The Journal of Open Source Software

Caustics: A Python Package for Accelerated Strong Gravitational Lensing Simulations

Connor Stone [1,2,3*¶], **Alexandre Adam** [1,2,3*], **Adam Coogan** [1,2,3,†*], **M. J. Yantovski-Barth** [1,2,3], **Andreas Filipp** [1,2,3], **Landung Setiawan** [4], **Cordero Core** [4], **Ronan Legin** [1,2,3], **Charles Wilson** [1,2,3], **Gabriel Missael Barco** [1,2,3], **Yashar Hezaveh** [1,2,3,5,6,7], and **Laurence Perreault-Levasseur** [1,2,3,5,6,7]

1 Ciela Institute - Montréal Institute for Astrophysical Data Analysis and Machine Learning, Montréal, Québec, Canada **2** Department of Physics, Université de Montréal, Montréal, Québec, Canada **3** Mila - Québec Artificial Intelligence Institute, Montréal, Québec, Canada **4** eScience Institute Scientific Software Engineering Center, 1410 NE Campus Pkwy, Seattle, WA 98195, USA **5** Center for Computational Astrophysics, Flatiron Institute, 162 5th Avenue, 10010, New York, NY, USA **6** Perimeter Institute for Theoretical Physics, Waterloo, Canada **7** Trottier Space Institute, McGill University, Montréal, Canada † Work done while at UdeM, Ciela, and Mila ¶ Corresponding author * These authors contributed equally.

DOI: 10.21105/joss.07081

Software

- Review
- Repository
- Archive

Editor: Ivelina Momcheva

Reviewers:

- @andigu
- @Jammy2211

Submitted: 19 June 2024
Published: 13 November 2024

Summary

Gravitational lensing is the deflection of light rays due to the gravity of intervening masses. This phenomenon is observed at a variety of scales and configurations, involving any non-uniform mass such as planets, stars, galaxies, clusters of galaxies, and even the large-scale structure of the Universe. Strong lensing occurs when the distortions are significant and multiple images of the background source are observed. The lens and lensed object(s) must be aligned on the sky within ~ 1 arcsecond for galaxy-galaxy lensing, or tens of arcseconds for cluster-galaxy lensing. As the discovery of lens systems has grown to the low thousands, these systems have become pivotal for precision measurements and addressing critical questions in astrophysics. Notably, they are a tool for understanding a wide range of astrophysical phenomena including: dark matter (e.g. Hezaveh et al., 2016; Vegetti & Vogelsberger, 2014), supernovae (e.g. Rodney et al., 2021), quasars (e.g. Peng et al., 2006), the first stars (e.g. Welch et al., 2022), and the Universe's expansion rate (e.g. K. C. Wong et al., 2020). With future surveys expected to discover hundreds of thousands of lensing systems, the modelling and simulation of such systems must be done at orders of magnitude larger scale than ever before. Here we present `caustics`, a Python package designed to facilitate machine learning and Bayesian methods to handle the extensive computational demands of modelling such a vast number of lensing systems.

Statement of need

The next generation of astronomical surveys, such as the Legacy Survey of Space and Time, the Roman Core Community Surveys, and the Euclid wide survey, are expected to uncover hundreds of thousands of gravitational lenses (Collett, 2015), dramatically increasing the scientific potential of gravitational lensing studies. Currently, analyzing a single lensing system can take several days or weeks, which will be infeasible as the number of known lenses increases by orders of magnitude. Thus, advancements such as computational acceleration via GPUs and/or algorithmic advances such as automatic differentiation are needed to reduce the analysis timescales. Machine learning will be critical to achieve the necessary speed to process these lenses, it will also be needed to meet the complexity of strong lens modelling. caustics is built with the future of lensing in mind, using PyTorch (Paszke et al., 2019) to accelerate the low-level computation and enable deep learning algorithms which rely on automatic differentiation. Automatic differentiation also benefits classical algorithms such as Hamiltonian Monte Carlo (Betancourt, 2017). With these tools available, caustics provides greater than two orders of magnitude acceleration to most standard operations, enabling previously impractical analyses at scale.

Several other simulation packages for strong gravitational lensing are already publicly available. The well-established lenstronomy package has been in use since 2018 (Birrer et al., 2021); GLAMER is a C++-based code for modelling complex and large dynamic range fields (Metcalf & Petkova, 2014); PyAutoLens is also widely used (Nightingale et al., 2021); GIGA-Lens is a specialized JAX (Bradbury et al., 2018) based gravitational lensing package (Gu et al., 2022); and Herculens is a more general JAX based lensing simulator package (Galan et al., 2022); among others (GRAVLENS Keeton, 2011; LENSTOOL Kneib et al., 2011; SLITRONOMY Galan et al., 2021; paltax Wagner-Carena et al., 2024). There are also several in-house codes developed for specialized analysis which are then not publicly released (e.g. Suyu & Halkola, 2010). The development of caustics has been primarily focused on three aspects: processing speed, user experience, and flexibility. The code is optimized to fully exploit PyTorch's capabilities, significantly enhancing processing speed. The user experience is streamlined by providing three interfaces to the code: configuration file, object-oriented, and functional, where each interface level requires more expertise but allows more capabilities. In this way, users with all levels of gravitational lensing simulation experience can effectively engage with the software. Flexibility is achieved by a determined focus on minimalism in the core functionality of caustics. All of these elements combine to make caustics a capable lensing simulator to support machine learning applications, and classical analysis.

