The Journal of Open Source Software

pyElli: An open source ellipsometry analysis tool for FAIR data

Marius J. Müller [1], **Sangam Chatterjee** [1], **and Florian Dobener** [1]

1 Institute of Experimental Physics I and Center for Materials Research (ZfM/LaMa), Justus Liebig University Giessen, Heinrich-Buff-Ring 16, Giessen, D-35392 Germany

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor: Bonan Zhu

Reviewers:

- @phoebe-p
- @Mil152

Submitted: 22 May 2025
Published: unpublished

Summary

pyElli is an open-source Python-based analysis tool for evaluating the linear optical interaction of layered materials. The code primarily targets spectroscopic ellipsometry (SE). In addition, it is adaptable to various transmission and reflection experiments featuring spectral and polarization resolution.

Various scientific fields use SE to determine the optical functions of materials and multiple-layer stacks often referred to simply as "optical constants" or "material parameters". The experimental "as-measured" SE data requires numerical analysis to deduce physically meaningful quantities. Typical approaches apply transfer-matrix methods (TMM) (*Guide to Using WVASE*, 2012; Tompkins & Irene, 2005). Here, an interaction matrix describes the optical response of each individual material layer. Determining the optical response of a multilayer system then requires matrix multiplication of the individual layers' matrices.

Ellipsometer hardware typically supports bundled proprietary software solutions enabling such analyses. Unfortunately, each manufacturer supplies their own adapted software solution. This approach promises efficient laboratory workflows - if working flawlessly. However, it binds scientists to specific optical models available and limits experiments to the ones supported in the provided software. Furthermore, data interchange can be cumbersome. For example, results may even be hard to reproduce on competing systems due to the use of other models or parameters. In addition, limitations of the specific software included with each instrument may stimulate scientists to use third-party software: bundled software packages may not support specific desirable kinds of analyses, such as including the response of optically anisotropic materials or simultaneous fitting of external experimental parameters.

pyElli offers an open-source alternative extending the capabilities of existing solutions, while aiming to remain as compatible as possible, by providing data imports from various manufacturers (Woollam VWASE, Woollam CompleteEASE, Sentech, Accurion). The code is designed with extensibility and adaptability in mind enabling the implementation of individually adapted models as well as data evaluation with custom tools. Typical examples for advanced use-cases are implementations of custom experimental geometries not covered by other software (Eberheim et al., 2022), imaging ellipsometry, or as a full FAIR data automated analysis pipeline for SE measurements. **pyElli** also supports recent advances in the standardization of ellipsometry data and models, addressing the need for FAIR data (Wilkinson et al., 2016).

pyElli aims to provide a straight-forward database of predefined dispersion models for analyzing materials. All optical models adhere closely to the literature (Hilfiker & Tiwald, 2018). The software easily includes the popular public-domain database for optical constants refractiveindex.info (Polyanskiy, 2024). This allows the inclusion of literature dispersions with a single line of code. Additional dispersion relations can be either hard coded, which is more efficient, or parsed from a text-based domain-specific language into a dispersion which can be fitted, e.g., polynomially.

pyElli supports multiple solving algorithms with different characteristics. Currently, two algorithms using different formulations are available: a fast algorithm based on a 2x2 matrix formulation (Byrnes, 2020) and a more complex 4x4 matrix formulation (Berreman, 1972; Castany, 2021; Castany & Molinaro, 2021). The 2x2 matrix algorithm divides the light into two perpendicular linearly polarized beams, which are solved separately. One of its limitations is eliminating the possibility to include birefringent materials. The 4x4 matrix approach fully solves Maxwell's equations. These equations describe the complete electromagnetic field inside each layer of the sample and couple these together using the matrix formalism. This allows finding solutions to more complex problems such as anisotropic materials, active media or magneto-optic samples.

pyElli ensures fast processing through fully vectorized algorithms for multiple wavelengths and by leveraging numerical algebra libraries like NumPy (Harris et al., 2020) and SciPy (Virtanen et al., 2020). Together, these runtime advantages enable the practical use of advanced fitting algorithms such as global optimizers while maintaining reasonable evaluation times. As a result, **pyElli** enables integrated in-situ monitoring and real-time data analysis of overlayer growth. Furthermore, the use of Python and vectorization libraries also facilitates the development of artificial intelligence-based SE data analysis.

Statement of need

The importance of publishing data according to the FAIR (Findable, Accessible, Interoperable, and Reusable) principles is growing (Wilkinson et al., 2016). Many research journals already require authors to add supporting data, and there is a growing expectation from sponsors for institutes and researchers to implement data governance. The FAIR principles have recently been extended to apply to research software as well since reproducing data requires not only the data itself but also the software used to create it (Barker et al., 2022). Producing FAIR data and using a FAIR and open analysis pipeline is especially important for SE, as the results are tightly related and dependent on the algorithms and models used for evaluation.

