical solutions of quantum mechanical equations

Shuhei Ohno [1,2], **Ahmad Jafar Arifi** [1,3], **and Lucas Happ** [1]

1 Few-body Systems in Physics Laboratory, RIKEN Nishina Center, Wako 351-0198, Japan **2** Graduate School of Nanobioscience, Yokohama City University, Yokohama 236-0027, Japan **3** Research Center for Nuclear Physics, Osaka University, Ibaraki 567-0047, Japan

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor: Owen Lockwood

Reviewers:

- @goerz
- @mdavezac

Submitted: 25 April 2025
Published: unpublished

Summary

Antique.jl provides implementations of analytical solutions to solvable quantum mechanical models. Analytical solutions are the most reliable benchmarks for software testing in the development of numerical methods. In addition to testing numerical methods, this package is useful for teaching quantum mechanics. Each solution has been tested using both a computer algebra system and numerical integration, thanks to the multiple dispatch in the Julia programming language.

Statement of need

In modern physics and chemistry, numerical methods for quantum mechanical equations have been developed to calculate the properties of hadrons, nuclei, atoms, molecules, and other systems. The quantum mechanical models with analytical solutions, such as the harmonic oscillator and hydrogen atom, are often used as benchmarks for software testing in the development of numerical methods (Hiyama & Kamimura, 2018). There are many articles and books on analytical solutions, but few software implementations. To our knowledge, Antique.jl is the first package on analytical solutions of solvable quantum mechanical models. The implementation for each model provides functions for the potential, the eigenvalues, the eigenfunctions and some other properties. Ten models are currently supported and more models will be added in the future.

Antique.jl was developed for researchers, lecturers, students, and any person who is interested in quantum mechanics. Since Julia is designed from the ground up for numerical and scientific computing (Bezanson et al., 2017), many packages for quantum mechanical calculations will be developed using this package in the future. In addition to benchmarking numerical methods, possible applications range from prototyping quark models in hadron physics to developing basis sets in quantum chemistry and teaching materials for quantum mechanics. The documentation includes graphs suitable for teaching materials. We hope that textbooks for quantum mechanics will be written using this package.

State of the field

To our knowledge, there is no dedicated package on analytical solutions; only a few implementations are available (Rodríguez-Gómez & Pérez-Martínez, 2020; Velieva & Fedorov, 2020). However, there are several packages on the special polynomials contained in the analytical solutions, such as associated Laguerre polynomials and associated Legendre polynomials, which appear in the wave functions of hydrogen-like atoms.

The special polynomials are usually defined by Rodrigues' formula, but their closed form is better for direct implementation. The correspondence between a Rodrigues' formula and a closed form is not usually provided in textbooks. Therefore, a closed form or an implementation corresponding to the Rodrigues' formula is needed. However, the definitions vary from textbook to textbook (Greiner, 2001; Griffiths & Schroeter, 2018), and package to package (ISO/IEC, 2020; Virtanen et al., 2020). The time and effort to survey, implement, and test the special polynomials is a barrier to implementing analytical solutions.

The Julia language removes this barrier by the multiple dispatch. The multiple dispatch makes it easier to test polynomials using both computer algebra systems (Gowda et al., 2022) and numerical integration (Johnson, 2013). The match of the closed form and the expansion of the Rodrigues' formula is checked using a computer algebra system. Additionally, the orthonormality has been tested using numerical integration. Finally, the orthonormality of the wave functions is tested using numerical integration, and the expectation value of the Hamiltonian is calculated from the eigenfunction to confirm its correspondence with the eigenvalues.

Example usage

The quantum mechanical models with analytical solutions are useful for software testing in the development of numerical methods. Figure 1 shows an example of a variational calculation for the hydrogen atom (Thijssen, 2007) and a comparison with the exact solution provided by Antique.jl.