Caustics fills a timely need for a differentiable lensing simulator. Several other fields have already benefited from such simulators, for example: gravitational wave analysis (Coogan et al., 2022; Edwards et al., 2023; K. W. K. Wong et al., 2023); astronomical image photometry (Stone et al., 2023); point spread function modelling (Desdoigts et al., 2023); time series analysis (Million et al., 2024); and even generic optimization for scientific problems (Nikolic, 2018). Caustics is built on lessons from other differentiable codes, with the goal of enabling machine learning techniques in the field of strong lensing. With caustics it will now be possible to analyze over 100,000 lenses in a timely manner (Hezaveh et al., 2017; Perreault Levasseur et al., 2017).

Scope

Caustics is a gravitational lensing simulator. The purpose of the project is to streamline the simulation of strong gravitational lensing effects on the light of a background source. The primary focus is on all transformations between the source plane(s) and the image plane through the lensing plane(s). There is minimal effort on modelling the observational elements of atmosphere, telescope optics, detector effects with most of these left to the users. A variety of parametric lensing profiles are included, such as: Singular Isothermal Ellipsoid (SIE), Elliptical Power Law (EPL), Pseudo-Jaffe, Navarro-Frenk-White (NFW), and External Shear. Additionally, it offers non-parametric representations such as a gridded convergence or a potential field. For the background source caustics provides a Sérsic light profile, as well as a pixelized light image. Users can easily extend these lens and source lists using templates provided in the documentation.

Once a lensing system has been defined, caustics can then perform various computational operations on the system such as raytracing through the lensing system, forwards and backwards. Users can compute the lensing potential, convergence, deflection field, time delay field, and magnification. All of these operations can readily be performed in a multi-plane setting to account for interlopers or multiple sources. Because the code is differentiable (via PyTorch), one can easily also compute the derivatives of these quantities, such as when finding critical curves or computing the Jacobian of the lens equation. For example, one can project interlopers from multiple lensing planes to a single plane by computing the effective convergence, obtained with the trace of the Jacobian.

With these building blocks in place, one can construct fast and accurate simulators used to produce training sets for machine learning models or for inference on real-world systems. Neural networks have become a widespread tool for amortized inference of gravitational lensing parameter (Hezaveh et al., 2017) or in the detection of gravitational lenses (Huang et al., 2021; Petrillo et al., 2017), but they require large and accurate training sets which can be created quickly with `caustics`. A demonstration of such a simulator is given in Figure 1 which also demonstrates the importance of sub-pixel sampling. This involves raytracing through the lensing mass and extracting the brightness of the background source. Further, the image then must be convolved with a PSF for extra realism. All of these operations are collected into a single simulator which users can access and use simply as a function of the relevant lensing and light source parameters. Because `caustics` is written in `PyTorch`, its simulators are differentiable, thus one can compute gradients through the forward model. This enables machine learning algorithms such as recurrent inference machines (Adam et al., 2023) and diffusion models (Adam et al., 2022).

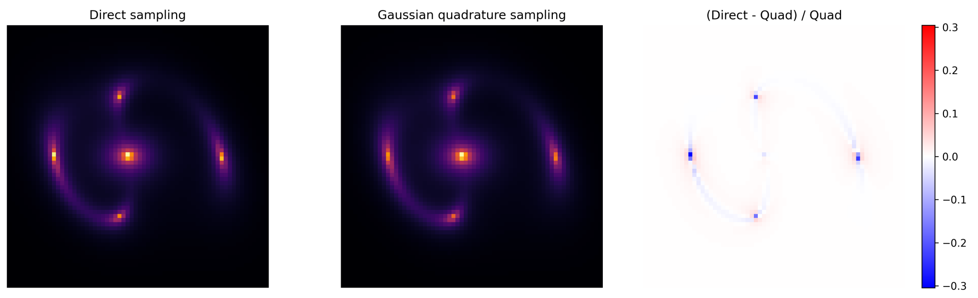

Figure 1: Example simulated gravitational lens system defined by a Sérsic source, SIE lens mass, and Sérsic lens light. Left, the pixel map is sampled only at the midpoint of each pixel. Middle, the pixel map is supersampled and then integrated using gaussian quadrature integration for greater accuracy. Right, the fractional difference between the two is shown. We can see that in this case the midpoint sampling is inaccurate by up to 30% of the pixel value in areas of high contrast. The exact inaccuracy depends greatly on the exact configuration.

