




1 NLSE: A Python package to solve the nonlinear 2 Schrödinger equation

3 **Tangui Aladjidi** ¹¶, **Clara Piekarski** ¹, and **Quentin Glorieux** ¹

4 ¹ Laboratoire Kastler Brossel, Sorbonne University, CNRS, ENS-PSL University, Collège de France; 4
5 Place Jussieu, 75005 Paris, France ¶ Corresponding author

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

Software

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

Editor: [Rocco Meli](#) 

Reviewers:

- [@Abinashbunty](#)
- [@oblivateandsurrender](#)

Submitted: 19 March 2024

Published: unpublished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](#)).

6 Summary

7 The non-linear Schrödinger equation (NLSE) is a general non-linear equation allowing to model
8 the propagation of light in non-linear media. This equation is mathematically isomorphic to
9 the Gross-Pitaevskii equation (GPE) ([Pitaevskij & Stringari, 2016](#)) describing the evolution
10 of cold atomic ensembles. Recently, the growing field of quantum fluids of light ([Carusotto
11 & Ciuti, 2013](#)) has proven a fruitful testbed for several fundamental quantum and classical
12 phenomena such as superfluidity ([Michel et al., 2018](#)) or turbulence ([Baker-Rasooli et al.,
13 2023](#)). Providing a flexible, modern and performant framework to solve these equations is a
14 crucial need to model realistic experimental scenarios.

15 Statement of need

Over the years, there have been several packages striving to provide performant split-step solvers for NLSE type equations. Here are a few examples:

- [FourierGPE.jl](#) for 1D to 3D Gross-Pitaevskii equations in the context of cold atoms in Julia.
- [GPUe](#) ([Schloss & O'Riordan, 2018](#)) for 1D to 3D Gross-Pitaevskii equations accelerated on GPU, in C++ (currently unmaintained).
- [py-fmas](#) for 1D NLSE in optical fibers, with a split-step method (currently unmaintained).

23 With our project, we bring similar performance to C++ and Julia implementations, while
24 striving for accessibility and maintainability by using the prevalent language in the physics
25 community, Python. Using an easy to extend object-oriented classes, users can readily input
26 experimental parameters to quickly model real setups.

27 Functionality

28 NLSE harnesses the power of pseudo-spectral schemes in order to solve efficiently the following
29 general type of equation:

$$i\partial_t\psi = -\frac{1}{2m}\nabla^2\psi + V\psi + g|\psi|^2\psi.$$

30 In order to take advantage of the computing power of modern Graphical Processing Units
31 (GPU) for Fast Fourier Transforms (FFT), the main workhorse of this code is the [Cupy](#) ([Okuta
32 et al., 2017](#)) package that maps [Numpy](#) ([Harris et al., 2020](#)) functionalities onto the GPU using
33 NVIDIA's [CUDA](#) API. It also heavily uses just-in-time compilation using [Numba](#) ([Lam et al.,
34 2015](#)) in order to optimize performance while having an easily maintainable Python codebase.

35 Compared to naive Numpy based CPU implementations, this package provides a 100 to 10000
36 times speedup for typical sizes [Figure 2](#). While optimized for the use with GPU, it also provides
37 a performant CPU fallback layer.

38 The goal of this package is to provide a natural framework for all physicists wishing to model
39 the propagation of light in non-linear media or the temporal evolution of Bose gases. It can
40 also be used to model the propagation of light in general. It supports lossy, non-linear and
41 non-local media.

42 It provides several classes to model 1D, 2D or 3D propagation, and leverages the array
43 functionalities of Numpy like broadcasting in order to allow scans of physical parameters to
44 most faithfully replicate experimental setups.

45 This code has been developed during the author's PhD thesis ([Aladjidi, 2023](#)) and used as
46 the main simulation tool for several publications like ([Glorieux et al., 2023](#)) and ([Baker-Rasooli
47 et al., 2023](#)).