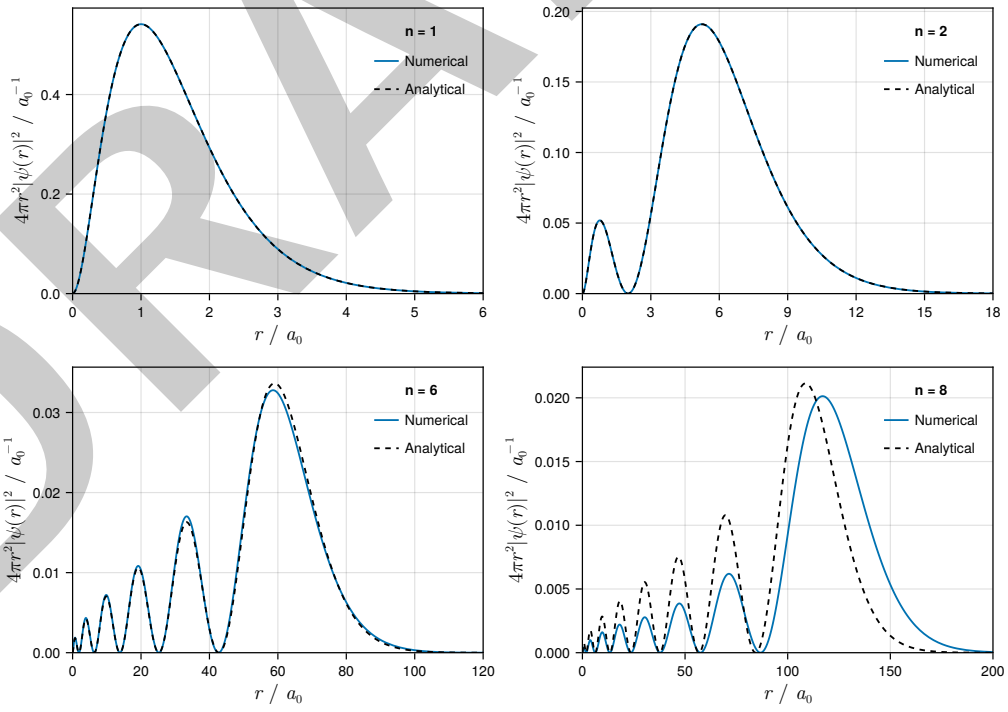

Figure 1: The radial density of four s-wave states of the hydrogen atom calculated by the Rayleigh-Ritz method, implemented in TwoBody.jl (Ohno, 2024). We employed Gaussian basis functions $\phi_n(r) = \exp(-\nu_n r^2)$ whose exponents were determined by the geometric progression defined in the previous study (Hiyama & Kamimura, 2018). The numerical solution (solid blue line) is compared to the analytical solution (dashed black line) using Antique.jl. This figure is drawn using Mekie.jl (Danisch & Krumbiegel, 2021).

The Journal of Open Source Software

Acknowledgement

We acknowledge all contributors and sponsors. Special thanks to Yuto Horikawa for his help with managing the documentation and advice on the coding style. S. O. was supported by the RIKEN Junior Research Associate Program. A. J. A. and L. H. were supported by the RIKEN Special Postdoctoral Researcher Program.

References

Bezanson, J., Edelman, A., Karpinski, S., & Shah, V. B. (2017). Julia: A fresh approach to numerical computing. *SIAM Review*, *59*(1), 65–98. https://doi.org/10.1137/141000671

Danisch, S., & Krumbiegel, J. (2021). Makie.jl: Flexible high-performance data visualization for Julia. *Journal of Open Source Software*, *6*(65), 3349. https://doi.org/10.21105/joss.03349

Gowda, S., Ma, Y., Cheli, A., Gwóźdź, M., Shah, V. B., Edelman, A., & Rackauckas, C. (2022). High-performance symbolic-numerics via multiple dispatch. *ACM Commun. Comput. Algebra*, *55*(3), 92–96. https://doi.org/10.1145/3511528.3511535

Greiner, W. (2001). *Quantum mechanics*. Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-56826-8

Griffiths, D. J., & Schroeter, D. F. (2018). *Introduction to quantum mechanics*. Cambridge University Press. https://doi.org/10.1017/9781316995433

Hiyama, E., & Kamimura, M. (2018). Study of various few-body systems using gaussian expansion method (GEM). *Frontiers of Physics*, *13*(6). https://doi.org/10.1007/s11467-018-0828-5

ISO/IEC. (2020). *ISO/IEC 14882:2020 Programming languages — C++*. International Organization for Standardization. https://www.iso.org/standard/79358.html

Johnson, S. G. (2013). *QuadGK.jl: Gauss–Kronrod integration in Julia*. https://github.com/JuliaMath/QuadGK.jl.

Ohno, S. (2024). *TwoBody.jl: A julia package for quantum mechanical two-body problems*. https://github.com/ohno/TwoBody.jl.

Rodríguez-Gómez, A., & Pérez-Martínez, A. L. (2020). A full-fledged analytical solution to the quantum harmonic oscillator for undergraduate students of science and engineering. *Physics*, *2*(4), 541–570. https://doi.org/10.3390/physics2040031

Thijssen, J. (2007). *Computational physics*. Cambridge University Press. https://doi.org/10.1017/cbo9781139171397

Velieva, T. R., & Fedorov, A. V. (2020). *Modeling a quantum harmonic oscillator using julia*. 158–167. http://ceur-ws.org/Vol-2639/#paper-14

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., … SciPy 1.0 Contributors. (2020). SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, *17*, 261–272. https://doi.org/10.1038/s41592-019-0686-2