The current scope of `caustics` does not include weak lensing, microlensing, or cluster scale lensing simulations. While the underlying mathematical frameworks are similar, the specific techniques commonly used in these areas are not yet implemented, although they represent an avenue for future development.

The scope of `caustics` ends at lensing simulation, thus it does not include functionality to optimize or sample the resulting functions. Users are encouraged to use already existing optimization and sampling codes such as `scipy.optimize` (Virtanen et al., 2020), `emcee` (Foreman-Mackey et al., 2013), `dynesty` (Speagle, 2020), `Pyro` (Bingham et al., 2019), and `torch.optim` (Paszke et al., 2019). Interfacing with these codes is easy and demonstrations are included in the documentation.

Performance

Here we discuss the performance enhancements enabled by `caustics`. The code allows operations to be batched, multi-threaded on CPUs, or offloaded to GPUs to optimize computational efficiency (via `PyTorch`). In Figure 2 we demonstrate this by sampling images of a Sérsic with an SIE model lensing the image (much like Figure 1). For CPU calculations we use an

Intel Gold 6148 Skylake and for the GPU we use an NVIDIA V100, all tests were done at 64 bit precision. In the two subfigures we show the performance for sampling a 128x128 image using the pixel midpoint (left), and sampling a "realistic" image (right) which is upsampled by a factor of 4 and convolved with a PSF. The "caustics unbached CPU" line means the "Number of samples" (x-axis) were computed by using a Python for-loop. The "caustics batched CPU/4CPU" lines are determined by sending all "Number of samples" through the simulator simultaneously using PyTorch vmap on a single CPU or on four CPUs respectively. The "caustics batched gpu" line is determined similarly, except that the computations are performed on the GPU. All parameters are randomly resampled for each simulation to avoid caching effects. This demonstrates a number of interesting facts about numerical performance in such scenarios.

We compare the performance with that of lenstronomy as our baseline. The most direct comparison between the two codes can be observed by comparing the lenstronomy line with the "caustics unbatched CPU" line. lenstronomy is written using the numba (Lam et al., 2015) package which compiles Python code into lower level C code. The left plot shows that caustics suffers from a significant overhead compared with lenstronomy, which is nearly twice as fast as the "caustics unbatched CPU" line. This occurs because the pure Python (interpreted language) elements of caustics are much slower than the C/Cuda PyTorch backends (compiled language). This is most pronounced when fewer computations are needed to perform a simulation. Despite this overhead, caustics showcases a strong performance when using the batched GPU setting, especially in the more realistic scenario with extra computations in the simulator including 4x oversampling of the raytracing and the PSF convolution.

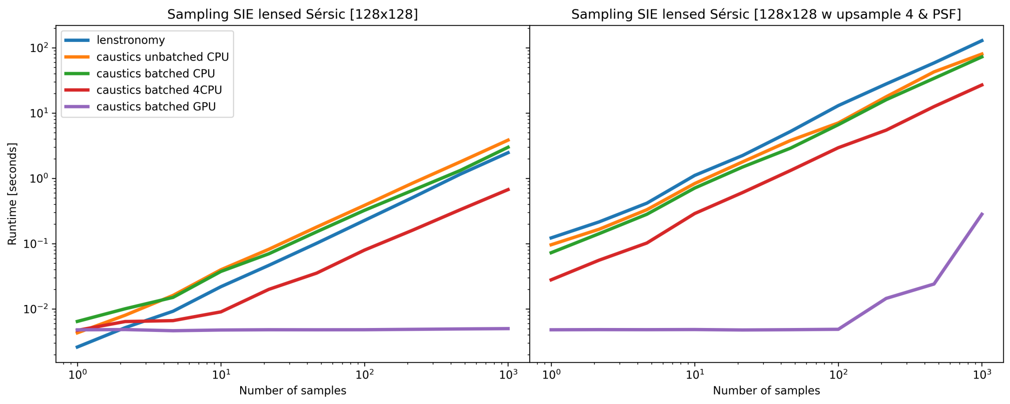

Figure 2: Runtime comparisons for a simple lensing setup. We compare the amount of time taken (y-axis) to generate a certain number of lensing realizations (x-axis) where a Sérsic model is lensed by an SIE mass distribution. For CPU calculations we use Intel Gold 6148 Skylake and for the GPU we use a NVIDIA V100, all tests were done at 64 bit precision. On the left, the lensing system is sampled 128 pixel resolution only at pixel midpoints. On the right, a more realistic simulation includes upsampled pixels and PSF convolution. From the two tests we see varying performance enhancements from compiled, unbatched, batched, multi-threaded, and GPU processing setups.

