Title
GIGA-Lens: Fast Bayesian Inference for Strong Gravitational Lens Modeling

Journal
The Astrophysical Journal, 935(1)

ISSN
0004-637X

Authors
Gu, A
Huang, X
Sheu, W
et al.

Publication Date
2022-08-01

DOI
10.3847/1538-4357/ac6de4

The Astrophysical Journal, 935:49 (17pp), 2022 August 10 https://doi.org/10.3847/1538-4357/ac6de4

GIGA-Lens: Fast Bayesian Inference for Strong Gravitational Lens Modeling

A. Gu[1,2], X. Huang[3,4], W. Sheu[1,2], G. Aldering[4], A. S. Bolton[5], K. Boone[6], A. Dey[5], A. Filipp[7,8], E. Jullo[9], S. Perlmutter[1,4], D. Rubin[10], E. F. Schlafly[11], D. J. Schlegel[4], Y. Shu[8,12], and S. H. Suyu[7,8,13]
[1] Department of Physics, University of California, Berkeley, Berkeley, CA 94720, USA; andi.gu@berkeley.edu
[2] Department of Electrical Engineering & Computer Sciences, University of California, Berkeley, Berkeley, CA 94720, USA
[3] Department of Physics & Astronomy, University of San Francisco, San Francisco, CA 94117, USA; xhuang22@usfca.edu
[4] Physics Division, Lawrence Berkeley National Laboratory, 1 Cyclotron Road, Berkeley, CA 94720, USA
[5] NSF's National Optical-Infrared Astronomy Research Laboratory, 950 N. Cherry Avenue, Tucson, AZ 85719, USA
[6] DiRAC Institute, Department of Astronomy, University of Washington, 3910 15th Ave NE, Seattle, WA 98195, USA
[7] Technische Universität München, Physik-Department, James-Franck-Straße 1, D-85748 Garching, Germany
[8] Max-Planck-Institut für Astrophysik, Karl-Schwarzschild-Str. 1, D-85748 Garching, Germany
[9] Aix-Marseille Univ, CNRS, CNES, LAM, Marseille, France
[10] Department of Physics & Astronomy, University of Hawaii, Honolulu, HI 96822, USA
[11] Lawrence Livermore National Laboratory, 7000 East Avenue, Livermore, CA 94550, USA
[12] Ruhr University Bochum, Faculty of Physics and Astronomy, Astronomical Institute (AIRUB), German Centre for Cosmological Lensing, D-44780 Bochum, Germany
[13] Academia Sinica Institute of Astronomy and Astrophysics (ASIAA), 11F of ASMAB, No. 1, Section 4, Roosevelt Road, Taipei 10617, Taiwan
Received 2022 February 15; accepted 2022 March 17; published 2022 August 12

Abstract

We present GIGA-Lens: a gradient-informed, GPU-accelerated Bayesian framework for modeling strong gravitational lensing systems, implemented in TensorFlow and JAX. The three components, optimization using multistart gradient descent, posterior covariance estimation with variational inference, and sampling via Hamiltonian Monte Carlo, all take advantage of gradient information through automatic differentiation and massive parallelization on graphics processing units (GPUs). We test our pipeline on a large set of simulated systems and demonstrate in detail its high level of performance. The average time to model a single system on four Nvidia A100 GPUs is 105 s. The robustness, speed, and scalability offered by this framework make it possible to model the large number of strong lenses found in current surveys and present a very promising prospect for the modeling of $\mathcal{O}(10^5)$ lensing systems expected to be discovered in the era of the Vera C. Rubin Observatory, Euclid, and the Nancy Grace Roman Space Telescope.

Unified Astronomy Thesaurus concepts: GPU computing (1969); Markov chain Monte Carlo (1889); Strong gravitational lensing (1643); Einstein rings (451); Bayesian statistics (1900); Cosmology (343)

1. Introduction

Strong gravitational lensing systems are a powerful tool for cosmology. They have been used to study how dark matter is distributed in galaxies and clusters (e.g., Kochanek 1991; Hogg & Blandford 1994; Broadhurst et al. 2000; Koopmans & Treu 2002; Bolton et al. 2006; Koopmans et al. 2006; Bradač et al. 2008; Huang et al. 2009; Vegetti & Koopmans 2009; Jullo et al. 2010; Grillo et al. 2015; Shu et al. 2015, 2016, 2017; Meneghetti et al. 2020) and are uniquely suited to probe the low end of the dark matter mass function and test the prediction of the cold dark matter (CDM) model beyond the local universe (e.g., Vegetti et al. 2010, 2012; Hezaveh et al. 2016; Ritondale et al. 2019; Diaz Rivero & Dvorkin 2020; Cağan Sengül et al. 2020, 2021; Gilman et al. 2021). Multiply lensed supernovae (SNe) are ideal for measuring time delays and H_0 because of their well-characterized light curves, and in the case of Type Ia, with the added benefit of standardizable luminosity (Refsdal 1964; Treu 2010; Oguri & Marshall 2010), provided microlensing can be accurately characterized (Yahalomi et al. 2017; Foxley-Marrable et al. 2018). Furthermore, SNe have the benefit of fading, so for these systems lens models can be validated using images that are uncontaminated by bright point sources (Ding et al. 2021). In recent years, strongly lensed SNe, both core-collapse (Kelly et al. 2015; Rodney et al. 2016) and Type Ia (Quimby et al. 2014; Goobar et al. 2017; Rodney et al. 2021), have been discovered. Time-delay H_0 measurements from multiply imaged SNe (e.g., Goldstein & Nugent 2017; Shu et al. 2018; Goldstein et al. 2018, 2019; Pierel & Rodney 2019; Suyu et al. 2020; Huber et al. 2022), combined with measurements from distance ladders (e.g., Freedman et al. 2019, 2020; Riess et al. 2019, 2022) and lensed quasars (e.g., Suyu et al. 2010, 2013; Treu & Marshall 2016; Bonvin et al. 2017; Birrer et al. 2020; Millon et al. 2020; Wong et al. 2020), can be an important test of the tension between H_0 measured locally and the value inferred from the cosmic microwave background (CMB; Planck Collaboration et al. 2020).

The introduction of neural networks to identify gravitational lens candidates in imaging surveys has been transformational (e.g., Jacobs et al. 2017; Metcalf et al. 2019; Jacobs et al. 2019a, 2019b; Cañameras et al. 2020). In our recent work, we discovered over 1500 new strong lenses (Huang et al. 2020, 2021) in the Dark Energy Spectroscopic Instrument (DESI) Legacy Imaging Surveys (Dey et al. 2019) by using residual neural networks. This trend will accelerate in the era of the Vera C. Rubin Observatory Legacy Survey of Space and Time (LSST), Euclid, and the Nancy Grace Roman Space Telescope, when $\mathcal{O}(10^5)$ strong-lensing systems are projected to be found in the next decade (Collett 2015). As the lens search efficiency has dramatically improved, there has been significant development on strong-lens modeling as well. For example, the widely used lenstronomy (Birrer & Amara 2018,

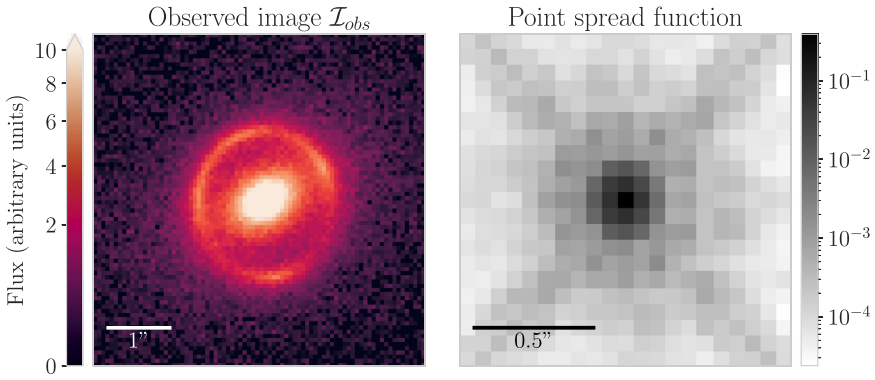


Figure 1. A lensing system simulated using `lenstronomy` is shown on the left. The parameters for this system are taken directly from the `lenstronomy` starting guide Jupyter Notebook (Birrer 2021). Although `lenstronomy` uses a Gaussian PSF for this system, we use a more realistic PSF calculated for the HST WFC3 F140W band using `TinyTim` (Krist et al. 2011) with an original pixel scale of $0\farcs13$. We then subsample this PSF to a scale of $0\farcs065$, as shown on the right. This system will be used to show the step-by-step workflow of our modeling pipeline. It will henceforth be called the reference system.

Birrer et al. 2021) is a fully fledged (including, e.g., a suite of plotting routines) and highly versatile lens modeling package. However, the computational cost for lens modeling remains high. For instance, Rojas et al. (2021) modeled 41 systems with a single deflector in *gri* bands from the Dark Energy Survey using a pipeline based on `lenstronomy`. They used Sérsic profiles for lens and source light, and for the lensing potential, they used a singular isothermal ellipsoid model and external shear. They reported that modeling a single system took 4.3 hr on average. In addition, `lenstronomy` uses particle swarm optimization (PSO). PSO is an easily parallelizable heuristic algorithm for nonconvex optimization. However, it does not have any guarantee of convergence, especially in high dimensions (see Section 2.3). Furthermore, `emcee` (Foreman-Mackey et al. 2013), a popular Markov Chain Monte Carlo (MCMC) algorithm in astrophysics, including being used in `lenstronomy`, relies on sampling techniques that also show undesirable behavior in high-dimensional parameter spaces (see Section 2.5.2). For lens modeling using high-resolution images, often more complex lens and source models will be used, and this can significantly increase the dimensionality of the parameter space. For strong gravitational lenses to realize their full potential as an effective probe for cosmology, it is crucial that we address these issues and make the process of modeling lensing systems robust, considerably faster, and scalable to high-dimensional parameter spaces. In this paper, we present a Bayesian lens modeling framework that fulfills these requirements. We describe our gradient-informed, GPU-accelerated (`GIGA-Lens`) framework in Section 2. In Section 3, we demonstrate the performance of our pipeline on a large set of simulated systems. We discuss our results and conclude in Section 4.

2. Lens Modeling

In this section, we introduce the `GIGA-Lens`[14] modeling framework. In the initial stages of the development of our pipeline, `lenstronomy` served as a helpful guide, specifically, in its approach of using optimization to find a region of high posterior density from which the MC sampler can be initialized. We believe that our framework represents a significant improvement on the `lenstronomy` modeling pipeline in terms of speed, optimization, and sampling. Our entire framework is implementated in both TensorFlow (Abadi et al. 2015) and JAX (Bradbury et al. 2018). Complete integration with either of these libraries confers significant advantages. It enables seamless execution of our code on graphics processing units (GPUs), which can achieve much faster gravitational lens simulation and modeling. Even modest, freely available GPUs are capable of performing basic linear algebra operations (which are at the core of lens modeling codes) one to two orders of magnitude faster than a typical CPU. In addition, our tight integration with TensorFlow allows us to use the TensorFlow Probability (Dillon et al. 2017) library for probabilistic modeling (which also provides support for JAX). Conveniently, this library has already implemented advanced statistical methods such as variational inference (VI) and adaptation algorithms (for step size and trajectory length; Hoffman et al. 2014, 2021) in Hamiltonian Monte Carlo (HMC). These features play a central role in our pipeline. In Section 2.1, we describe our lens model, both its physical and probabilistic aspects. Obtaining the gradient for the posterior with automatic differentiation is presented in Section 2.2.[15] Next, we detail the three main steps in our modeling pipeline. In Section 2.3, we find the global optimal values for the lensing parameters using multistart gradient descent. In Section 2.4, we estimate the covariance matrix around the global optimum using VI. This allows us to sample the posterior distribution efficiently with HMC in Section 2.5.

2.1. Model Specification

2.1.1. Physical Model

Given a lens model fully described by a set of parameters Θ, the predicted counts per second at arbitrary positions on the image plane, $\mathcal{I}_{\rm model}(x, y; \Theta)$, can be determined by evaluating the deflection angle at (x, y), using this deflection to ray-trace (e.g., Narayan & Bartelmann 1997) onto the source plane, setting the surface brightness of the lensed source at (x, y) to the corresponding surface brightness on the source plane (via surface brightness conservation), then adding the lens light, and finally convolving with the point-spread function (PSF).[16] To demonstrate how our modeling pipeline works, we simulate a

[14] Our framework is publicly available on GitHub under an open-source MIT license: https://github.com/giga-lens/gigalens.

[15] To our knowledge, Chianese et al. (2020) was the first to apply automatic differentiation in the context of strong lensing.

[16] The sky brightness, $I_{\rm sky}$, can also be included in the model parameters Θ. In this work, we assume that the sky brightness has been subtracted.

reference system (Figure 1) using `lenstronomy`. The lens model used for this simulation consists of an elliptical power-law[17] (EPL) mass model for the lens (Tessore & Metcalf 2015) and external shear. The EPL model is characterized by the surface mass density in units of the critical density, or convergence,

$$\kappa(x_{\rm lens}, y_{\rm lens}) = \frac{3-\gamma_{\rm epl}}{2}\left(\frac{\theta_{\rm E}}{\sqrt{q x_{\rm lens}^2 + y_{\rm lens}^2/q}}\right)^{\gamma_{\rm epl}-1}, \quad (1)$$

where $\theta_{\rm E}$ is the Einstein radius; $\gamma_{\rm epl}$ is the mass profile slope; the coordinates $x_{\rm lens}$, $y_{\rm lens}$ are aligned with the major and minor axes of the lens; and q is the axial ratio. The transformation between the image coordinates (x, y) and the lens-centric coordinates $(x_{\rm lens}, y_{\rm lens})$ is

$$\begin{bmatrix} x_{\rm lens} \\ y_{\rm lens} \end{bmatrix} = \begin{bmatrix} \cos\phi & \sin\phi \\ -\sin\phi & \cos\phi \end{bmatrix}\begin{bmatrix} x - x_{\rm epl} \\ y - y_{\rm epl} \end{bmatrix}, \quad (2)$$

where ϕ is the position angle and $x_{\rm epl}$ and $y_{\rm epl}$ are the lens center. The deflection angle for this model can be written in terms of the Gaussian hypergeometric function, which can be calculated iteratively, converging to a high degree of accuracy in a small number of iterations. The truncation error reaches $\lesssim 10^{-16}$ within 35 iterations for $q \gtrsim 0.5$ (Tessore & Metcalf 2015).