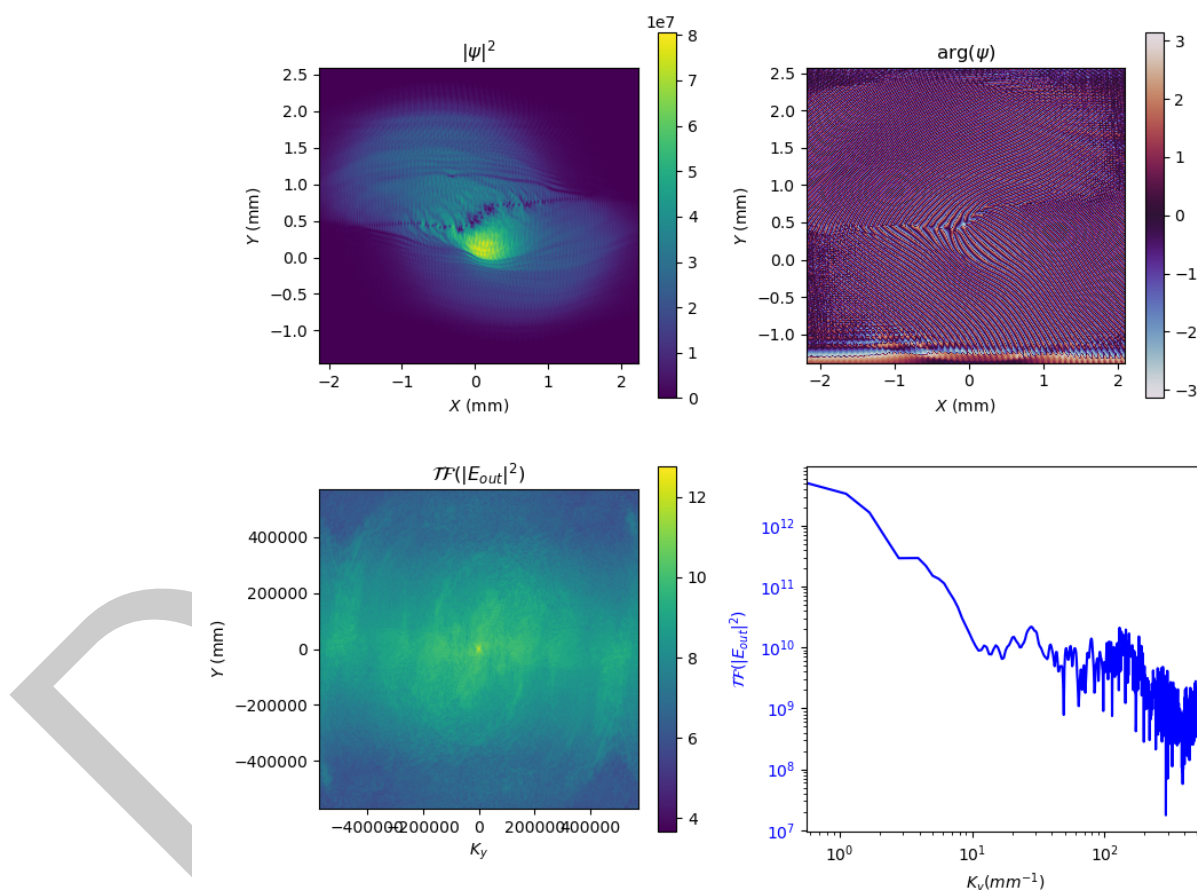


Figure 1: Example of an output of the solver. A shearing layer is observed nucleating vortices, that are attracted towards the center due to an attractive potential. The density and phase of the field are represented as well as the momentum distribution in order to get a quick overview of the state of the field.