Comparing the "caustics unbatched CPU" and "caustics batched CPU" lines we see that batching can provide more efficient use of the same, single CPU, computational resources. However, in the realistic scenario the batching has minimal performance enhancement, likely because the Python overhead of a for-loop is no longer significant compared to the large number of numerical operations being performed.

Comparing "caustics batched CPU" and "caustics batched 4CPU" we see that PyTorch's automatic multi-threading capabilities can indeed provide performance improvements. However, the improvement is not a direct multiple of the number of CPUs due to overheads. For tasks that are "embarrassingly parallel," such as running multiple MCMC chains, it is more effective to parallelize at the job level rather than at the thread level to avoid these overheads.

The most dramatic improvements are observed when comparing any CPU operations with "caustics batched gpu". Although communication between the CPU and GPU can be slow, consolidating calculations into fewer, larger batches allow `caustics` to fully exploit GPU capabilities. In the midpoint sampling, the GPU never "saturates" meaning that it runs equally fast for any number of samples. In the realistic scenario, we reach the saturation limit of the GPU memory at 100 samples and can no longer simultaneously model all the systems, we thus enter a linear regime in runtime just like the CPU for any number of simulations. The V100 GPUs have 16 GB of memory, with 100 images at 128x128 resolution, upsampled on each axis by 4 times (16 times the memory), and 64bit precision each operation requires approximately 200MB of memory. Since gravitational lensing requires a number of intermediate calculations[1] such as computing FFTs for convolution, plus PyTorch overheads, this fills the GPU memory. An A100 GPU with 80GB of memory can remain in the flat regime much further before saturation, giving even further performance improvements over CPU computations. Nonetheless, it is possible to easily achieve roughly 1000X speedup over CPU performance, making GPUs by far the most efficient method to perform large lensing computations such as running many MCMC chains or sampling many lensing realizations (e.g. for training machine learning models).

User experience

Caustics offers a tiered interface system designed to cater to users with varying levels of expertise in gravitational lensing simulation. This section outlines the three levels of interfaces that enhance the user experience by providing different degrees of complexity and flexibility.

Configuration file interface: The most accessible level of interaction is through configuration files. Users can define simulators in `.yaml` format, specifying parameters such as lens models, light source characteristics, and image processing details like PSF convolution and sub-pixel integration. The users can then load such a simulator in a single line of Python and carry on using that simulator as a pure function `f(x)` which takes in parameters such as the Sérsic index, position, SIE Einstein radius, etc. and returns an image. This interface is straightforward for new users and for simplifying the sharing of simulation configurations between users.

Object-oriented interface: This intermediate level allows users to manipulate lenses and light sources as objects. The users can build simulators just like the configuration file interface, or they can interact with the objects in a number of other ways accessing further details about each lens. Each lensing object has (where meaningful) a convergence, potential, time delay, and deflection field plus any associated quantities such as magnification. We provide examples to visualize all of these. Users can apply the full flexibility of Python to these lensing objects and construct customized analysis code, although there are many default routines which enable one to quickly perform typical analysis tasks.

For both the object-oriented and configuration file interfaces, the final simulator object can be analyzed in a number of ways, Figure 3 demonstrates how one can investigate the structure of a simulator in the form of a directed acyclic graph of calculations. Note that one can also fix a subset of parameter values, making them "static" instead of the default, which is "dynamic". Users can produce such a graph representation for any `caustics` simulator.

[1]"Kernlizing" operations by packing multiple mathematical operations into a single call to the GPU can both reduce the memory load and increase the speed of such calculations. This is an avenue for further development for caustics.

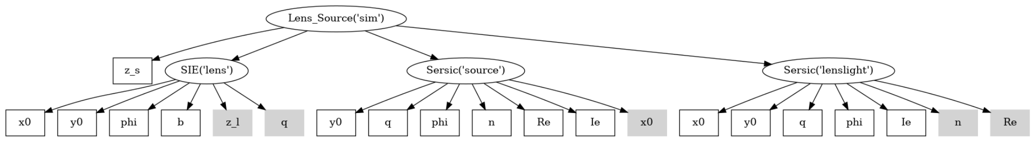

Figure 3: Example directed acyclic graph representation of the simulator from Figure 2. Ellipses are `caustics` objects and squares are parameters; open squares are dynamic parameters and greyed squares are static parameters. Parameters are passed at the top level node (`Lens_Source`) and flow down the graph automatically to all other objects which require parameter values to complete a lensing simulation.