To include the effects of the local environment, we include an external shear with a deflection angle

$$\alpha_{\rm ext}(x, y) = (\gamma_{\rm ext,1}x + \gamma_{\rm ext,2}y, \gamma_{\rm ext,2}x - \gamma_{\rm ext,1}y). \quad (3)$$

Finally, for our simulations, we model our lens and source light with an elliptical Sérsic profile (Sérsic 1963):

$$I(x_{\rm light}, y_{\rm light}) = I_0 \exp\left(-b_n\left(\left(\frac{\sqrt{q x_{\rm light}^2 + y_{\rm light}^2/q}}{R_{\rm eff}}\right)^{1/n} - 1\right)\right), \quad (4)$$

where $b_n = 1.9992n - 0.3271$, $R_{\rm eff}$ is the effective radius (half-light radius), n is the Sérsic index, and $x_{\rm light}$ and $y_{\rm light}$ are aligned with the major and minor axes of the light profile. The transformation between image coordinates (x, y) and light coordinates $(x_{\rm light}, y_{\rm light})$ is identical in form to Equation (2). We will use (x_l, y_l) to denote the lens light center (in this work, the lens light center is not fixed to the lens mass center) and (x_s, y_s) to denote the source light center. Finally, in lens modeling, eccentricities are often used through the reparameterization,

$$(\epsilon_1, \epsilon_2) = \frac{1-q}{1+q}(\cos(2\phi), \sin(2\phi)), \quad (5)$$

where ϕ is the position angle. We will use $(\epsilon_{\rm epl,1}, \epsilon_{\rm epl,2})$ to denote the lens mass eccentricities, $(\epsilon_{l,1}, \epsilon_{l,2})$ for the lens light eccentricities, and $(\epsilon_{s,1}, \epsilon_{s,2})$ for the source light eccentricities. Similarly, R_l and R_s will denote the lens and source light effective radius.

[17] We note that the EPL model is equivalent to the power-law elliptical mass density profile (PEMD; Barkana 1998).

In practice, $\mathcal{I}_{\rm model}(x, y; \Theta)$ is vectorized to be evaluated on a grid of pixels simultaneously. This grid is supersampled by some integer factor $k_{\rm super}$ (in this work, $k_{\rm super} = 2$) from the image coordinates, so there is an additional step in evaluating $\mathcal{I}_{\rm model}$ that consists of downsampling (by averaging) the predicted image by $k_{\rm super}$ back to the image coordinate grid.

2.1.2. Probabilistic Model

Our probabilistic model comprises a likelihood function $\mathcal{L}(\Theta; \mathcal{I}_{\rm obs}) \equiv p(\mathcal{I}_{\rm obs}|\Theta)$ and a prior $p(\Theta)$, where $\Theta \in \mathbb{R}^d$ are the lensing system parameters and $\mathcal{I}_{\rm obs}$ is the observed image (in units of counts/sec). The likelihood function requires a pre-processed observed image $\mathcal{I}_{\rm obs}$, as well as specification of the background (sky) Gaussian noise $\sigma_{\rm bkg}$ and exposure time $t_{\rm exp}$. For this work, we define

$$\log\mathcal{L}(\Theta; \mathcal{I}_{\rm obs}) = -\frac{1}{2}\sum_{x,y} \times\left[\frac{(\mathcal{I}_{\rm obs}(x, y) - \mathcal{I}_{\rm model}(x, y; \Theta))^2}{\sigma_{\rm tot}^2(x, y; \Theta)} + \log(2\pi\sigma_{\rm tot}^2(x, y; \Theta))\right]$$
$$\sigma_{\rm tot}^2(x, y; \Theta) = \sigma_{\rm bkg}^2 + \frac{\mathcal{I}_{\rm model}(x, y; \Theta)}{\mathcal{G}\cdot t_{\rm exp}}, \quad (6)$$

where $\mathcal{I}_{\rm model}(x, y; \Theta)$ is our forward model for the observed image, $\mathcal{G}$ is the gain (in e^- count^{-1}), and the sum is over all pixels (x, y) in the observed data. The second term in the variance map arises from Poisson shot noise: since the expected value of the electron counts is $t_{\rm exp}\mathcal{I}_{\rm model}\mathcal{G}$, the Poisson variance is therefore also $t_{\rm exp}\mathcal{I}_{\rm model}\mathcal{G}$, and so the Poisson variance of the model image is $\frac{t_{\rm exp}\mathcal{I}_{\rm model}\mathcal{G}}{\mathcal{G}^2\cdot t_{\rm exp}^2} = \frac{\mathcal{I}_{\rm model}}{\mathcal{G}\cdot t_{\rm exp}}$. The form of this likelihood assumes independent per-pixel noise. However, the variance map $\sigma_{\rm tot}^2$ can be specified to account for correlated pixels and to incorporate prior information about the noise at each pixel. For example, an alternative form for $\sigma_{\rm tot}^2$, defined in terms of the observed image, is sometimes used, $\sigma_{\rm tot}^2(x, y) = \sigma_{\rm bkg}^2 + \frac{\mathcal{I}_{\rm obs}(x,y)}{\mathcal{G}\cdot t_{\rm exp}}$, so that the total variance is independent of Θ. This is done for computational efficiency. For example, it allows for linear light profile parameters to be solved for in closed form. We note, however, that this definition for the variance may induce biases at low signal-to-noise ratios (Horne 1986).[18] Therefore, we opt to use the more rigorous definition. In our pipeline, this incurs little additional computational cost. We also point out that the first term in the log-likelihood corresponds to

$$\chi^2 = \sum_{x,y}\frac{(\mathcal{I}_{\rm obs}(x, y) - \mathcal{I}_{\rm model}(x, y; \Theta))^2}{\sigma_{\rm tot}^2(x, y; \Theta)}, \quad (7)$$

and the second term is the normalization factor.

The prior for the parameters is typically defined as a product of independent distributions, each of which can be tuned either to reflect prior knowledge about lensing parameters or to match a

[18] Another issue with using $\mathcal{I}_{\rm obs}$ to define the noise map $\sigma_{\rm tot}^2$ is that it may have negative pixel values due to Gaussian noise. In `lenstronomy`, the negative pixels are set to zero, which is not the most rigorous way to estimate the Poisson noise (however, the final likelihood computation in lenstronomy still uses $\mathcal{I}_{\rm model}$ to define the noise map). Despite the slight difference in these two approaches, for our simulated systems (Section 3), the results using both are virtually identical.

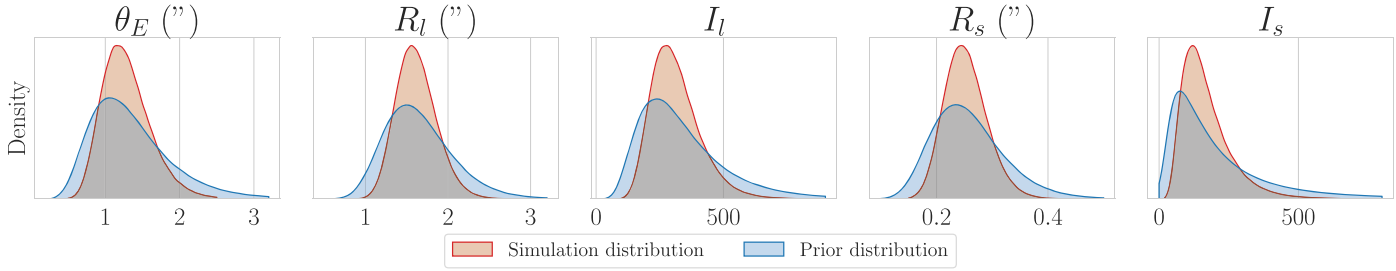


Figure 2. Probability density for five components of the simulation and prior distribution that are lognormal.

physical understanding of the particular system that is being modeled (e.g., the Einstein radius can typically be estimated to within 20% from visual inspection). Given our physical understanding of the model, our prior ought to vanish for certain regions of parameter space, such as $\theta_E < 0$. In our modeling, we make use of mappings that naturally enforce these constraints. We describe these mappings in Section 2.2.

In this work, we use a "simulation distribution" to generate a full set of 22 parameters for 100 lensing systems, and the prior distribution used to model these lenses (see Section 3) is a broadened version of this simulation distribution (note that some rows contain two parameters):

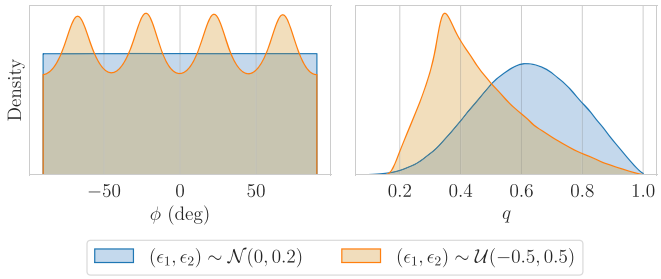


Figure 3. The corresponding distributions for ϕ, q when the ellipticities are normally distributed (blue) and uniformly distributed (orange).

$$
\begin{aligned}
\text{Lens mass: } &\begin{cases} \theta_E & \sim\exp(\mathcal{N}(\ln 1.25, 0.25/0.4)) & \text{[Einstein radius(}''\text{)]} \\ \gamma_{\rm epl} & \sim\mathcal{TN}(2, 0.25/0.5; 1, 3) & \text{[Mass slope]} \\ \epsilon_{epl,1},\ \epsilon_{epl,2} & \sim\mathcal{N}(0, 0.1/0.2) & \text{[Lens mass eccentricities]} \\ x_{\rm epl},\ y_{\rm epl} & \sim\mathcal{N}(0, 0.03/0.06) & \text{[Lens mass center(}''\text{)]} \\ \gamma_{\rm ext,1},\ \gamma_{\rm ext,2} & \sim\mathcal{N}(0, 0.05/0.1) & \text{[External shear components]} \end{cases} \\
\text{Lens light: } &\begin{cases} R_l & \sim\exp(\mathcal{N}(\ln 1.6, 0.15/0.25)) & \text{[Lens Sérsic radius(}''\text{)]} \\ n_l & \sim\mathcal{U}(2/0.5, 6/8) & \text{[Lens Sérsic index]} \\ \epsilon_{l,1},\ \epsilon_{l,2} & \sim\mathcal{TN}(0, 0.05/0.1; -0.15, 0.15) & \text{[Lens light eccentricities]} \\ x_l,\ y_l & \sim\mathcal{N}(0, 0.01/0.02) & \text{[Lens light center(}''\text{)]} \\ I_l & \sim\exp(\mathcal{N}(\ln 300, 0.3/0.5)) & \text{[Lens half light intensity]} \end{cases} \\
\text{Source light: } &\begin{cases} R_s & \sim\exp(\mathcal{N}(\ln 0.25, 0.15/0.25)) & \text{[Source Sérsic radius(}''\text{)]} \\ n_s & \sim\mathcal{U}(0.5, 4/8) & \text{[Source Sérsic index]} \\ \epsilon_{s,1},\ \epsilon_{s,2} & \sim\mathcal{TN}(0, 0.15/0.3; -0.5, 0.5) & \text{[Source light eccentricities]}, \\ x_s,\ y_s & \sim\mathcal{N}(0, 0.25/0.5) & \text{[Source light center(}''\text{)]} \\ I_s & \sim\exp(\mathcal{N}(\ln 150, 0.5/0.9)) & \text{[Source half light intensity]} \end{cases}
\end{aligned}
\tag{8}
$$

where $\mathcal{U}(a, b)$ is a uniform distribution with support $[a, b]$, $\mathcal{N}(\mu, \sigma)$ is Gaussian with mean μ and standard deviation σ, and $\mathcal{TN}(\mu, \sigma; x_{\rm low}, x_{\rm high})$ is a truncated Gaussian with support $[x_{\rm low}, x_{\rm high}]$. For the distribution parameters, we use the notation a/b to indicate that the simulation distribution uses the parameter a while the prior uses b. For instance, when generating our data set, we sample $x_{\rm epl}$ from $\mathcal{N}(0, 0.03)$, while during modeling, our prior for $x_{\rm epl}$ is $\mathcal{N}(0, 0.06)$.

Note that $\exp(\mathcal{N}(\ldots))$ is a lognormal distribution, used for five parameters. Since these distributions are not as intuitive, we show them in Figure 2. The simulation distribution for the light amplitudes I_l and I_s has been chosen in such a way that the typical signal-to-noise ratio of the arcs is 100 (with a range between 30 and 200) and is comparable to the amplitudes used in the `lenstronomy` starting guide Jupyter Notebook (Birrer 2021).

Although the distributions in Equation (8) can be specified in any format, in this work we make a number of deliberate choices. Most importantly, for parameters such as ellipticities (e.g., $\epsilon_{\rm epl,1}$, $\epsilon_{\rm epl,2}$) and centers (e.g., $x_{\rm epl}$, $y_{\rm epl}$), we use normal distributions to reflect their rotational symmetry. For instance, in Figure 3 we show how a normal distribution for the ellipticities results in a uniform distribution for the position

angle ϕ, whereas independent uniform distributions for the ellipticities break rotational symmetry, resulting in a nonuniform distribution for ϕ and a highly non-Gaussian distribution for q.

2.2. Obtaining Gradient Information

For the models discussed in Section 2.1.1, $\mathcal{I}_{\rm model}(x, y; \Theta)$ is a differentiable mapping with respect to the model parameters, Θ. Therefore, so is the posterior density (Equation (6)) within the support of the prior.[19] This is essential. As we will show in Sections 2.3 and 2.5, gradient information about the posterior is necessary to successfully achieve fast convergence for complex lens models. Although the posterior is theoretically differentiable, to obtain a gradient with numerical differentiation (i.e., finite difference) would be prohibitively expensive. Therefore, we exploit the fact that the mapping $\mathcal{I}_{\rm model}$ is composed of a sequence of differentiable operations (e.g., convolution) and use automatic differentiation (AD; Wengert 1964). Crucially with AD, which is implemented in both TensorFlow and JAX, the additional computational cost of evaluating the gradient is independent of the number of parameters (Baydin et al. 2018). The full gradient $\frac{\partial \mathcal{I}_{\rm model}}{\partial \Theta_i}$ that we are interested in can be calculated in approximately the same amount of time it takes to calculate $\mathcal{I}_{\rm model}$ itself.