An open-source toolkit, **pyElli** has many inherent benefits over proprietary software. For SE, optical models vary between manufacturers and translation can be difficult without comprehensive documentation. **pyElli's** open-source nature makes optical models extendable, auditable, and fully comprehensive. Each version of **pyElli** is associated with a DOI and a Zenodo upload, allowing for reliable referencing and reproducibility of analysis results. It supports reading and writing standardized files in the nexus format and is also integrated as an example in the research data management software NOMAD (Scheidgen et al., 2023).

In summary, **pyElli** aims to provide the means of more straight-forward data analysis, reproducibility, and FAIR data management within the ellipsometry community.

Software with similar functionalities

Other notable Python open-source software for solving transfer-matrices is available, but tends to focus on different aspects:

- PyGTM (Passler & Paarmann, 2017, 2019) provides a non-vectorized, extensive general transfer matrix approach. It allows calculation of additional parameters, like the local strength of the electric field at any position in the multilayer stack.
- PyLlama (Bay et al., 2022) focuses on the simulation of liquid crystals and uses non-vectorized TMM and a scattering matrix algorithm (rigorous coupled-wave analysis, RCWA).
- RayFlare (Pearce, 2021) is a complete toolkit to simulate the physical and electrical properties of solar cells. It provides a vectorized version of the 2x2 algorithm(Byrnes, 2020) and a scattering matrix approach (S4).

- tmm_fast (Luce et al., 2022) is a vectorized variant of Byrnes' algorithm for artificial intelligence-based analysis of multilayer stacks.
- tmmax (Bahrem Serhat Danis, 2025) is a JIT-compilable version of the 2x2 matrix method, leveraging the *JAX* toolkit.
- PyMoosh (Langevin et al., 2024) is a comprehensive toolkit for computing the optical properties of multilayered structures, with a plethora of available scattering and transfer matrix algorithms.

Example: Building a model for an oxide layer on silicon

This example aims to illustrate the straight-forward implementation and building of an optical model in **pyElli**. The chosen model system is a thin SiO_2 layer on bulk Si. A Cauchy dispersion function describes the SiO_2. Tabulated literature values for Si are loaded from the refractiveindex.info database.

The necessary libraries are loaded before building the model: **pyElli** is imported from the module `elli`. The `ParamsHist` wrapper around the `Parameters` class from lmfit (Newville et al., 2024) is imported from `elli.fitting`. It adds history functionality to revert any undesired model changes and to return to an earlier set of parameters. Importing `linspace` from `numpy` enables the generation of a wavelength axis.

```
from numpy import linspace
import elli
from elli.fitting import ParamsHist, fit
```

Initially, we define the fit parameters. The Cauchy model for SiO_2 is defined by the `SiO2_n0` and `SiO2_n1` parameters along with its layer thickness measured in nanometers. Optionally, additional settings for *lmfit* parameters like constraints or bounds might be added; refer to *lmfit's* documentation for a list of parameter arguments or consult our verbose basic example.

```
params = ParamsHist()
params.add("SiO2_n0", value=1.452)
params.add("SiO2_n1", value=36.0)
params.add("SiO2_thickness", value=20)
```

Next, the Cauchy model is created using the `Cauchy` class and the defined parameters. All undefined Cauchy coefficients are kept at their default value of zero in this particular case. Subsequently, the `.get_mat()` method is called on the created object to automatically convert the dispersion into an isotropic material. A list of different dispersion classes and their usage is given in the documentation.

```
SiO2 = elli.Cauchy(
  params["SiO2_n0"],
  params["SiO2_n1"],
).get_mat()
```

Next, tabulated literature values for silicon is read from a database. Therefor, the code instantiates the refractiveindex.info database and queries it for the material (`Si`) and author (`Aspnes`). It is also possible to search the database to get a list of matching entries. See the documentation for details.

```
rii_db = elli.db.RII()
Si = rii_db.get_mat("Si", "Aspnes")
```

All materials are now instantiated and enable the build of the layered structure. The `Structure` class takes three arguments:

- The incident half-space, which is typically air.

- The layer stack as a list of `Layer` objects. The `Layer` object is created by adding the material and the layer thickness.
- The lower half-space, which represents the substrate.