Functional interface: The functional interface avoids the object-oriented `caustics` code, instead giving the users access to individual mathematical operations related to lensing, most of which are drawn directly from gravitational lensing literature. All such functions include references in their documentation to the relevant papers and equation numbers from which they are derived. These equations have been tested and implemented in a reasonably efficient manner. This interface is ideal for researchers and developers looking to experiment with novel lensing techniques or to modify existing algorithms while leveraging robust, pre-tested components.

Each layer is in fact built on the one below it, making the transition from one to the other a matter of following documentation and code references. This makes the transition easy because one can very clearly see how their current analysis can be reproduced in the lower level. `Caustics` thus provides a straightforward pipeline for users to move from beginner to expert. Users at all levels are encouraged to investigate the documentation as the code includes extensive information for all methods, including units for most functions. This transparency not only aids in understanding and utilizing the functions correctly but also enhances the reliability and educational value of the software.

Flexibility

The flexibility of caustics is fundamentally linked to its design philosophy, which is focused on providing a robust yet adaptable framework for gravitational lensing simulations. A focus on minimalism in the core functionality means that research-ready analysis routines must be built by users. To facilitate this, our Jupyter notebook tutorials include examples of many typical analysis tasks, with the details laid out for the users so they can simply copy and modify to suit their particular analysis task. Thus, we achieve flexibility both by allowing many analysis paradigms, and by supporting the easy development of production code. With the simulator paradigm designed to produce simple functions (typically with the form `f(x)`) it is easy to interface `caustics` simulators with any other Python packages. Users are not constrained to pre-built analysis routines or design decisions and are instead encouraged to extend and interface with other packages.

Research is an inherently dynamic process and gravitational lensing is an evolving field. Designing flexible codes for such environments ensures long-lasting relevance. `Caustics` is well positioned to grow and evolve with the needs of the community.

Machine Learning

One of the core purposes of `caustics` is to advance the application of machine learning to strong gravitational lensing analysis. This is accomplished through two avenues. First, as demonstrated in Figure 2, `caustics` efficiently generates large samples of simulated mock lensing images by leveraging GPUs. Since many machine learning algorithms are “data hungry”, this translates to better performance with more examples to learn from. Literature on machine learning applications in strong gravitational lensing underscores the benefits of this generation capacity (Brehmer et al., 2019; Chianese et al., 2020; Coogan et al., 2020; Karchev, Coogan, et al., 2022; Karchev, Anau Montel, et al., 2022; Mishra-Sharma & Yang, 2022). Second, the differentiable nature of `caustics` allows it to be integrated directly into machine learning workflows. This could mean using `caustics` as part of a loss function. Alternatively, it could be through a statistical paradigm like diffusion modelling, in which `caustics` would be directly integrated in the sampling procedure. It has already been shown that differentiable lensing simulators, coupled with machine learning and diffusion modelling, can massively improve source reconstruction in strong gravitational lenses (Adam et al., 2022) and in weak lensing (Remy et al., 2023).

Conclusions

Here we have presented `caustics`, a gravitational lensing simulator framework which allows for greater than 100 times speedup over traditional CPU implementations by leveraging GPU resources. `Caustics` is fully-featured, meaning one can straightforwardly model any strong lensing system with state-of-the-art techniques. The code and documentation facilitate the transition of users from beginner to expert by providing three interfaces which allow increasingly more flexibility in how one wishes to model a lensing system. `Caustics` is designed to be the gravitational lensing simulator of the future and to meet the hundreds of thousands of lenses soon to be discovered with modern computational resources.

Acknowledgements

This research was enabled by a generous donation by Eric and Wendy Schmidt with the recommendation of the Schmidt Futures Foundation. We acknowledge the generous software support of the University of Washington Scientific Software Engineering Center (SSEC) at the eScience Institute, via matching through the Schmidt Futures Virtual Institute for Scientific Software (VISS). CS acknowledges the support of a NSERC Postdoctoral Fellowship and a CITA National Fellowship. This research was enabled in part by support provided by Calcul Québec and the Digital Research Alliance of Canada. The work of A.A. and R.L. was partially funded by NSERC CGS D scholarships. Y.H. and L.P. acknowledge support from the National Sciences and Engineering Council of Canada grants RGPIN-2020-05073 and 05102, the Fonds de recherche du Québec grants 2022-NC-301305 and 300397, and the Canada Research Chairs Program. Thanks to Simon Birrer for communications regarding benchmarking of `lenstronomy`. Thanks to James Nightingale and Andi Gu for thoughtful reviews of the codebase, and thanks to Ivelina Momcheva for a thorough editorial review.