Although we can calculate the gradient efficiently with AD, there are regions of parameter space where this gradient is undefined as a result of having hard boundaries for the parameters. This can have a number of undesirable effects for modeling. For example, any algorithm with an iterative update that has a finite step size for the parameters may result in certain solutions being updated to positions outside the support of the prior. Therefore, we instead use smooth, invertible functions (termed "bijectors"), g, to map from an unconstrained space to the support of the prior, supp(p). For instance, the Einstein radius has a semi-infinite support $\mathbb{R}_{>0}$, so we use the exponential map

$$g_{\theta_{\rm E}}\colon z \mapsto e^z. \tag{9}$$

For other parameters, such as the mass profile slope γ, that may have a finite support $(a, b) = (1, 3)$, a convenient bijector to use is the sigmoid mapping:

$$g_{\gamma}\colon z \mapsto a + \frac{b-a}{1+e^{-z}}. \tag{10}$$

The joint bijector $g\colon \mathbb{R}^d \to \text{supp}(p)$ is typically constructed component-wise from the bijectors for each parameter.[20] That is, $\Theta = g(\tilde{\Theta})$, using the notation $\tilde{\cdot}$ to henceforth denote quantities in the unconstrained space. This allows us to optimize or sample over the unconstrained space $\mathbb{R}^d$, rather than manually enforcing constraints on the parameters. Since the volume element in the unconstrained space is different from the constrained parameter space, the prior needs to be modified,

$$\log \tilde{p}(\tilde{\Theta}) = \log p(g(\tilde{\Theta})) + \log|J|, \tag{11}$$

where J is the Jacobian of g evaluated at $\tilde{\Theta}$. The likelihood remains unchanged,

$$\log \tilde{p}(\mathcal{I}_{\rm obs}|\tilde{\Theta}) = \log p(\mathcal{I}_{\rm obs}|g(\tilde{\Theta})). \tag{12}$$

Thus, for the posterior,

$$\log \tilde{p}(\tilde{\Theta}|\mathcal{I}_{\rm obs}) = \log p(\Theta|\mathcal{I}_{\rm obs}) + \log|J|. \tag{13}$$

The constrained parameters that correspond to physical quantities in the lens model (e.g., $\theta_{\rm E}$) are hereafter called physical parameters. With the posterior distribution thus reparameterized, it is straightforward to compute its derivatives using AD.

2.3. Maximum a Posteriori Estimate

When sampling from a high-dimensional posterior using Monte Carlo (MC) algorithms, since all but a small fraction of the parameter space has a vanishing posterior density, arbitrarily chosen initializations for these samplers will take a long time to converge to the posterior distribution. Therefore, it is necessary to identify a region of high posterior density from which MC samplers can be initialized. Typically, the maximum a posteriori (MAP) estimate serves this purpose well:

$$\begin{aligned}\tilde{\Theta}^*_{\rm MAP} &= \underset{\tilde{\Theta}\in\mathbb{R}^d}{\text{argmax}}\ \log \tilde{p}(\tilde{\Theta}|\mathcal{I}_{\rm obs}) \\ &= \underset{\tilde{\Theta}\in\mathbb{R}^d}{\text{argmin}}\ (-\log \tilde{p}(\mathcal{I}_{\rm obs}|\tilde{\Theta}) - \log \tilde{p}(\tilde{\Theta})).\end{aligned} \tag{14}$$

Here and below, the notation $\cdot^*$ will denote a local optimum, and the notation $\cdot^*_{\rm MAP}$ will indicate that this local optimum is also globally optimal. Finding $\tilde{\Theta}^*_{\rm MAP}$ is an optimization task, with the objective function f being the negative (unnormalized) log posterior density:

$$f(\tilde{\Theta}) = -(\log \tilde{p}(\mathcal{I}_{\rm obs}|\tilde{\Theta}) + \log \tilde{p}(\tilde{\Theta})). \tag{15}$$

There are different approaches to global optimization for nonconvex, multimodal functions such as Equation (15). For example, `lenstronomy` uses PSO. Although this method is attractive because it requires only the evaluation of the objective function f (and not its gradient), a major weakness is that there are "little to no guarantees" (Sengupta et al. 2018) for finding the global, or even local, minimum. We take the approach of gradient descent, which has a number of advantages over heuristic optimization techniques such as PSO. Most importantly, gradient descent can at least guarantee convergence toward a local minimum. Furthermore, when close enough to a local minimum, $\tilde{\Theta}^*$, gradient descent approximately achieves a geometric convergence rate. That is, $f(\tilde{\Theta}^{(k)}) - f(\tilde{\Theta}^*)$ is upper bounded by $\mathcal{O}(\beta^k)$, where $\beta < 1$ and $\tilde{\Theta}^{(k)}$ is the candidate solution on the kth iteration, called the kth iterate (Nesterov 2004). To achieve the global optimum, we disperse a large number of samples $n_{\rm MAP}$ throughout a wide region of the parameter space and carry out gradient descent on each of these samples (Martí 2003; György & Kocsis 2011). Since the iterates of these samples will quickly and reliably converge toward local

[19] Although the gradient is continuous everywhere, it may vanish in certain regions of the parameter space owing to extreme misalignment of the lens and source. We avoid this by setting our prior in such a way that $\sqrt{(x_s - x_{\rm epl})^2 + (y_s - y_{\rm epl})^2} \sim \theta_{\rm E}$.

[20] Although g can be specified in any desired form, we find that simply constructing it component-wise from the standard bijectors Equations (9) and (10) is sufficient. This component-wise construction is also inexpensive to evaluate, since each of the standard bijectors is elementary.

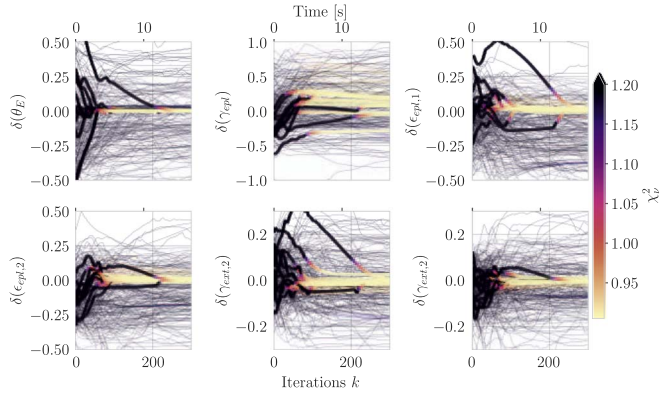


Figure 4. The error trajectories of the lensing parameters (i.e., the difference between the kth iterate and the ground truth) for the reference system (see Figure 1) over the course of gradient descent. Trajectories that end in $\chi^2_\nu \leqslant 1.01$ (where $\chi^2_\nu = \chi^2/\mathrm{DOF}$) are shown with thicker lines, all of which converge to the global minimum (see Figure 5). We note that a wide range of samples converge to the correct solution (see text).

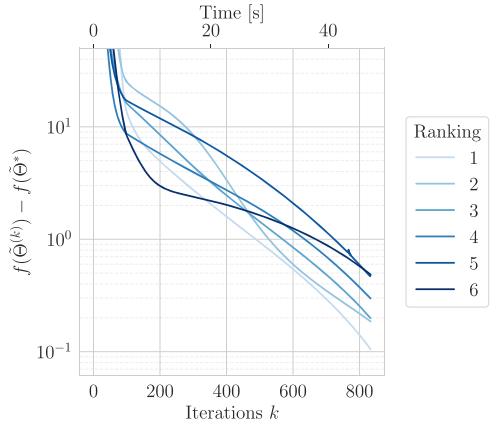


Figure 5. Six of the best-performing loss trajectories, ranked, over 800 iterations of gradient descent. We have run 800 iterations (after which the total χ^2 difference is $\lesssim 0.5$) solely for demonstration purposes. In our pipeline, however, we terminate the gradient descent at the 300th iteration because at that stage the best-performing trajectory always ends up at the global minimum. In addition, with the y-axis in log scale, note the approximate geometric convergence of each solution starting around the 200th iteration, after the initial steep descent.

minima, we only need to ensure that n_{MAP} is large enough such that at least one of the samples reaches sufficient proximity of the global minimum. The loss function $f(\tilde{\Theta})$ is multimodal in the sense that there exist multiple local minima,[21] but we find that a moderate number of samples $n_{\mathrm{MAP}} = 300$ and $K_{\mathrm{MAP}} = 300$ iterations of gradient descent are sufficient for consistent identification of the global optimum. After K_{MAP} iterations, we take the best of the n_{MAP} samples to be the MAP estimate $\tilde{\Theta}^*_{\mathrm{MAP}}$. As we show in Figure 4, many samples, even some that start far away from the global minimum, converge to the neighborhood of the correct solution. If the gradient descent were run for more iterations, each of them would reach the same, globally optimal solution (see Figure 5). Typically $\sim 5\%$ (and at least 1%, for systems with a low signal-to-noise ratio) of the samples converge to the global optimum, with the other samples eventually converging to local optima. This demonstrates the robustness of the multistart gradient descent method.

Each of the n_{MAP} samples is initialized by sampling from the prior: $\tilde{\Theta}_i^{(1)} \sim \tilde{p}(\tilde{\Theta})$ for $i = 1,\ldots,n_{\mathrm{MAP}}$. Finally, similar to the training of neural networks, we use the Adam optimizer (Kingma & Ba 2017) with an initial large learning rate[22] $\alpha = 10^{-2}$ to accelerate "learning" and escape spurious local minima, and we decay it to $\alpha = 10^{-3}$ over 300 iterations to help the optimization converge and avoid instabilities (You et al. 2019). This is reflected in the loss trajectories shown in Figure 5, which show a period of rapid improvement in the first $\sim$200 iterations toward the neighborhood that surrounds the global mode, followed by approximate geometric convergence.

As noted in Section 2.2, with AD, the gradient can be calculated at virtually no additional computational cost. Furthermore, after initializing n_{MAP} samples, the optimization can be done simultaneously by virtue of our code's parallelization. Combining these two performance enhancements, this step of our modeling pipeline is fast: to find the global optimum, it takes just 17 s (see Section 2.6, Table 1) to run 300 iterations of gradient descent with $n_{\mathrm{MAP}} = 300$ (Figure 4).

2.4. Variational Inference

After finding the MAP estimate, $\tilde{\Theta}^*_{\mathrm{MAP}}$, it is necessary to do an intermediate step of analysis before sampling via MC. Specifically, we estimate the posterior covariance matrix of the lens parameters Σ. This covariance estimate plays an important auxiliary role for MC sampling: it sets a scale for each of the parameters that is used to define a proposal distribution in the sampling step, as we will show in Section 2.5. We find this covariance estimate by using VI (Blei et al. 2017) to fit a multivariate normal $\mathcal{N}(\tilde{\mu}, \tilde{\Sigma})$ (called the "surrogate" posterior) with a probability density $\tilde{q}(\tilde{\Theta}; \tilde{\mu}, \tilde{\Sigma})$ to the true posterior.[23] This means minimizing the Kullback–Leibler (KL) divergence between the two distributions:

$$\tilde{\mu}^*_{\mathrm{VI}}, \tilde{\Sigma}^*_{\mathrm{VI}} = \underset{\tilde{\mu},\tilde{\Sigma}}{\mathrm{argmin}}\ \mathrm{KL}(\tilde{q}(\tilde{\Theta}; \tilde{\mu}, \tilde{\Sigma}) || \tilde{p}(\tilde{\Theta}|\mathcal{I}_{\mathrm{obs}})). \tag{16}$$

Since the posterior density $\tilde{p}(\tilde{\Theta}|\mathcal{I}_{\mathrm{obs}})$ is intractable, we decompose the KL:

$$\begin{aligned}
&\mathrm{KL}(\tilde{q}(\tilde{\Theta}; \tilde{\mu}, \tilde{\Sigma}) || \tilde{p}(\tilde{\Theta}|\mathcal{I}_{\mathrm{obs}})) \\
&= \mathbb{E}_{\tilde{\Theta}\sim\mathcal{N}(\tilde{\mu},\tilde{\Sigma})}\left[\log \tilde{q}(\tilde{\Theta}; \tilde{\mu}, \tilde{\Sigma}) - \log \frac{\tilde{p}(\mathcal{I}_{\mathrm{obs}}, \Theta)}{p(\mathcal{I}_{\mathrm{obs}})}\right] \\
&= \underbrace{\mathbb{E}_{\tilde{\Theta}}[\log \tilde{q}(\tilde{\Theta}; \tilde{\mu}, \tilde{\Sigma}) - \log \tilde{p}(\mathcal{I}_{\mathrm{obs}}, \tilde{\Theta})]}_{\text{ELBO loss}} + \underbrace{\mathbb{E}_{\tilde{\Theta}}[\log p(\mathcal{I}_{\mathrm{obs}})]}_{\text{Independent of } \tilde{\mu},\tilde{\Sigma}},
\end{aligned} \tag{17}$$

where $\mathbb{E}_{\tilde{\Theta}}$ denotes the expectation value with respect to the surrogate posterior $\tilde{q}$. Therefore, minimizing the KL divergence is equivalent to minimizing the evidence lower bound (ELBO). This is tractable since $\tilde{p}(\mathcal{I}_{\mathrm{obs}}, \tilde{\Theta})$ can be expressed as the product of the prior $\tilde{p}(\tilde{\Theta})$ and likelihood $\tilde{p}(\mathcal{I}_{\mathrm{obs}}|\tilde{\Theta})$, each of

[21] However, the posterior is *not* multimodal in the sense that each of these local minima has vanishing posterior density relative to the global mode. Although each of these local minima has a large effect on the performance of MAP, for sampling, they are irrelevant.