The incident and lower half-space are modeled as infinite to terminate the calculation. Consequently, the calculation inherently cannot capture backside scattering from the bottom substrate surface.

```
structure = elli.Structure(
    elli.AIR,
    [elli.Layer(SiO2, params["SiO2_thickness"])],
    Si,
)
```

Finally, calling the `evaluate(...)` method of the `structure` object triggers the calculation. The example uses a `wavelengths` array from 210 nm to 800 nm for the calculation range and an angle of incidence of 70 degree.

```
wavelengths = linspace(210, 800, 100)
angle = 70
result = structure.evaluate(wavelengths, angle)
```

The calculation is stored in the `result` variable, which is a `Result` object. This object can hold all input parameters and the calculation results. Methods like `psi`, `delta`, `R`, etc., will deliver the desired output. Alternatively, the `@fit` decorator in `elli.fitting` automatically calls a widget-based fitting GUI for Jupyter notebooks. Figure 1 shows the output when used with this example model and some experimental data.

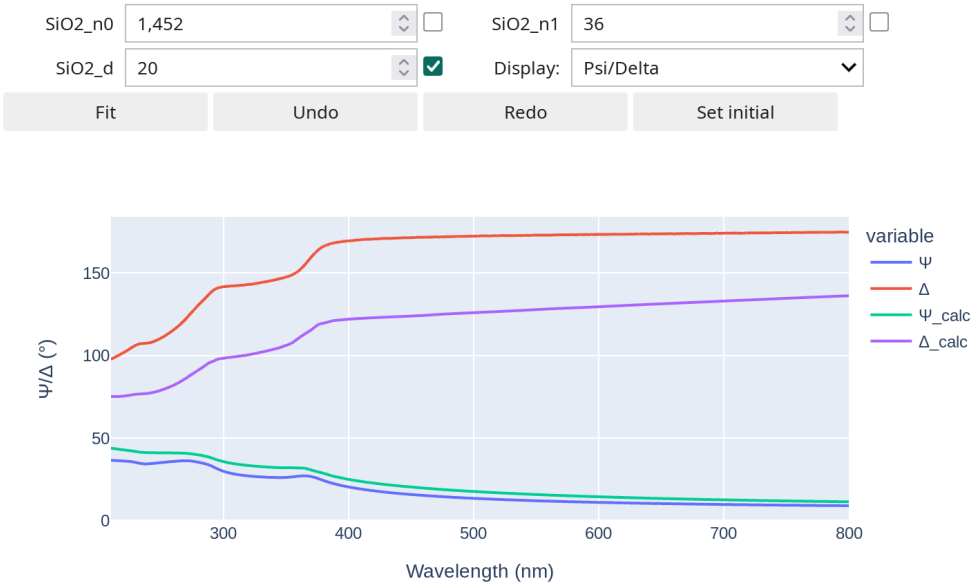

Figure 1: The ipywidgets based fitting GUI.

The examples provide additional information.

Acknowledgements

Financial support is provided by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation), under grant No. 398143140 (FOR 2824). SC also acknowledges the LOEWE

Transferprofessur "HIMAT".

We thank Olivier Castany and Céline Molinaro for their implementation of the Berreman formalism, Steven J. Byrnes for his 2x2 transfer-matrix-method, and Mikhail Polyanskiy for curating the refractiveindex.info database.

References

Bahrem Serhat Danis, E. Z. (2025). *Tmmax: Transfer matrix method with jax* (Version 1.0.0). https://github.com/bahremsd/tmmax

Barker, M., Chue Hong, N. P., Katz, D. S., Lamprecht, A.-L., Martinez-Ortiz, C., Psomopoulos, F., Harrow, J., Castro, L. J., Gruenpeter, M., Martinez, P. A., & Honeyman, T. (2022). Introducing the FAIR principles for research software. *Scientific Data*, *9*(1), 622. https://doi.org/10.1038/s41597-022-01710-x

Bay, M. M., Vignolini, S., & Vynck, K. (2022). PyLlama: A stable and versatile python toolkit for the electromagnetic modelling of multilayered anisotropic media. *Computer Physics Communications*, *273*, 108256. https://doi.org/10.1016/j.cpc.2021.108256

Berreman, D. W. (1972). Optics in stratified and anisotropic media: 4×4-matrix formulation. *J. Opt. Soc. Am.*, *62*(4), 502–510. https://doi.org/10.1364/JOSA.62.000502

Byrnes, S. J. (2020). *Multilayer optical calculations*. https://arxiv.org/abs/1603.02720

Castany, O. (2021). *Berreman4x4*. https://sourceforge.net/projects/berreman4x4/files/documentation.pdf

Castany, O., & Molinaro, C. (2021). *Berreman4x4*. https://github.com/Berreman4x4/Berreman4x4

Eberheim, K., Dues, C., Attaccalite, C., Müller, M. J., Schwan, S., Mollenhauer, D., Chatterjee, S., & Sanna, S. (2022). Tetraphenyl tetrel molecules and molecular crystals: From structural properties to nonlinear optics. *The Journal of Physical Chemistry C*, *126*(7), 3713–3726. https://doi.org/10.1021/acs.jpcc.1c10107

Guide to using WVASE. (2012). [Computer software]. J. A. Woollam Co., Inc.