References

Adam, A., Coogan, A., Malkin, N., Legin, R., Perreault-Levasseur, L., Hezaveh, Y., & Bengio, Y. (2022). Posterior samples of source galaxies in strong gravitational lenses with score-based priors. *Machine Learning and the Physical Sciences Workshop*, E1. https://doi.org/10.48550/arXiv.2211.03812

Adam, A., Perreault-Levasseur, L., Hezaveh, Y., & Welling, M. (2023). Pixelated Reconstruction of Foreground Density and Background Surface Brightness in Gravitational Lensing Systems Using Recurrent Inference Machines. *The Astrophysical Journal*, *951*(1), 6. https://doi.org/10.3847/1538-4357/accf84

Betancourt, M. (2017). A Conceptual Introduction to Hamiltonian Monte Carlo. *arXiv e-Prints*, arXiv:1701.02434. https://doi.org/10.48550/arXiv.1701.02434

Bingham, E., Chen, J. P., Jankowiak, M., Obermeyer, F., Pradhan, N., Karaletsos, T., Singh, R., Szerlip, P., Horsfall, P., & Goodman, N. D. (2019). Pyro: Deep universal probabilistic programming. *The Journal of Machine Learning Research*, *20*(1), 973–978.

Birrer, S., Shajib, A. J., Gilman, D., Galan, A., Aalbers, J., Million, M., Morgan, R., Pagano, G., Park, J. W., Teodori, L., Tessore, N., Ueland, M., Vyvere, L. V. de, Wagner-Carena, S., Wempe, E., Yang, L., Ding, X., Schmidt, T., Sluse, D., … Amara, A. (2021). Lenstronomy II: A gravitational lensing software ecosystem. *Journal of Open Source Software*, *6*(62), 3283. https://doi.org/10.21105/joss.03283

Bradbury, J., Frostig, R., Hawkins, P., Johnson, M. J., Leery, C., Maclaurin, D., Necula, G., Paszke, A., VanderPlas, J., Wanderman-Milne, S., & Zhang, Q. (2018). *JAX: Composable transformations of Python+NumPy programs* (Version 0.3.13). http://github.com/google/jax

Brehmer, J., Mishra-Sharma, S., Hermans, J., Louppe, G., & Cranmer, K. (2019). Mining for Dark Matter Substructure: Inferring Subhalo Population Properties from Strong Lenses with Machine Learning. *The Astrophysical Journal*, *886*(1), 49. https://doi.org/10.3847/1538-4357/ab4c41

Chianese, M., Coogan, A., Hofma, P., Otten, S., & Weniger, C. (2020). Differentiable strong lensing: uniting gravity and neural nets through differentiable probabilistic programming. *Monthly Notices of the RAS*, *496*(1), 381–393. https://doi.org/10.1093/mnras/staa1477

Collett, T. E. (2015). THE POPULATION OF GALAXY–GALAXY STRONG LENSES IN FORTHCOMING OPTICAL IMAGING SURVEYS. *The Astrophysical Journal*, *811*(1), 20. https://doi.org/10.1088/0004-637X/811/1/20

Coogan, A., Edwards, T. D. P., Chia, H. S., George, R. N., Freese, K., Messick, C., Setzer, C. N., Weniger, C., & Zimmerman, A. (2022). Efficient gravitational wave template bank generation with differentiable waveforms. *Physical Review D*, *106*(12), 122001. https://doi.org/10.1103/PhysRevD.106.122001

Coogan, A., Karchev, K., & Weniger, C. (2020). Targeted Likelihood-Free Inference of Dark Matter Substructure in Strongly-Lensed Galaxies. *arXiv e-Prints*, arXiv:2010.07032. https://doi.org/10.48550/arXiv.2010.07032

Desdoigts, L., Pope, B. J. S., Dennis, J., & Tuthill, P. G. (2023). Differentiable optics with Lux: I—deep calibration of flat field and phase retrieval with automatic differentiation. *Journal of Astronomical Telescopes, Instruments, and Systems*, *9*(2), 028007. https://doi.org/10.1117/1.JATIS.9.2.028007

Edwards, T. D. P., Wong, K. W. K., Lam, K. K. H., Coogan, A., Foreman-Mackey, D., Isi, M., & Zimmerman, A. (2023). ripple: Differentiable and Hardware-Accelerated Waveforms for Gravitational Wave Data Analysis. *arXiv e-Prints*, arXiv:2302.05329. https://doi.org/10.48550/arXiv.2302.05329

Foreman-Mackey, D., Hogg, D. W., Lang, D., & Goodman, J. (2013). emcee: The MCMC Hammer. *Publications of the ASP*, *125*(925), 306. https://doi.org/10.1086/670067

Galan, A., Peel, A., Joseph, R., Courbin, F., & Starck, J.-L. (2021). SLITRONOMY: Towards a fully wavelet-based strong lensing inversion technique. *Astronomy and Astrophysics*, *647*, A176. https://doi.org/10.1051/0004-6361/202039363

Galan, A., Vernardos, G., Peel, A., Courbin, F., & Starck, J.-L. (2022). Using wavelets to capture deviations from smoothness in galaxy-scale strong lenses. *Astronomy and Astrophysics*, *668*, A155. https://doi.org/10.1051/0004-6361/202244464