[22] The learning rate is used as a multiplier for the gradient when updating parameters in gradient descent, with the simplest implementation being $\tilde{\Theta}_i^{(k+1)} = \tilde{\Theta}_i^{(k)} - \alpha \nabla f(\tilde{\Theta}_i^{(k)})$. The Adam optimizer is slightly more complex and rescales the components of the gradient before applying the update. We refer readers to Kingma & Ba (2017) for a more detailed description of the Adam update rule.

[23] This roughly corresponds to calculating the posterior mode and evaluating the Hessian of the log posterior density around the mode.

Table 1
Summary of the Modeling Pipeline

Step	Output	Hyperparameters	Initialization	Execution Time
1. MAP (Section 2.3)	An estimate of the posterior mode $\tilde{\Theta}^*_{\rm MAP}$	$K_{\rm MAP}$: 300 $n_{\rm MAP}$: 300 α: $10^{-2} \xrightarrow{\text{lin, 300}} 10^{-3}$	$\tilde{\Theta}_i^{(1)} \sim \tilde{p}(\tilde{\Theta})$ $i = 1,\ldots,n_{\rm MAP}$	17 s
2. VI (Section 2.4)	An estimate of the posterior mean $\tilde{\mu}^*_{\rm VI}$ and covariance $\tilde{\Sigma}^*_{\rm VI}$	$K_{\rm VI}$: 1000 $n_{\rm VI}$: 500 α: $0 \xrightarrow{\text{quad, 500}} 10^{-3}$	$\tilde{\mu}^{(1)} = \tilde{\Theta}^*_{\rm MAP}$ $\tilde{\Sigma}^{(1)} = 10^{-6}\mathbb{I}$	52 s
3. HMC (Section 2.5)	Samples drawn from the posterior $p(\Theta\|\mathcal{I}_{\rm obs})$	$n_{\rm burn}$: 250 $n_{\rm sample}$: 750 $n_{\rm HMC}$: 50 ϵ: 0.3 L: 5 $\tilde{M}$: $(\tilde{\Sigma}^*_{\rm VI})^{-1}$	Initialize $n_{\rm HMC}$ walkers by sampling from the VI posterior	36 s
Total				**105 s**

Note. The hyperparameters for each step are defined in their respective subsections (first column). We adopt the notation $\alpha_1 \xrightarrow{s,\,k} \alpha_2$ to indicate a learning rate that changes from α_1 to α_2 over k iterations with a polynomial schedule s (in our case, linear or quadratic). The rightmost column indicates typical execution times for each modeling step on four A100 GPUs. On a single A100 GPU, the run time is approximately 3.5 times longer (see Figure 8), totaling ∼6 minutes.

which is readily available. The gradient of this loss is also expressible as an expectation (Ranganath et al. 2014):

$$\nabla_{\tilde{\mu},\tilde{\Sigma}}\mathrm{KL} = \nabla_{\tilde{\mu},\tilde{\Sigma}}\mathrm{ELBO} = \mathbb{E}_{\tilde{\Theta}}[(\log \tilde{q}(\tilde{\Theta};\,\tilde{\mu},\,\tilde{\Sigma}) - \log \tilde{p}(\mathcal{I}_{\rm obs},\,\tilde{\Theta}))\nabla_{\tilde{\mu},\tilde{\Sigma}} \log \tilde{q}(\tilde{\Theta};\,\tilde{\mu},\,\tilde{\Sigma})]. \quad (18)$$

In practice, at each iteration, we use AD to calculate $\nabla_{\tilde{\mu},\tilde{\Sigma}} \log \tilde{q}(\tilde{\Theta};\,\tilde{\mu},\,\tilde{\Sigma})$ and approximate the expectation in Equation (18) with a finite number of samples[24] $n_{\rm VI}$ drawn from $\mathcal{N}(\tilde{\mu},\,\tilde{\Sigma})$. This forms an estimator for the true gradient $\nabla_{\tilde{\mu},\tilde{\Sigma}}\mathrm{KL}$, which we use to do gradient descent with the Adam optimizer to minimize the ELBO in Equation (17). We note that the covariance matrix $\tilde{\Sigma}$ is constrained to be positive semidefinite. In keeping with our method of using unconstraining bijectors in Section 2.2, we use a Cholesky bijection mapping unconstrained real vectors to positive semidefinite matrices and optimize over this unconstrained space.[25] We initialize our VI with the MAP estimate $\tilde{\mu}^{(1)} = \tilde{\Theta}^*_{\rm MAP}$ and a diagonal covariance matrix $\tilde{\Sigma}^{(1)} = 10^{-6}\mathbb{I}$, based on the intuition that it will be easier for VI to approximate the true covariance starting from an underestimate. Based on a coarse optimization for the reference system, we run VI for $K_{\rm VI} = 1000$ iterations using $n_{\rm VI} = 500$, with the learning rate being increased quadratically from $\alpha = 0$ to $\alpha = 10^{-3}$ over 500 iterations.[26] This slow increase in learning rate is to avoid initial instabilities in the optimization that may result from the crude initial guess for the covariance $\tilde{\Sigma}^{(1)}$. In our TensorFlow implementation of the modeling pipeline, we use the TensorFlow Probability methods for VI, whereas in our JAX implementation we calculate Equation (18) directly.

The resulting best-fit distribution is only an approximation, since the true posterior is not necessarily Gaussian (see the Appendix). Even when the marginals of the posterior appear to be Gaussian, this does not necessarily imply that the full posterior is jointly Gaussian (see Figure 6 and Dutta & Genton 2014). This is consistent with the fact that the VI posterior does not always exactly agree with the true posterior (i.e., HMC samples). Nonetheless, we find that the VI covariance matrix is almost always a sufficiently good estimate of the true covariance.

2.5. Hamiltonian Monte Carlo

For the last step, sampling, we will use HMC (Duane et al. 1987; Neal 2011). HMC relies on gradient information about the posterior distribution and is known to have several advantages over gradient-free MCMC samplers (e.g., `emcee`). Conveniently, this gradient information is readily available via AD. Using HMC, we achieve highly efficient sampling. These results are shown in Section 2.5.1. Furthermore, we compare the performance of HMC and the widely used `emcee` sampler in Section 2.5.2 and show that in high-dimensional spaces HMC is strongly preferred.

[24] This way of doing VI is sometimes termed stochastic VI (Hoffman et al. 2013) because of the stochasticity induced by the finite number of samples $n_{\rm VI}$ used at each iteration.

[25] The bijector uses the fact that any covariance matrix can be written in terms of its Cholesky decomposition $\Sigma = LL^T$, where the Cholesky factor L is a lower triangular matrix with a nonnegative diagonal. The unconstrained space of real vectors $\mathbb{R}^{d(d+1)/2}$ is then mapped to a covariance matrix by first reshaping the vector into a lower triangular matrix, exponentiating the diagonal entries (which defines a valid Cholesky factor L), and then multiplying LL^T to find the corresponding covariance matrix.

[26] We start from $\alpha = 0$ to allow Adam to adjust its first- and second-order moment estimates (see Kingma & Ba 2017).

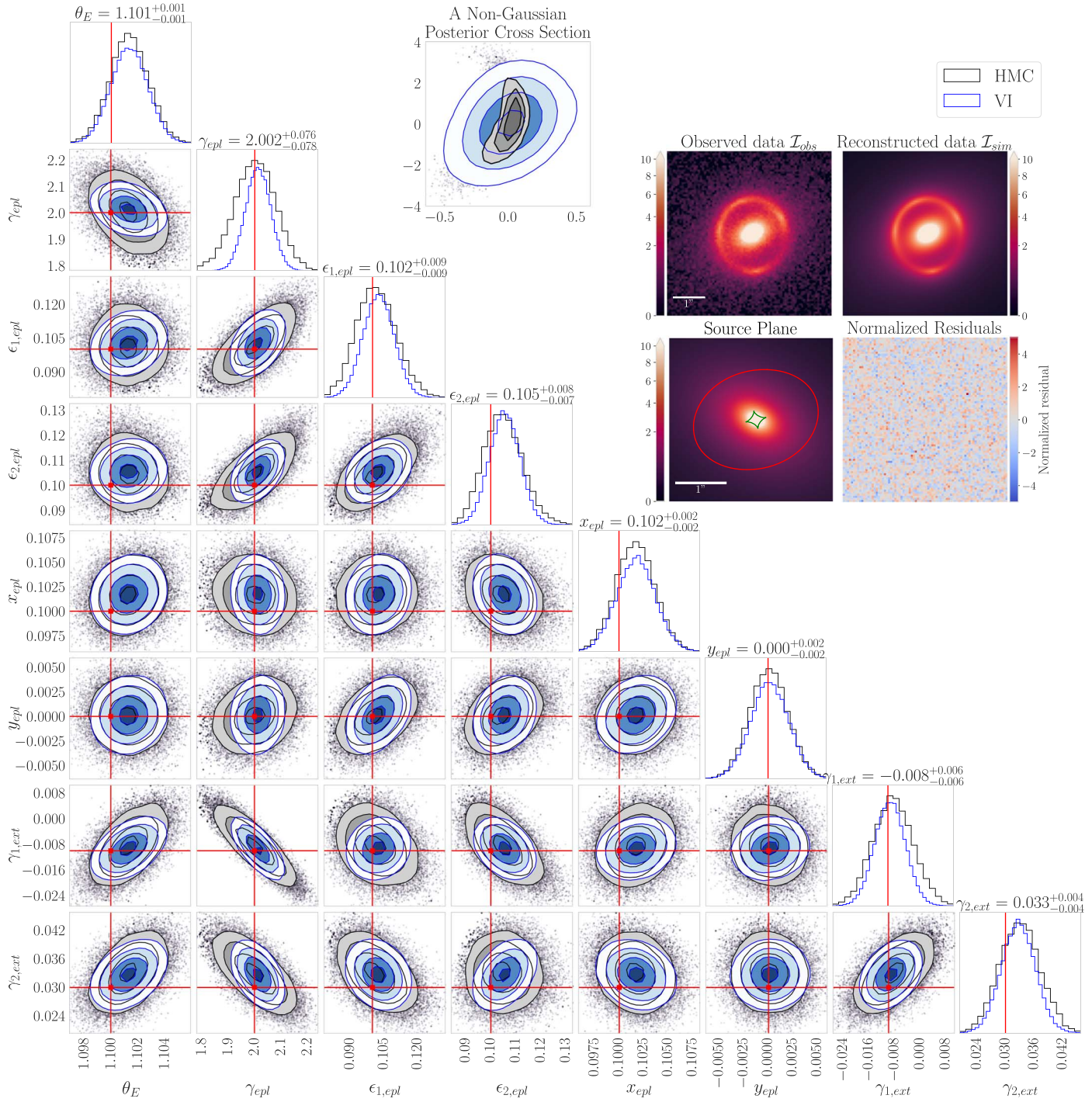


Figure 6. A corner plot of the posterior samples for the reference system (see Equation (8) for the definition of each parameter). We show only the lensing parameters, marginalizing out all light parameters (marginalized posteriors for all 22 parameters are shown in Figure 7). The posterior samples are first obtained in unconstrained space from both VI and HMC and then converted to physical parameters by applying the bijector g (see text). The samples and 0.5σ, 1σ, 1.5σ, and 2σ contours (corresponding to roughly 12%, 39%, 68%, and 86% of the probability mass) for both VI and HMC are shown in blue and gray, respectively, and the ground truth is in red. In the top right inset, we show the model-reconstructed image, residuals (normalized by the square root of the noise variance map), and the reconstructed source (together with the caustic shown in green and critical curve in red) using the Bayesian mean estimate. Despite the approximately Gaussian marginal distributions on the corner plot for the true posterior, we show a random cross section of the posterior that is banana shaped (top inset), demonstrating that the full posterior is not perfectly Gaussian (see text). Since the VI ansatz is a multivariate Gaussian, this is consistent with the fact that the marginals for the VI posterior do not entirely coincide with those of the true posterior.

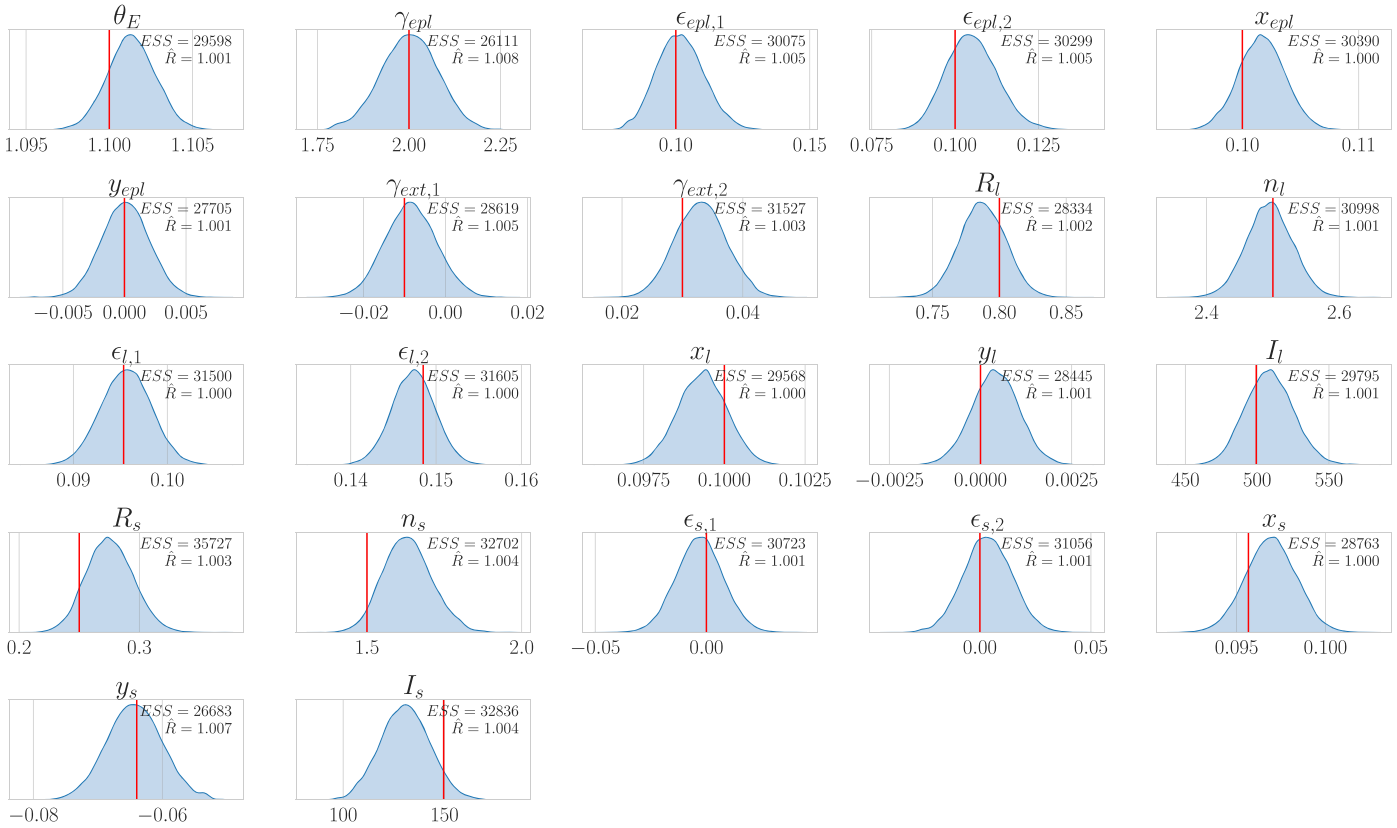


Figure 7. Marginalized posterior samples for all 22 parameters of the reference system. As in Figure 6, the ground truth (input values) is marked in red. We report the ESS and PSRF $\hat{R}$ for our posterior samples. Note that for each parameter ESS > 26,000 and $\hat{R} < 1.01$. This is achieved with just under 36 s of HMC sampling (Table 1).