Harris, C. R., Millman, K. J., Walt, S. J. van der, Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., Kerkwijk, M. H. van, Brett, M., Haldane, A., Río, J. F. del, Wiebe, M., Peterson, P., … Oliphant, T. E. (2020). Array programming with NumPy. *Nature*, *585*(7825), 357–362. https://doi.org/10.1038/s41586-020-2649-2

Hilfiker, J. N., & Tiwald, T. (2018). Dielectric function modeling. In H. Fujiwara & R. W. Collins (Eds.), *Spectroscopic ellipsometry for photovoltaics: Volume 1: Fundamental principles and solar cell characterization* (pp. 115–153). Springer International Publishing. https://doi.org/10.1007/978-3-319-75377-5_5

Langevin, D., Bennet, P., Khaireh-Walieh, A., Wiecha, P., Teytaud, O., & Moreau, A. (2024). PyMoosh: A comprehensive numerical toolkit for computing the optical properties of multilayered structures. *J. Opt. Soc. Am. B*, *41*(2), A67–A78. https://doi.org/10.1364/JOSAB.506175

Luce, A., Mahdavi, A., Marquardt, F., & Wankerl, H. (2022). TMM-fast, a transfer matrix computation package for multilayer thin-film optimization: tutorial. *J. Opt. Soc. Am. A*, *39*(6), 1007–1013. https://doi.org/10.1364/JOSAA.450928

Newville, M., Otten, R., Nelson, A., Stensitzki, T., Ingargiola, A., Allan, D., Fox, A., Carter, F., Michał, Osborn, R., Pustakhod, D., Weigand, S., Ineuhaus, Aristov, A., Glenn, Mark, mgunyho, Deil, C., Hansen, A. L. R., … Persaud, A. (2024). *Lmfit/lmfit-py: 1.3.2* (Version

1.3.2). Zenodo. https://doi.org/10.5281/zenodo.12785036

Passler, N. C., & Paarmann, A. (2017). Generalized 4 × 4 matrix formalism for light propagation in anisotropic stratified media: Study of surface phonon polaritons in polar dielectric heterostructures. *J. Opt. Soc. Am. B*, *34*(10), 2128–2139. https://doi.org/10.1364/JOSAB.34.002128

Passler, N. C., & Paarmann, A. (2019). Generalized 4 × 4 matrix formalism for light propagation in anisotropic stratified media: Study of surface phonon polaritons in polar dielectric heterostructures: erratum. *J. Opt. Soc. Am. B*, *36*(11), 3246–3248. https://doi.org/10.1364/JOSAB.36.003246

Pearce, P. M. (2021). RayFlare: Flexible optical modelling of solar cells. *Journal of Open Source Software*, *6*(65), 3460. https://doi.org/10.21105/joss.03460

Polyanskiy, M. N. (2024). Refractiveindex.info database of optical constants. *Scientific Data*, *11*(1), 94. https://doi.org/10.1038/s41597-023-02898-2

Scheidgen, M., Himanen, L., Ladines, A. N., Sikter, D., Nakhaee, M., Fekete, Á., Chang, T., Golparvar, A., Márquez, J. A., Brockhauser, S., Brückner, S., Ghiringhelli, L. M., Dietrich, F., Lehmberg, D., Denell, T., Albino, A., Näsström, H., Shabih, S., Dobener, F., … Draxl, C. (2023). NOMAD: A distributed web-based platform for managing materials science research data. *Journal of Open Source Software*, *8*(90), 5388. https://doi.org/10.21105/joss.05388

Tompkins, H., & Irene, E. A. (2005). *Handbook of ellipsometry*. William Andrew. ISBN: 9780815517474

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

Wilkinson, M. D., Dumontier, M., Aalbersberg, Ij. J., Appleton, G., Axton, M., Baak, A., Blomberg, N., Boiten, J.-W., Silva Santos, L. B. da, Bourne, P. E., Bouwman, J., Brookes, A. J., Clark, T., Crosas, M., Dillo, I., Dumon, O., Edmunds, S., Evelo, C. T., Finkers, R., … Mons, B. (2016). The FAIR guiding principles for scientific data management and stewardship. *Scientific Data*, *3*(1), 160018. https://doi.org/10.1038/sdata.2016.18