Gu, A., Huang, X., Sheu, W., Aldering, G., Bolton, A. S., Boone, K., Dey, A., Filipp, A., Jullo, E., Perlmutter, S., Rubin, D., Schlafly, E. F., Schlegel, D. J., Shu, Y., & Suyu, S. H. (2022). GIGA-Lens: Fast Bayesian Inference for Strong Gravitational Lens Modeling. *The Astrophysical Journal*, *935*(1), 49. https://doi.org/10.3847/1538-4357/ac6de4

Hezaveh, Y., Dalal, N., Holder, G., Kisner, T., Kuhlen, M., & Perreault Levasseur, L. (2016).

Measuring the power spectrum of dark matter substructure using strong gravitational lensing. *Journal of Cosmology and Astroparticle Physics*, *2016*(11), 048. https://doi.org/10.1088/1475-7516/2016/11/048

Hezaveh, Y., Perreault Levasseur, L., & Marshall, P. J. (2017). Fast automated analysis of strong gravitational lenses with convolutional neural networks. *Nature*, *548*(7669), 555–557. https://doi.org/10.1038/nature23463

Huang, X., Storfer, C., Gu, A., Ravi, V., Pilon, A., Sheu, W., Venguswamy, R., Banka, S., Dey, A., Landriau, M., Lang, D., Meisner, A., Moustakas, J., Myers, A. D., Sajith, R., Schlafly, E. F., & Schlegel, D. J. (2021). Discovering New Strong Gravitational Lenses in the DESI Legacy Imaging Surveys. *The Astrophysical Journal*, *909*(1), 27. https://doi.org/10.3847/1538-4357/abd62b

Karchev, K., Anau Montel, N., Coogan, A., & Weniger, C. (2022). Strong-Lensing Source Reconstruction with Denoising Diffusion Restoration Models. *arXiv e-Prints*, arXiv:2211.04365. https://doi.org/10.48550/arXiv.2211.04365

Karchev, K., Coogan, A., & Weniger, C. (2022). Strong-lensing source reconstruction with variationally optimized Gaussian processes. *Monthly Notices of the RAS*, *512*(1), 661–685. https://doi.org/10.1093/mnras/stac311

Keeton, C. R. (2011). *GRAVLENS: Computational Methods for Gravitational Lensing*. Astrophysics Source Code Library, record ascl:1102.003.

Kneib, J.-P., Bonnet, H., Golse, G., Sand, D., Jullo, E., & Marshall, P. (2011). *LENSTOOL: A Gravitational Lensing Software for Modeling Mass Distribution of Galaxies and Clusters (strong and weak regime)*. Astrophysics Source Code Library, record ascl:1102.004.

Lam, S. K., Pitrou, A., & Seibert, S. (2015). Numba: A llvm-based python jit compiler. *Proceedings of the Second Workshop on the LLVM Compiler Infrastructure in HPC*, 1–6.

Metcalf, R. B., & Petkova, M. (2014). GLAMER - I. A code for gravitational lensing simulations with adaptive mesh refinement. *Monthly Notices of the RAS*, *445*(2), 1942–1953. https://doi.org/10.1093/mnras/stu1859

Million, M., Michalewicz, K., Dux, F., Courbin, F., & Marshall, P. J. (2024). Image deconvolution and PSF reconstruction with STARRED: a wavelet-based two-channel method optimized for light curve extraction. *arXiv e-Prints*, arXiv:2402.08725. https://doi.org/10.48550/arXiv.2402.08725

Mishra-Sharma, S., & Yang, G. (2022). Strong Lensing Source Reconstruction Using Continuous Neural Fields. *Machine Learning for Astrophysics*, 34. https://doi.org/10.48550/arXiv.2206.14820

Nightingale, James. W., Hayes, R. G., Kelly, A., Amvrosiadis, A., Etherington, A., He, Q., Li, N., Cao, X., Frawley, J., Cole, S., Enia, A., Frenk, C. S., Harvey, D. R., Li, R., Massey, R. J., Negrello, M., & Robertson, A. (2021). 'PyAutoLens': Open-source strong gravitational lensing. *Journal of Open Source Software*, *6*(58), 2825. https://doi.org/10.21105/joss.02825

Nikolic, B. (2018). Acceleration of Non-Linear Minimisation with PyTorch. *arXiv e-Prints*, arXiv:1805.07439. https://doi.org/10.48550/arXiv.1805.07439

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., … Chintala, S. (2019). PyTorch: An imperative style, high-performance deep learning library. In *Advances in neural information processing systems 32* (pp. 8024–8035). Curran Associates, Inc. http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