2.5.1. Sampler Configuration

We sample from the posterior $\tilde{p}(\tilde{\Theta}|\mathcal{I}_{\rm obs})$ in the unconstrained parameter space with HMC and convert back to samples of the physical parameters $p(\Theta|\mathcal{I}_{\rm obs})$ using the bijector g. We initialize $n_{\rm HMC}$ chains[27] by sampling from the surrogate posterior calculated by VI (namely, $\mathcal{N}(\tilde{\mu}^*_{\rm VI}, \tilde{\Sigma}^*_{\rm VI})$) and, as is typical for MC samplers, run $n_{\rm burn}$ burn-in steps before sampling $n_{\rm sample}$ times.

Correctly configuring the HMC sampler is important to realizing its advantages. There are three hyperparameters in HMC: the step size ϵ, the number of leapfrog steps L, and the mass matrix $\tilde{M}$ that defines the momentum distribution. Much work has been done to adaptively set the first two hyperparameters ϵ and L. We use the methods introduced by Hoffman et al. (2014, 2021), as implemented by TensorFlow Probability, to automatically tune ("autotune") the hyperparameters during the first 80%[28] of burn-in phase. They are then fixed in the remaining burn-in steps, since their adaptation generally prevents chains from reaching the stationary distribution (hence the use of only the first 80% of burn-in steps for adaptation). Part of this tuning is adjusting the step size ϵ to achieve a target acceptance probability for each proposal: a step size that is too small will result in slower sampling, but a step size that is too large will result in too many rejected proposals. Optimal values for the target acceptance probability range from 0.6 to 0.8 (Betancourt 2018); we use 0.75. The remaining hyperparameter, the mass matrix $\tilde{M}$, defines the momentum distribution, and this provides a way to inform the HMC algorithm about the scales and correlations of the parameters. We can significantly improve the sampling efficiency by setting $\tilde{M}$ to be the inverse covariance matrix of the posterior (Neal 2011, pp.113–160)—this is called "preconditioned" HMC. This was the main purpose of Section 2.4: we set $\tilde{M} = (\tilde{\Sigma}^*_{\rm VI})^{-1}$, the inverse of the inferred covariance matrix from VI as defined by Equation (16). There have been proposals (e.g., Sountsov & Hoffman 2021) to adapt the mass matrix $\tilde{M}$ on the fly during the burn-in steps, which may render the VI step irrelevant. For now, these methods are not yet well tested (however, they may be incorporated in future work); hence, VI remains a necessary step of our modeling pipeline. Note that a secondary use of the VI step is that each of the $n_{\rm HMC}$ chains is initialized (see Table 1) by sampling from the VI posterior, $\mathcal{N}(\tilde{\mu}^*_{\rm VI}, \tilde{\Sigma}^*_{\rm VI})$.

We show in Figure 7 the posterior samples for our reference system generated using $n_{\rm HMC} = 50$ chains and $n_{\rm burn}$, $n_{\rm sample} = 250, 750$. As with the VI step, these hyperparameters were roughly tuned on the reference system. We report two metrics that are widely used in the statistics literature to measure the degree to which our sampler has converged. These are known as the effective sample size (ESS) and potential scale reduction factor (PSRF), $\hat{R}$ (Gelman & Rubin 1992). The former measures the effective number of independent samples we have drawn from the posterior by accounting for autocorrelation

[27] We use multiple chains, $n_{\rm HMC} = 50$, first because our framework naturally lends itself to parallelization, so it is more efficient to sample for 750 iterations using 50 chains rather than sampling for $750 \cdot 50$ iterations using one chain. Using multiple chains is also necessary to evaluate sampling diagnostics such as $\hat{R}$ (see the end of this section).

[28] As recommended by Tensorflow Probability (see https://www.tensorflow.org/probability/api_docs/python/tfp/mcmc/DualAveragingStepSizeAdaptation).

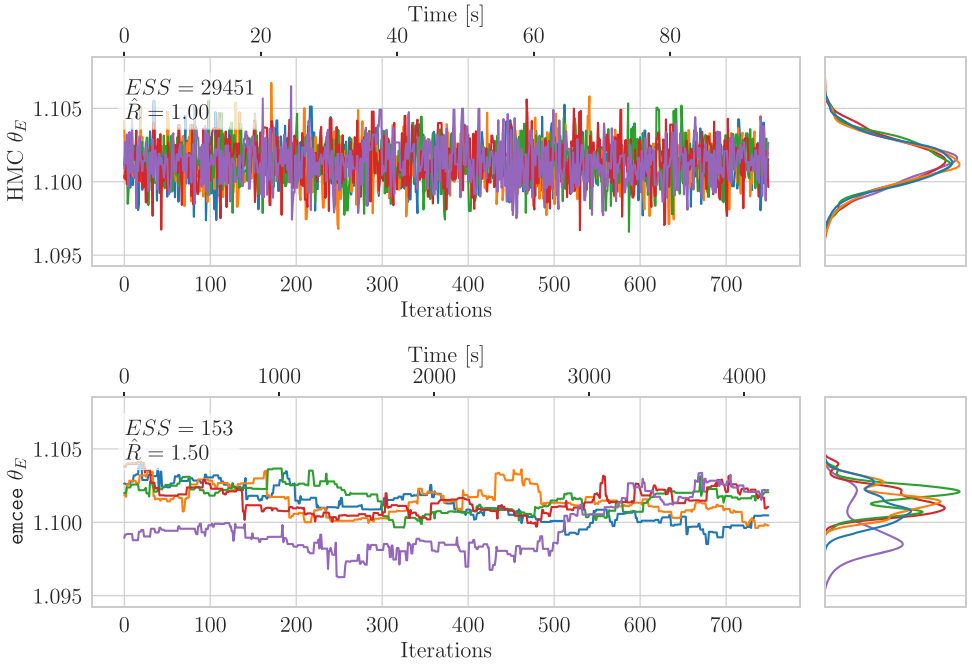


Figure 8. A comparison of HMC with `emcee` performance for the Einstein radius. We illustrate five randomly chosen chains (from 50 total chains) for both HMC and `emcee` over 750 sampling iterations. The times shown are for HMC run on a *single* A100 GPU and `emcee` run on a modern CPU. Note the poor interchain mixing in `emcee`, which leads to a high $\hat{R}$. In particular, for `emcee`, $\hat{R}$ is much higher than the recommended value of 1.1. This is representative behavior for all other physical parameters.

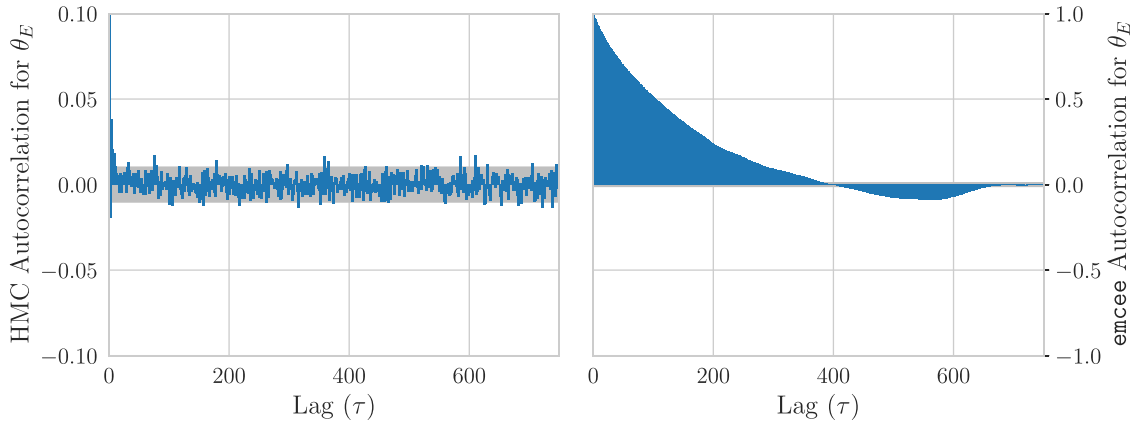


Figure 9. Autocorrelation of the Einstein radius, $\hat{\rho}(\tau)$, using HMC and `emcee` (see text). This is representative behavior for all other physical parameters. Note that the vertical scale for HMC is 1/10 that of `emcee`.

within each chain, and the latter is the ratio of the average within-chain variance to the variance of the pooled samples across all chains. A large ESS and an $\hat{R}$ that is close to 1 indicate that convergence has been achieved (in Gelman & Rubin 1992, it is suggested that an appropriate condition is $\hat{R} < 1.1$).

2.5.2. Comparison of HMC and `emcee`

Foreman-Mackey et al. (2013) implemented an affine-invariant ensemble sampler, `emcee`. It is a popular MCMC algorithm in astrophysics. This is the default sampler that `lenstronomy` uses. Here we compare the performance of HMC with `emcee`, which is gradient-free, by applying both to the reference system (Figure 1). To make the comparison as fair as possible, for `emcee` sampling we initialize the sampler with the `lenstronomy` recommended configuration, as detailed in Birrer (2021), and for our HMC sampling we initialize the sampler as detailed in Table 1. Furthermore, we run our pipeline on a single A100 GPU and `emcee` on a single CPU. `lenstronomy` uses uniform priors for each parameter, whereas we use the prior described in Equation (8). However, we have found that the difference in priors has virtually no effect on the sampling results. Finally, for both modeling pipelines, we use the supersampling factor $k_{\rm super} = 2$ and the PSF shown in Figure 1.

We take two axes of comparison between HMC and `emcee`. First, we observe that our sampling process is significantly more efficient than `emcee`, as evidenced by the rate at which HMC generates independent samples, ~ 40 ESS/iter ($\sim$300 ESS s^{-1}, on a single A100 GPU), whereas for `emcee` it is ~ 0.2 ESS/iter ($\sim$0.04 ESS s^{-1}) (see Figure 8). Second, we compare the convergence of the two samplers. We find that although both sampling methods agree in terms of their central values, they exhibit dramatically different convergence behavior. In Figure 8, we show that individual `emcee` chains tend to devolve to random

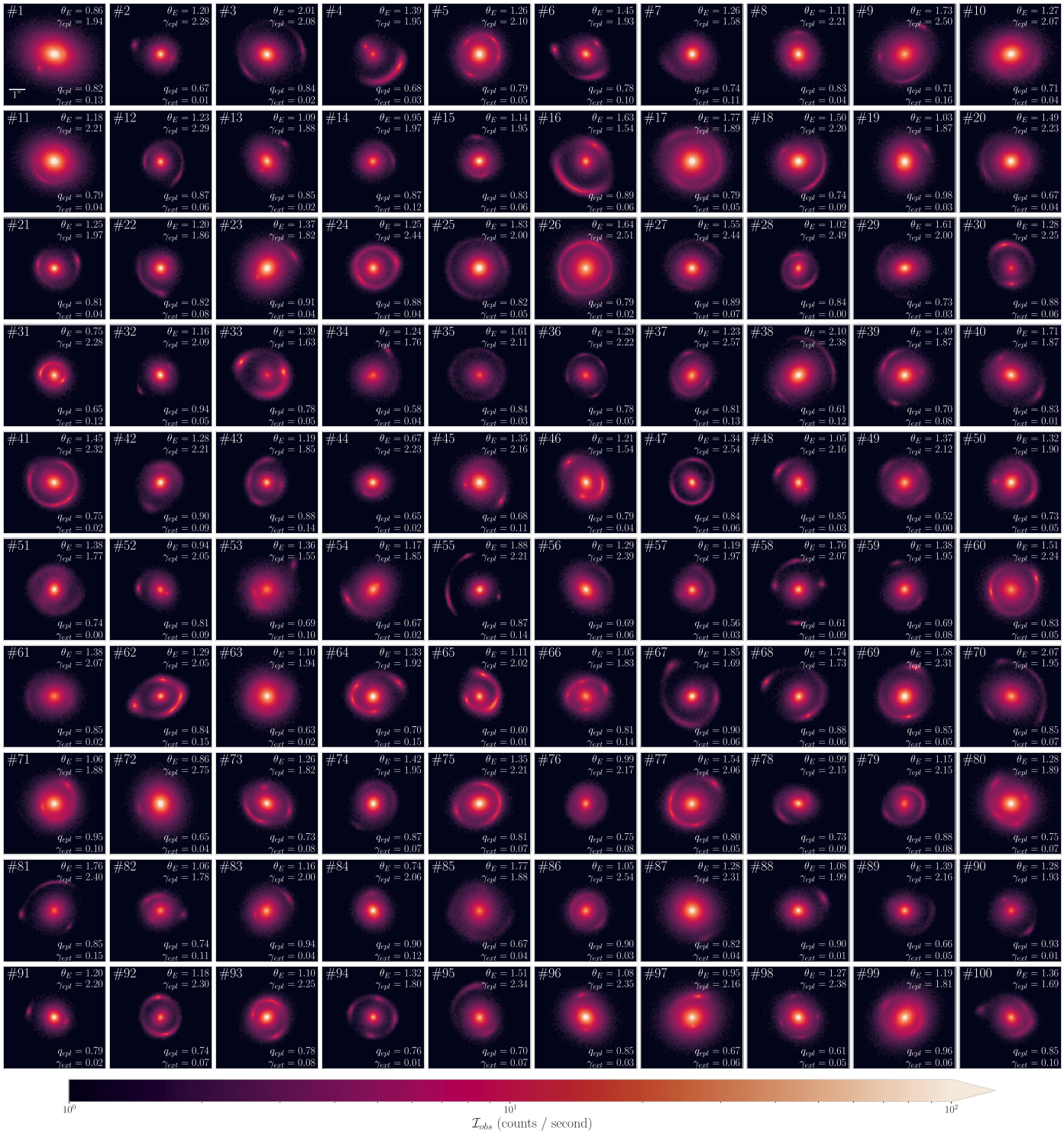


Figure 10. A sample of 100 lenses simulated using `lenstronomy`. We include the effects of Gaussian noise with standard deviation $\sigma_{\rm bkg} = 0.2$, Poisson shot noise with an exposure time $t_{\rm exp} = 100$ s with $\mathcal{G} = 1$ (assuming HST observations), and the PSF (see Figure 1). The pixel scale is $0\farcs065$, and the cutout size is $5\farcs2 \times 5\farcs2$ (80 pixels by 80 pixels).