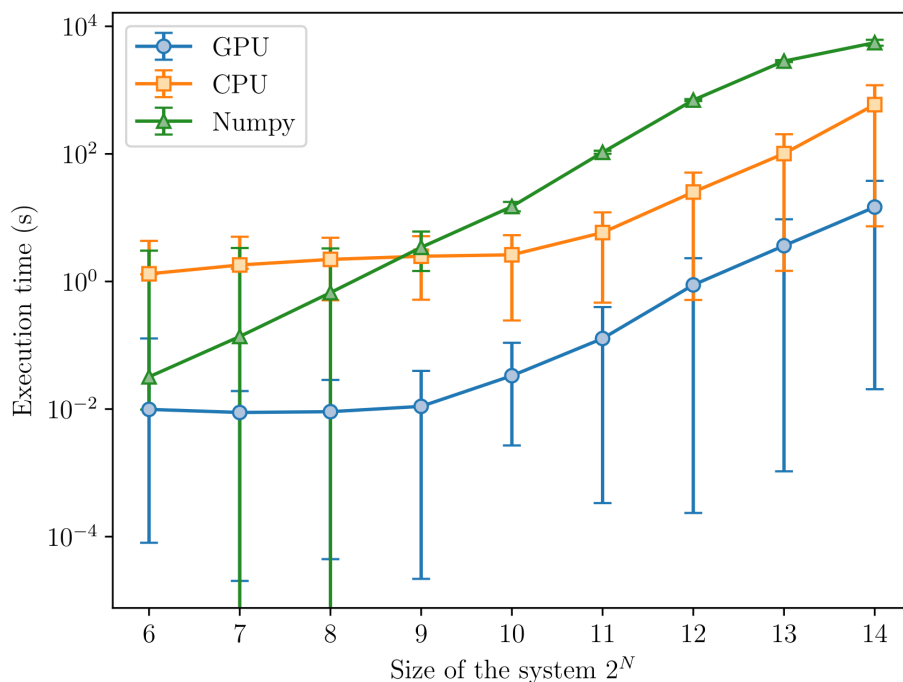


Figure 2: CPU vs GPU benchmark for 1 cm of propagation (200 evolution steps).

48 Acknowledgements

49 We acknowledge contributions from Myrann Baker-Rasooli as our most faithful beta tester.

50 Authors contribution

51 T.A wrote the original code and is the main maintainer, C.P extended the functionalities to
52 include coupled systems. Q.G supervised the project.

53 References

- 54 Aladjidi, T. (2023). *Full optical control of quantum fluids of light in hot atomic vapors* (Theses
55 No. 2023SORUS406, Sorbonne Université). <https://theses.hal.science/tel-04391272>
- 56 Baker-Rasooli, M., Liu, W., Aladjidi, T., Bramati, A., & Glorieux, Q. (2023). Turbulent
57 dynamics in a two-dimensional paraxial fluid of light. *Physical Review A*, *108*(6), 063512.
58 <https://doi.org/10.1103/PhysRevA.108.063512>
- 59 Carusotto, I., & Ciuti, C. (2013). Quantum fluids of light. *Rev. Mod. Phys.*, *85*(1), 299–366.
60 <https://doi.org/10.1103/RevModPhys.85.299>
- 61 Glorieux, Q., Aladjidi, T., Lett, P. D., & Kaiser, R. (2023). Hot atomic vapors for nonlinear
62 and quantum optics. *New Journal of Physics*, *25*(5), 051201. <https://doi.org/10.1088/1367-2630/acce5a>
63
- 64 Harris, C. R., Millman, K. J., Walt, S. J. van der, Gommers, R., Virtanen, P., Cournapeau, D.,
65 Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., Kerkwijk,

- 66 M. H. van, Brett, M., Haldane, A., Ríó, J. F. del, Wiebe, M., Peterson, P., ... Oliphant,
67 T. E. (2020). Array programming with NumPy. *Nature*, 585(7825), 357–362. <https://doi.org/10.1038/s41586-020-2649-2>
68
- 69 Lam, S. K., Pitrou, A., & Seibert, S. (2015). Numba: A llvm-based python jit compiler.
70 *Proceedings of the Second Workshop on the LLVM Compiler Infrastructure in HPC*, 1–6.
- 71 Michel, C., Boughdad, O., Albert, M., Larré, P.-É., & Bellec, M. (2018). Superfluid motion
72 and drag-force cancellation in a fluid of light. *Nat. Comm.*, 9(1), 2108. <https://doi.org/10.1038/s41467-018-04534-9>
73
- 74 Okuta, R., Unno, Y., Nishino, D., Hido, S., & Loomis, C. (2017). CuPy: A NumPy-
75 compatible library for NVIDIA GPU calculations. *Proceedings of Workshop on Machine*
76 *Learning Systems (LearningSys) in the Thirty-First Annual Conference on Neural Information*
77 *Processing Systems (NIPS)*. http://learningsys.org/nips17/assets/papers/paper_16.pdf
- 78 Pitaevskij, L. P., & Stringari, S. (2016). *Bose-einstein condensation and superfluidity*. Oxford
79 University Press. ISBN: 978-0-19-875888-4
- 80 Schloss, J. R., & O’Riordan, L. J. (2018). GPUE: Graphics processing unit gross–pitaevskii
81 equation solver. *Journal of Open Source Software*, 3(32), 1037. [https://doi.org/10.21105/](https://doi.org/10.21105/joss.01037)
82 [joss.01037](https://doi.org/10.21105/joss.01037)

DRAFT