Peng, C. Y., Impey, C. D., Rix, H.-W., Kochanek, C. S., Keeton, C. R., Falco, E. E., Lehár, J., & McLeod, B. A. (2006). Probing the Coevolution of Supermassive Black Holes and Galaxies Using Gravitationally Lensed Quasar Hosts. *The Astrophysical Journal*, *649*(2), 616–634. https://doi.org/10.1086/506266

Perreault Levasseur, L., Hezaveh, Y. D., & Wechsler, R. H. (2017). Uncertainties in Parameters Estimated with Neural Networks: Application to Strong Gravitational Lensing. *The Astrophysical Journal Letters*, *850*(1), L7. https://doi.org/10.3847/2041-8213/aa9704

Petrillo, C. E., Tortora, C., Chatterjee, S., Vernardos, G., Koopmans, L. V. E., Verdoes Kleijn, G., Napolitano, N. R., Covone, G., Schneider, P., Grado, A., & McFarland, J. (2017). Finding strong gravitational lenses in the Kilo Degree Survey with Convolutional Neural Networks. *Monthly Notices of the RAS*, *472*(1), 1129–1150. https://doi.org/10.1093/mnras/stx2052

Remy, B., Lanusse, F., Jeffrey, N., Liu, J., Starck, J.-L., Osato, K., & Schrabback, T. (2023). Probabilistic mass-mapping with neural score estimation. *Astronomy and Astrophysics*, *672*, A51. https://doi.org/10.1051/0004-6361/202243054

Rodney, S. A., Brammer, G. B., Pierel, J. D. R., Richard, J., Toft, S., O'Connor, K. F., Akhshik, M., & Whitaker, K. E. (2021). A gravitationally lensed supernova with an observable two-decade time delay. *Nature Astronomy*, *5*, 1118–1125. https://doi.org/10.1038/s41550-021-01450-9

Speagle, J. S. (2020). DYNESTY: a dynamic nested sampling package for estimating Bayesian posteriors and evidences. *Monthly Notices of the RAS*, *493*(3), 3132–3158. https://doi.org/10.1093/mnras/staa278

Stone, C. J., Courteau, S., Cuillandre, J.-C., Hezaveh, Y., Perreault-Levasseur, L., & Arora, N. (2023). ASTROPHOT: fitting everything everywhere all at once in astronomical images. *Monthly Notices of the RAS*, *525*(4), 6377–6393. https://doi.org/10.1093/mnras/stad2477

Suyu, S. H., & Halkola, A. (2010). The halos of satellite galaxies: the companion of the massive elliptical lens SL2S J08544-0121. *Astronomy and Astrophysics*, *524*, A94. https://doi.org/10.1051/0004-6361/201015481

Vegetti, S., & Vogelsberger, M. (2014). On the density profile of dark matter substructure in gravitational lens galaxies. *Monthly Notices of the RAS*, *442*(4), 3598–3603. https://doi.org/10.1093/mnras/stu1284

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

Wagner-Carena, S., Lee, J., Pennington, J., Aalbers, J., Birrer, S., & Wechsler, R. H. (2024). A Strong Gravitational Lens Is Worth a Thousand Dark Matter Halos: Inference on Small-Scale Structure Using Sequential Methods. *arXiv e-Prints*, arXiv:2404.14487. https://doi.org/10.48550/arXiv.2404.14487

Welch, B., Coe, D., Zackrisson, E., de Mink, S. E., Ravindranath, S., Anderson, J., Brammer, G., Bradley, L., Yoon, J., Kelly, P., Diego, J. M., Windhorst, R., Zitrin, A., Dimauro, P., Jiménez-Teja, Y., Abdurro'uf, Nonino, M., Acebron, A., Andrade-Santos, F., … Vikaeus, A. (2022). JWST Imaging of Earendel, the Extremely Magnified Star at Redshift z = 6.2. *The Astrophysical Journal Letters*, *940*(1), L1. https://doi.org/10.3847/2041-8213/ac9d39

Wong, K. C., Suyu, S. H., Chen, G. C.-F., Rusu, C. E., Million, M., Sluse, D., Bonvin, V., Fassnacht, C. D., Taubenberger, S., Auger, M. W., Birrer, S., Chan, J. H. H., Courbin, F., Hilbert, S., Tihhonova, O., Treu, T., Agnello, A., Ding, X., Jee, I., … Meylan, G. (2020). H0LiCOW - XIII. A 2.4 per cent measurement of H_0 from lensed quasars: 5.3σ tension between early- and late-Universe probes. *Monthly Notices of the RAS*, *498*(1), 1420–1439. https://doi.org/10.1093/mnras/stz3094

Wong, K. W. K., Isi, M., & Edwards, T. D. P. (2023). Fast Gravitational-wave Parameter Estimation without Compromises. *The Astrophysical Journal*, *958*(2), 129. https://doi.org/10.3847/1538-4357/acf5cd