walks. This random walk behavior manifests itself in three ways. First, `emcee` makes slow progress exploring the posterior, whereas HMC draws virtually independent samples each iteration, traversing the posterior very efficiently. Second, compared with HMC, we observe high interchain variance in `emcee`, evidenced qualitatively by the differing marginal distributions for each of the individual chains and quantitatively by the substantially higher $\hat{R}$ for `emcee` (Figure 8). Relatedly, in Figure 9 we find that the autocorrelation time for HMC is much lower than that of `emcee`: within just 10 iterations, the autocorrelation shrinks to negligible levels, compared to `emcee`, which has a characteristic autocorrelation lag of $\sim$300. The empirical autocorrelation at lag τ for a

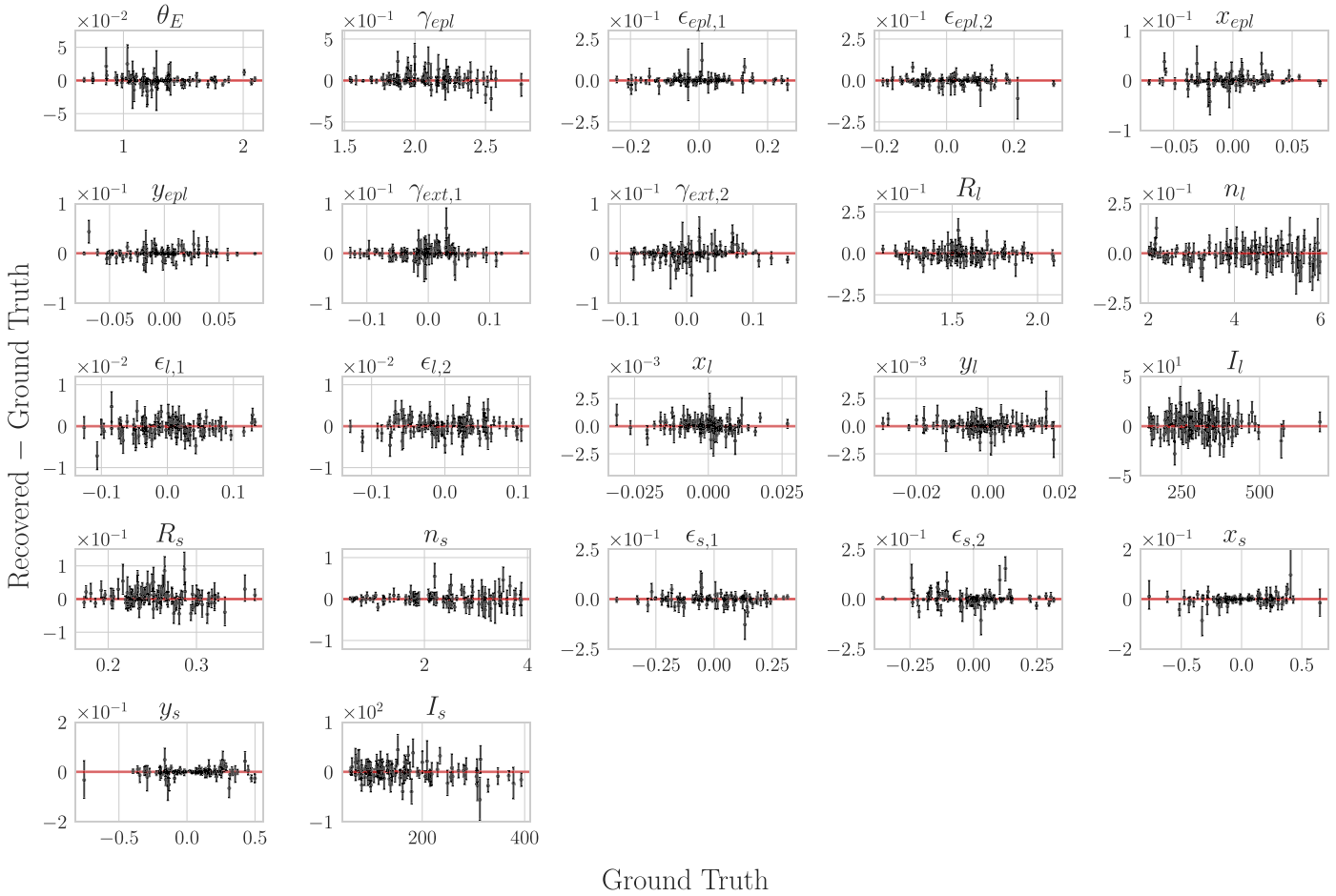


Figure 11. Difference between the recovered parameters and ground truth (input values) for the 100 simulated systems in Figure 10. The points are the mean of the posterior, and the uncertainties correspond to the 68% highest posterior density interval. Note that as n_l and n_s increase, their uncertainties increase as well. This is because the light becomes more compact at higher Sérsic indices, resulting in higher degeneracy between the Sérsic indices and the half-light radii.

single MC chain $f_n|n=1,\ldots,N$ is defined by Sokal (1997):

$$\hat{\rho}(\tau) = \hat{c}(\tau)/\hat{c}(0), \quad \text{where}$$
$$\hat{c}(\tau) = \frac{1}{N-\tau}\sum_{n=1}^{N-\tau}(f_n - \mu_f)(f_{n+\tau} - \mu_f)$$
$$\text{and } \mu_f = \frac{1}{N}\sum_{n=1}^{N} f_n. \quad (19)$$

We emphasize that this autocorrelation is independent of the iteration number. That is, burn-in does not remove autocorrelation, nor does running a chain for a very long time.

Our investigation of `emcee` revealed undesirable characteristics even for low to moderate dimensional spaces, as shown above in the case of the reference system with 22 parameters. While in this regime it is possible that, through tuning and longer sampling time, higher-quality convergence can still be achieved using `emcee`, Betancourt (2018) pointed out that in higher dimensions any gradient-free sampler is likely to be much less efficient compared with HMC. Furthermore, Huijser et al. (2017) found that in moderate ($\sim$50) to high (>100) dimensions affine-invariant ensemble samplers (such as `emcee`) can have more severe problems. They showed that for high-dimensional posteriors, in addition to slow convergence, an affine-invariant sampler can misleadingly appear to converge even when it has not. In strong-lens modeling, it is critical to avoid this pernicious behavior, since models for high-resolution observed data that use complex light profiles such as shapelets (Birrer et al. 2015), wavelets (Galan et al. 2021), or pixelization (Nightingale et al. 2021) can easily have $\gtrsim 50$ parameters. The modeling of perturbations to the smooth lensing potential, whether due to dark matter subhalos or line-of-sight halos, will require even more. If we wish to fit these sophisticated models to observed data, we must be able to do robust inference in spaces of moderate to high dimensions.

2.6. Pipeline Summary and Hyperparameter Settings

Our pipeline is a sequence of three steps, with the ultimate goal of producing a collection of samples from the posterior distribution from which robust statistical inferences can be made. We summarize these three steps in Table 1 and report the hyperparameter and initialization settings that we used for the reference system.

From our experience of using `lenstronomy`, the PSO initialization usually needs to be at least somewhat close to the optimum. With multistart gradient descent, we find this to be unnecessary. While samples that start near the optimum are virtually assured to reach it, as expected, those that start far away can often succeed as well (see Figure 4). This suggests that multistart gradient descent has a much weaker dependence on initialization than PSO. In the next section, we will show the application of our pipeline to 100 simulated systems. We find that the MAP initialization in Table 1 does not need to be adjusted to successfully model these systems, providing further

Table 2
Summary Statistics of Lensing Parameters

Parameter	Mean Error	μ_z	$\langle\hat{R}\rangle$	max $\hat{R}$	$\langle$ESS$\rangle$	min *ESS*
$\boldsymbol{\theta_E}$	−0.00026	-0.04 ± 0.09	1.001	1.013	35465	30846
$\boldsymbol{\gamma_{epl}}$	0.01608	0.12 ± 0.08	1.001	1.017	35407	28045
$\boldsymbol{\epsilon_{epl,1}}$	0.00235	0.08 ± 0.09	1.001	1.011	35590	29438
$\epsilon_{epl,2}$	−0.00159	0.01 ± 0.10	1.001	1.006	35505	31213
$\boldsymbol{x_{epl}}$	0.00031	-0.10 ± 0.10	1.001	1.004	35617	33434
$\boldsymbol{y_{epl}}$	0.00082	0.06 ± 0.09	1.001	1.008	35569	32745
$\boldsymbol{\gamma_{ext,1}}$	0.00088	0.07 ± 0.09	1.001	1.012	35542	27926
$\boldsymbol{\gamma_{ext,2}}$	−0.00060	-0.06 ± 0.09	1.001	1.010	35382	30456
$\boldsymbol{R_l}$	−0.00203	-0.09 ± 0.09	1.000	1.007	35768	33695
$\boldsymbol{n_l}$	−0.00321	-0.05 ± 0.08	1.000	1.008	35777	33752
$\boldsymbol{\epsilon_{l,1}}$	−0.00038	-0.21 ± 0.10	1.000	1.003	35444	33593
$\boldsymbol{\epsilon_{l,2}}$	−0.00006	-0.01 ± 0.10	1.001	1.003	35495	33247
$\boldsymbol{x_l}$	0.00003	0.03 ± 0.09	1.001	1.003	35687	33086
$\boldsymbol{y_l}$	0.00006	0.13 ± 0.08	1.000	1.003	35641	32861
$\boldsymbol{I_l}$	1.20330	0.07 ± 0.08	1.000	1.007	35758	33603
$\boldsymbol{R_s}$	0.00599	0.13 ± 0.09	1.000	1.005	35497	32109
$\boldsymbol{n_s}$	0.01192	0.04 ± 0.09	1.000	1.005	35613	32820
$\boldsymbol{\epsilon_{s,1}}$	−0.00255	-0.09 ± 0.10	1.000	1.004	35556	31342
$\boldsymbol{\epsilon_{s,2}}$	0.00282	0.05 ± 0.10	1.001	1.005	35711	31810
$\boldsymbol{x_s}$	−0.00087	-0.06 ± 0.09	1.001	1.017	35443	26822
$\boldsymbol{y_s}$	−0.00092	0.03 ± 0.08	1.001	1.017	35470	29876
$\boldsymbol{I_s}$	−0.09109	-0.05 ± 0.09	1.000	1.003	35580	33416

Note. We show the errors for the 22 lensing parameters in Figure 11. Mean error denotes the average difference between the recovered parameters and the ground truth for the 100 simulated systems. The notation z denotes errors that have been scaled by the posterior standard deviation. For example, for a given system, if the posterior mean and variance of the Einstein radius are $\mathbb{E}[\theta_E]$, $\mathbb{V}[\theta_E]$ and the ground truth Einstein radius is $\bar{\theta}_E$, then $z[\theta_E] = (\mathbb{E}[\theta_E] - \bar{\theta}_E)/\sqrt{\mathbb{V}[\theta_E]}$. We report the average (over all 100 systems) scaled error μ_z for each parameter and find that they are all consistent with zero bias. We also report statistics for the MC convergence diagnostics, including the mean and extremal values. Specifically, for any given parameter, the max $\hat{R}$ and min *ESS* values are the largest $\hat{R}$ and smallest ESS for that parameter across all 100 simulated systems.

evidence that multistart gradient descent is not sensitive to initialization, so long as the prior is broad and $n_{\rm MAP}$ is sufficiently large. The remaining initializations (for VI and HMC) do not need to be changed either. Furthermore, the hyperparameters in Table 1 that were chosen for modeling the reference system have also been found to suffice for the 100 simulated systems in Section 3.

The total execution time for our reference system is 6 minutes on a cutting-edge A100 GPU (available through NERSC Perlmutter early access[29]). On a GPU node on Perlmutter, which has four A100 GPUs,[30] it takes 105 s (Table 1).

3. Results

To demonstrate the performance of our lens modeling pipeline, we simulate a sample of 100 systems using `lenstronomy` (see Figure 10). The parameters for these systems are sampled from the simulation distribution defined in Equation (8). Our prior, also defined in Equation (8), has the same center as the simulation distribution but has been broadened considerably so that it is less informative.

We apply our modeling pipeline as described in Table 1 to each of these systems and show the excellent agreement with the ground truth (input values) in Figure 11. The hyperparameters listed in Table 1 were roughly tuned (to the appropriate order of magnitude) on the reference system and left unchanged when modeling the sample of 100 simulated systems. The average time to model one simulated system is comparable to the reference system (see Table 1). Moreover, we find that our pipeline consistently exhibits favorable MC convergence (Table 2): even the largest $\hat{R}$ for any parameter over all 100 simulated systems was 1.017, and the smallest ESS was 26,822, an order of magnitude higher than the typical value with `emcee`.

4. Discussion and Conclusion

In this work we present a new framework for modeling strong gravitational lenses that is robust, efficient, and scalable to high-dimensional parameter spaces. We achieve this via algorithmic improvements and extensive use of two technologies. For the former, we use multistart gradient descent in place of PSO, and HMC augmented with VI in place of `emcee`. For the latter, first, massive parallel processing on GPUs allows us to simulate thousands of systems at once, orders of magnitude faster than existing lensing codes that use CPUs. This fast simulation capability is key for efficient forward modeling. Second, automatic differentiation provides access to gradient information that is a highly valuable guide for each step in our pipeline, at virtually no additional computational cost.

We have demonstrated our pipeline's performance on a large set of simulated systems. We make a reasonably general choice for our lens model (EPL + external shear, with lens and source light modeled with Sérsic profiles) in this work. But we emphasize that our modeling methodology is an overarching framework. The capabilities described above are applicable to

[29] https://www.nersc.gov/systems/perlmutter/

[30] Currently, only the JAX implementation of our pipeline supports distributed computing over multiple GPUs, due to the lack of support for distributed computing on TensorFlow (outside of neural networks).

any parameterized lens model. For instance, if we opt instead to use shapelets (Birrer et al. 2015) as a source light model, for small n_{max} (Shajib et al. 2021), only $\sim 10\%$ more computation time is needed. More importantly, as we showed in Section 2.5.2, a gradient-informed modeling pipeline is necessary to do rigorous statistical inference on models with many parameters. Fifty-one of the lensing systems that we discovered in Huang et al. (2020, 2021) have been observed with the Hubble Space Telescope (HST; ID: 15867; PI: Huang). We will apply the `GIGA-Lens` framework to model a subset of these systems and report the results in an upcoming publication (X. Huang et al. 2022, in preparation).

In this work, we have developed the core components for a gradient-based lens modeling framework. There is much room for expansion within this framework. For instance, although we did not find significant multimodality in the posterior for the model we consider in this work (i.e., all local modes have vanishing posterior density compared to the global mode), it is unclear whether this will still be the case for more complex lens and source models. We believe that this can be addressed using more advanced samplers (using HMC as a substrate) such as adiabatic MC (Betancourt 2015), parallel tempering (Earl & Deem 2005), or annealed importance sampling (Neal 2001). The latter is also capable of estimating normalizing constants, which enables the computation of Bayes factors. This is necessary for model comparison, which is particularly useful for tasks such as the modeling of subhalos and line-of-sight low-mass halos.

Finally, the execution time can very likely be significantly shortened from the 105 s reported in this work via a combination of technological and algorithmic improvements. For the former, we plan to use eight A100 GPUs, and we expect that this will cut the execution time roughly in half, bringing the total time to below 1 minute. In addition, further improvement on GPU speed is almost a certainty. For the latter, on one hand, advances in mass matrix adaptation for HMC (Stan Development Team 2021) may allow the VI step to be eliminated, potentially offering up to a factor of two speed gain. On the other hand, the VI step can be improved to fit the posterior exactly (Kingma et al. 2017; Papamakarios et al. 2018), allowing HMC to be eliminated from the pipeline. This framework and its further improvements make it possible, for the first time, that the $\mathcal{O}(10^5)$ strong lenses expected to be discovered in the next-generation surveys can be modeled on a reasonable timescale.

This work was supported in part by the Director, Office of Science, Office of High Energy Physics of the US Department of Energy under contract No. DE-AC025CH11231. This research used resources of the National Energy Research Scientific Computing Center (NERSC), a U.S. Department of Energy Office of Science User Facility operated under the same contract as above and the Computational HEP program in The Department of Energys Science Office of High Energy Physics provided resources through the "Cosmology Data Repository" project (grant No. KA2401022). X.H. acknowledges the University of San Francisco Faculty Development Fund. Support for HST program 15867 was provided by NASA through a grant from the Space Telescope Science Institute, which is operated by the Association of Universities for Research in Astronomy, Inc., under NASA contract NAS 5-26555. S.H.S. thanks the Max Planck Society for support through the Max Planck Research Group. E.J. acknowledges funding from Excellence Initiative of Aix-Marseille University —A*MIDEX, a French "Investissements d'Avenir" program (AMX-19-IET-008—IPhU). Y.S. acknowledges support from the Max Planck Society and the Alexander von Humboldt Foundation in the framework of the Max Planck–Humboldt Research Award endowed by the Federal Ministry of Education and Research. This research used resources of the National Energy Research Scientific Computing Center, a DOE Office of Science User Facility supported by the Office of Science of the U.S. Department of Energy under contract No. DE-AC02-05CH11231 using NERSC award HEP-ERCAP0021270.

Software: TensorFlow (Abadi et al. 2015), TensorFlow Probability (Dillon et al. 2017), JAX (Bradbury et al. 2018), Optax (Hessel et al. 2020), lenstronomy (Birrer & Amara 2018), emcee (Foreman-Mackey et al. 2013), Matplotlib (Hunter 2007), seaborn (Waskom 2021), corner.py (Foreman-Mackey 2016), TinyTim (Krist et al. 2011), NumPy (Harris et al. 2020).

Appendix

Below, we show modeling results for four types of typical systems: folds, cusps, crosses, and doubles (Figure A1.). As shown in Figure A2, for all four systems, the posterior mean agrees (within uncertainty) with the ground truth. For the fold (Figure A2, panel (a)), we point out the clear banana-shaped posterior (for similar examples with cluster lensing, see Jullo et al. 2007), as well as the weaker constraint on γ (the standard deviation here is ± 0.1, compared to the more typical ± 0.05 for the other systems). This is worth keeping in mind when doing density profile slope studies. Furthermore, note the tendency for the VI posterior in (a) to underestimate the

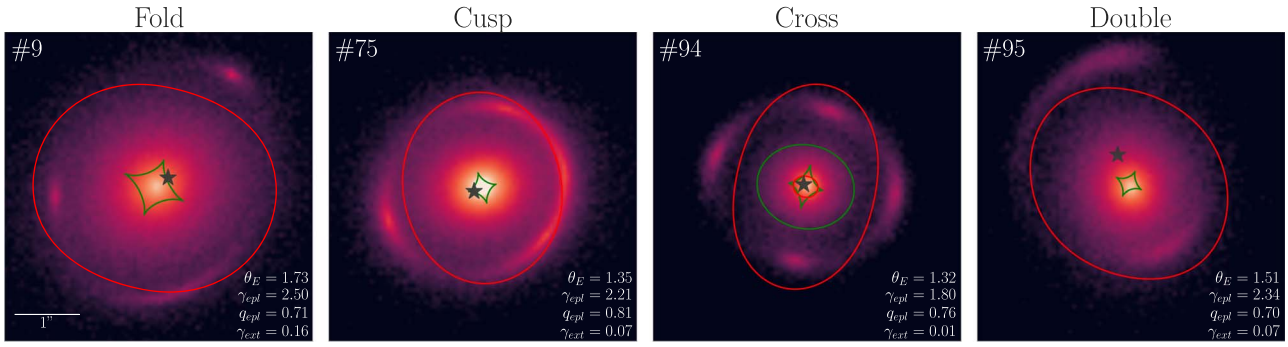
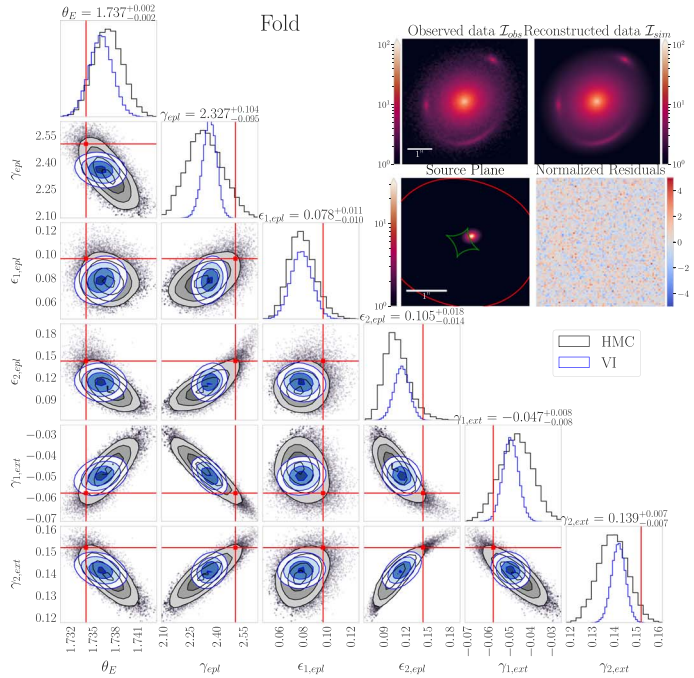
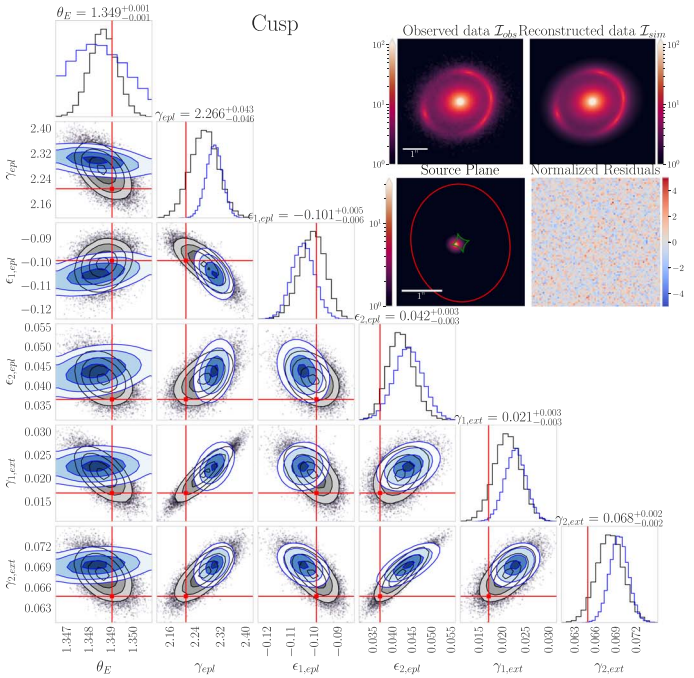


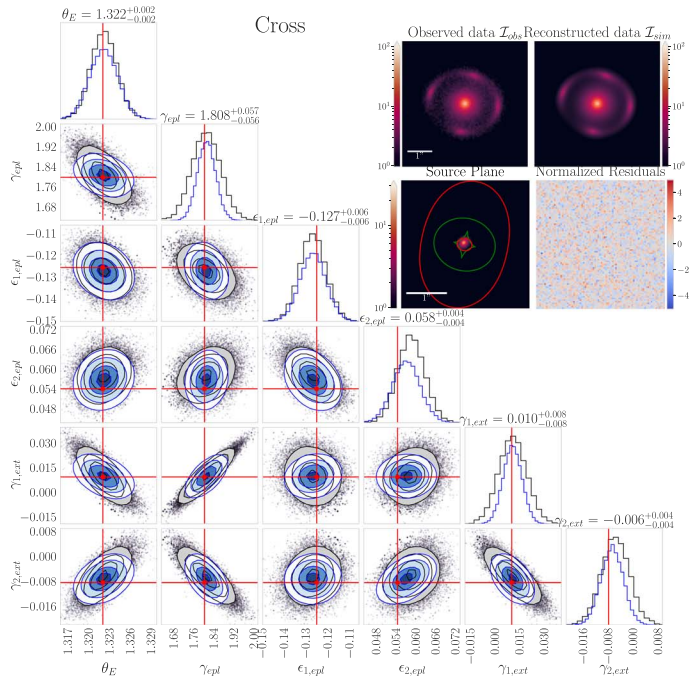
Figure A1. Four archetypal lensing systems selected from Figure 10. The numbers in the upper left corner refer to the ordering in Figure 10. All four systems have a comparable signal-to-noise ratio. The source location is marked with a star, the critical curves are in red, and the caustics are in green. Observe the presence of an inner critical curve and caustic in the cross system, due to the fact that $\gamma_{\rm epl} < 2$ (e.g., O'Riordan et al. 2019).



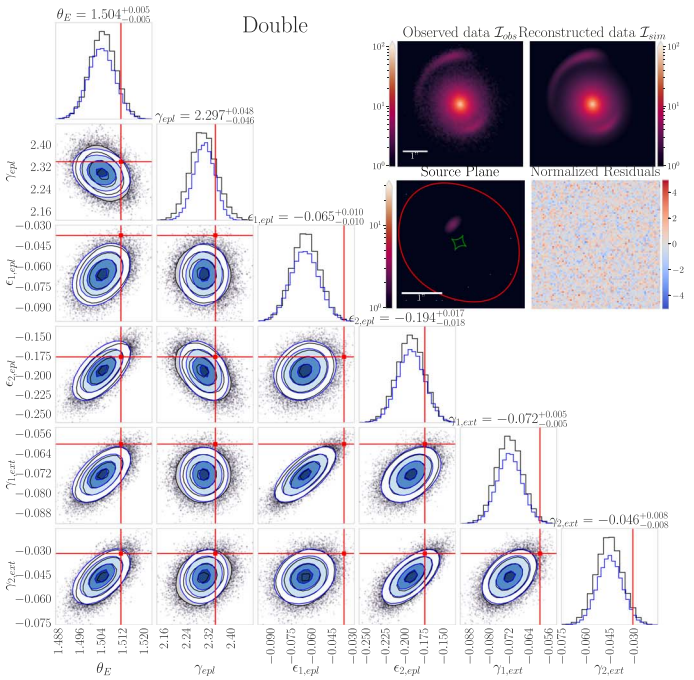
(a) Folds



(b) Cusps



(c) Crosses



(d) Doubles

Figure A2. Modeling results for each of the four archetypal systems. The samples and 0.5σ, 1σ, 1.5σ, and 2σ contours (corresponding to roughly 12%, 39%, 68%, and 86% of the probability mass) for both VI and HMC are shown in blue and gray, respectively.

posterior scale. In contrast, for the cusp (b), the VI posterior overestimates the posterior scale for θ_E. Notably, for the cross (c), the results are qualitatively similar to the results for the reference system (which is also an approximate cross). That is, the degree of agreement between the ground truth and posterior mean is comparable to that of the reference system, and in both cases, the marginals of the VI posterior are similar to those of the true posterior. Finally, for the double (d), the marginals of the VI nearly perfectly agree with those of the true posterior.

ORCID iDs

A. Gu https://orcid.org/0000-0003-2748-7333
X. Huang https://orcid.org/0000-0001-8156-0330
A. S. Bolton https://orcid.org/0000-0002-9836-603X
K. Boone https://orcid.org/0000-0002-5828-6211
A. Dey https://orcid.org/0000-0002-4928-4003
E. Jullo https://orcid.org/0000-0002-9253-053X
S. Perlmutter https://orcid.org/0000-0002-4436-4661
D. Rubin https://orcid.org/0000-0001-5402-4647
E. F. Schlafly https://orcid.org/0000-0002-3569-7421
D. J. Schlegel https://orcid.org/0000-0002-5042-5088
Y. Shu https://orcid.org/0000-0002-9063-698X
S. H. Suyu https://orcid.org/0000-0001-5568-6052

References

Abadi, M., Agarwal, A., Barham, P., et al. 2015, TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems https://www.tensorflow.org/
Barkana, R. 1998, ApJ, 502, 531
Baydin, A. G., Pearlmutter, B. A., Radul, A. A., & Siskind, J. M. 2018, J. Mach. Learn. Res., 18, 1, https://jmlr.org/papers/v18/17-468.html
Betancourt, M. 2018, arXiv:1701.02434
Betancourt, M. J. 2015, arXiv:1405.3489
Birrer, S. 2021, lenstronomy Starting Guide, https://github.com/sibirrer/lenstronomy_extensions/blob/v1.8.1/lenstronomy_extensions/Notebooks/starting_guide.ipynb
Birrer, S., & Amara, A. 2018, PDU, 22, 189
Birrer, S., Amara, A., & Refregier, A. 2015, ApJ, 813, 102
Birrer, S., Shajib, A. J., Galan, A., et al. 2020, A&A, 643, A165
Birrer, Simon, Shajib, Anowar, Gilman, Daniel, et al. 2021, JOSS, 6, 3283
Blei, D. M., Kucukelbir, A., & McAuliffe, J. D. 2017, J. Am. Stat. Assoc., 112, 859
Bolton, A. S., Burles, S., Koopmans, L. V. E., Treu, T., & Moustakas, L. A. 2006, ApJ, 638, 703
Bonvin, V., Courbin, F., Suyu, S. H., et al. 2017, MNRAS, 465, 4914
Bradač, M., Allen, S. W., Treu, T., et al. 2008, ApJ, 687, 959
Bradbury, J., Frostig, R., Hawkins, P., et al. 2018, JAX: Composable Transformations of Python+NumPy Programs, v0.2.5, GitHub, http://github.com/google/jax
Broadhurst, T., Huang, X., Frye, B., & Ellis, R. 2000, ApJL, 534, L15
Cañameras, R., Schuldt, S., Suyu, S. H., et al. 2020, A&A, 644, A163
Çağan Şengül, A., Dvorkin, C., Ostdiek, B., & Tsang, A. 2021, Substructure Detection Reanalyzed: Dark Perturber shown to be a Line-of-Sight Halo, arXiv:2112.00749
Çağan Şengül, A., Tsang, A., Diaz Rivero, A., et al. 2020, PhRvD, 102, 063502
Chianese, M., Coogan, A., Hofma, P., Otten, S., & Weniger, C. 2020, MNRAS, 496, 381
Collett, T. E. 2015, ApJ, 811, 20
Dey, A., Schlegel, D. J., Lang, D., et al. 2019, AJ, 157, 168
Diaz Rivero, A., & Dvorkin, C. 2020, PhRvD, 101, 023515
Dillon, J. V., Langmore, I., Tran, D., et al. 2017, arXiv:1711.10604
Ding, X., Liao, K., Birrer, S., et al. 2021, MNRAS, 504, 5621
Duane, S., Kennedy, A., Pendleton, B. J., & Roweth, D. 1987, PhLB, 195, 216
Dutta, S., & Genton, M. G. 2014, J. Multivar. Anal., 132, 82
Earl, D. J., & Deem, M. W. 2005, PCCP, 7, 3910
Foreman-Mackey, D. 2016, JOSS, 1, 24
Foreman-Mackey, D., Hogg, D. W., Lang, D., & Goodman, J. 2013, PASP, 125, 306
Foxley-Marrable, M., Collett, T. E., Vernardos, G., Goldstein, D. A., & Bacon, D. 2018, MNRAS, 478, 5081
Freedman, W. L., Madore, B. F., Hatt, D., et al. 2019, ApJ, 882, 34
Freedman, W. L., Madore, B. F., Hoyt, T., et al. 2020, ApJ, 891, 57
Galan, A., Peel, A., Joseph, R., Courbin, F., & Starck, J.-L. 2021, A&A, 647, A176
Gelman, A., & Rubin, D. B. 1992, StaSc, 7, 457
Gilman, D., Bovy, J., Treu, T., et al. 2021, MNRAS, 507, 2432
Goldstein, D. A., & Nugent, P. E. 2017, ApJL, 834, L5
Goldstein, D. A., Nugent, P. E., & Goobar, A. 2019, ApJS, 243, 6
Goldstein, D. A., Nugent, P. E., Kasen, D. N., & Collett, T. E. 2018, ApJ, 855, 22
Goobar, A., Amanullah, R., Kulkarni, S. R., et al. 2017, Sci, 356, 291
Grillo, C., Suyu, S. H., Rosati, P., et al. 2015, ApJ, 800, 38
György, A., & Kocsis, L. 2011, J. Artif. Intell., 41, 407
Harris, C. R., Millman, K. J., van der Walt, S. J., et al. 2020, Natur, 585, 357
Hessel, M., Budden, D., Viola, F., et al. 2020, Optax: Composable Gradient Transformation and Optimisation, in JAX!, v0.0.1, GitHub, http://github.com/deepmind/optax
Hezaveh, Y. D., Dalal, N., Marrone, D. P., et al. 2016, ApJ, 823, 37
Hoffman, M., Radul, A., & Sountsov, P. 2021, in Proc. of Machine Learning Research 130, Proc. of The 24th Int. Conf. on Artificial Intelligence and Statistics, ed. A. Banerjee & K. Fukumizu, 3907, https://proceedings.mlr.press/v130/hoffman21a.html
Hoffman, M. D., Blei, D. M., Wang, C., & Paisley, J. 2013, J. Mach. Learn. Res., 14, 1303, https://jmlr.org/papers/v14/hoffman13a.html
Hoffman, M. D., & Gelman, A. 2014, J. Mach. Learn. Res., 15, 1593, https://jmlr.org/papers/v15/hoffman14a.html
Hogg, D. W., & Blandford, R. D. 1994, MNRAS, 268, 889
Horne, K. 1986, PASP, 98, 609
Huang, X., Morokuma, T., Fakhouri, H. K., et al. 2009, ApJ, 707, L12
Huang, X., Storfer, C., Gu, A., et al. 2021, ApJ, 909, 27
Huang, X., Storfer, C., Ravi, V., et al. 2020, ApJ, 894, 78
Huber, S., Suyu, S. H., Ghoshdastidar, D., et al. 2022, A&A, 658, A157
Huijser, D., Goodman, J., & Brewer, B. J. 2017, arXiv:1509.02230
Hunter, J. D. 2007, CSE, 9, 90
Jacobs, C., Collett, T., Glazebrook, K., et al. 2019a, ApJS, 243, 17
Jacobs, C., Collett, T., Glazebrook, K., et al. 2019b, MNRAS, 484, 5330
Jacobs, C., Glazebrook, K., Collett, T., More, A., & McCarthy, C. 2017, MNRAS, 471, 167
Jullo, E., Kneib, J.-P., Limousin, M., et al. 2007, NJPh, 9, 447
Jullo, E., Natarajan, P., Kneib, J. P., et al. 2010, Sci, 329, 924
Kelly, P. L., Filippenko, A. V., Burke, D. L., et al. 2015, Sci, 347, 1459
Kingma, D. P., & Ba, J. 2017, arXiv:1412.6980
Kingma, D. P., Salimans, T., Jozefowicz, R., et al. 2017, arXiv:1606.04934
Kochanek, C. S. 1991, ApJ, 373, 354
Koopmans, L. V. E., & Treu, T. 2002, ApJL, 568, L5
Koopmans, L. V. E., Treu, T., Bolton, A. S., Burles, S., & Moustakas, L. A. 2006, ApJ, 649, 599
Krist, J. E., Hook, R. N., & Stoehr, F. 2011, Proc. SPIE, 8127, 81270J
Martí, R. 2003, in Handbook of Metaheuristics, International Series in Operations Research & Management Science, ed. F. Glover & G. A. Kochenberger (Boston, MA: Springer), 355
Meneghetti, M., Davoli, G., Bergamini, P., et al. 2020, Sci, 369, 1347
Metcalf, R. B., Meneghetti, M., Avestruz, C., et al. 2019, A&A, 625, A119
Millon, M., Galan, A., Courbin, F., et al. 2020, A&A, 639, A101
Narayan, R., & Bartelmann, M. 1997, arXiv:astro-ph/9606001
Neal, R. 2011, in Handbook of Markov Chain Monte Carlo, ed. S. Brooks et al. (Boca Raton, FL: Chapman and Hall/CRC), 113
Neal, R. M. 2001, Stat. Comput., 11, 125
Nesterov, Y. 2004, Introductory Lectures on Convex Optimization (Berlin: Springer), 15
Nightingale, J. W., Hayes, R. G., Kelly, A., et al. 2021, JOSS, 6, 2825
Oguri, M., & Marshall, P. J. 2010, MNRAS, 405, 2579
O'Riordan, C. M., Warren, S. J., & Mortlock, D. J. 2019, MNRAS, 487, 5143
Papamakarios, G., Pavlakou, T., & Murray, I. 2018, arXiv:1705.07057
Pierel, J. D. R., & Rodney, S. 2019, ApJ, 876, 107
Planck Collaboration, Aghanim, N., Akrami, Y., et al. 2020, A&A, 641, A6
Quimby, R. M., Oguri, M., More, A., et al. 2014, Sci, 344, 396
Ranganath, R., Gerrish, S., & Blei, D. 2014, in Proc. of Machine Learning Research 33, Proc. of the Seventeenth Int. Conf. on Artificial Intelligence and Statistics, ed. S. Kaski & J. Corander (Reykjavik: PMLR), 814, https://proceedings.mlr.press/v33/ranganath14.html

Refsdal, S. 1964, MNRAS, 128, 307
Riess, A. G., Casertano, S., Yuan, W., Macri, L. M., & Scolnic, D. 2019, ApJ, 876, 85
Riess, A. G., Yuan, W., Macri, L. M., et al. 2022, ApJL, 934, L7
Ritondale, E., Vegetti, S., Despali, G., et al. 2019, MNRAS, 485, 2179
Rodney, S. A., Brammer, G. B., Pierel, J. D. R., et al. 2021, NatAs, 5, 1118
Rodney, S. A., Strolger, L. G., Kelly, P. L., et al. 2016, ApJ, 820, 50
Rojas, K., Savary, E., Clément, B., et al. 2021, arXiv:2109.00014
Sengupta, S., Basak, S., & Peters, R. 2018, Mach. Learn. Knowl. Extr., 1, 157
Sérsic, J. 1963, Boletín de la Asociación Argentina de Astronomía, 6, 41, http://sedici.unlp.edu.ar/handle/10915/73765
Shajib, A. J., Treu, T., Birrer, S., & Sonnenfeld, A. 2021, MNRAS, 503, 2380
Shu, Y., Bolton, A. S., Brownstein, J. R., et al. 2015, ApJ, 803, 71
Shu, Y., Bolton, A. S., Mao, S., et al. 2018, ApJ, 864, 91
Shu, Y., Bolton, A. S., Moustakas, L. A., et al. 2016, ApJ, 820, 43
Shu, Y., Brownstein, J. R., Bolton, A. S., et al. 2017, ApJ, 851, 48
Sokal, A. D. 1997, Monte Carlo Methods in Statistical Mechanics: Foundations and New Algorithms (Boston, MA: Springer),
Sountsov, P., & Hoffman, M. D. 2021, arXiv:2110.11576
Stan Development Team 2021, Stan Modeling Language Users Guide and Reference Manual, v2.28, https://mc-stan.org/docs/2_28/reference-manual/hmc-algorithm-parameters.html#euclidean-metric
Suyu, S. H., Auger, M. W., Hilbert, S., et al. 2013, ApJ, 766, 70
Suyu, S. H., Huber, S., Cañameras, R., et al. 2020, A&A, 644, A162
Suyu, S. H., Marshall, P. J., Auger, M. W., et al. 2010, ApJ, 711, 201
Tessore, N., & Metcalf, R. 2015, A&A, 580, A79
Treu, T. 2010, ARA&A, 48, 87
Treu, T., & Marshall, P. J. 2016, A&ARv, 24, 11
Vegetti, S., & Koopmans, L. V. E. 2009, MNRAS, 400, 1583
Vegetti, S., Koopmans, L. V. E., Bolton, A., Treu, T., & Gavazzi, R. 2010, MNRAS, 408, 1969
Vegetti, S., Lagattuta, D. J., McKean, J. P., et al. 2012, Natur, 481, 341
Waskom, M. L. 2021, JOSS, 6, 3021
Wengert, R. E. 1964, Commun. ACM, 7, 463
Wong, K. C., Suyu, S. H., Chen, G. C.-F., et al. 2020, MNRAS, 498, 1420
Yahalomi, D. A., Schechter, P. L., & Wambsganss, J. 2017, arXiv:1711.07919
You, K., Long, M., Wang, J., & Jordan, M. I. 2019, arXiv:1908.01878