


Accelerating galaxy dynamical modeling using a neural network for joint lensing and kinematic analyses

Matthew R. Gomer¹ , Sebastian Ertl^{2,3}, Luca Biggio⁴, Han Wang^{2,3}, Aymeric Galan^{3,2,5}, Lyne Van de Vyvere¹, Dominique Sluse¹, Georgios Vernardos^{5,6,7}, and Sherry H. Suyu^{3,2,8}

¹ STAR Institute, Quartier Agora, Allée du Six Août 19c, 4000 Liège, Belgium
e-mail: mgomer@uliege.be

² Max-Planck-Institut für Astrophysik, Karl-Schwarzschild Str. 1, 85748 Garching, Germany
e-mail: ertlseb@mpa-garching.mpg.de

³ Technical University of Munich, TUM School of Natural Sciences, Department of Physics, James-Franck-Straße 1, 85748 Garching, Germany

⁴ Eidgenössische Technische Hochschule Zürich, Rämistrasse 101, 8092 Zürich, Switzerland

⁵ Institute of Physics, Laboratory of Astrophysics – École Polytechnique Fédérale de Lausanne (EPFL), 1290 Versoix, Switzerland

⁶ Department of Astrophysics, American Museum of Natural History, Central Park West and 79th Street, NY 10024, USA

⁷ Department of Physics and Astronomy, Lehman College of the City University of New York, 250 Bedford Park Boulevard, West Bronx, NY 10468, USA

⁸ Academia Sinica Institute of Astronomy and Astrophysics (ASIAA), 11F of ASMA, No. 1, Section 4, Roosevelt Road, Taipei 10617, Taiwan

Received 19 July 2023 / Accepted 16 September 2023

ABSTRACT

Strong gravitational lensing is a powerful tool to provide constraints on galaxy mass distributions and cosmological parameters, such as the Hubble constant, H_0 . Nevertheless, inference of such parameters from images of lensing systems is not trivial as parameter degeneracies can limit the precision in the measured lens mass and cosmological results. External information on the mass of the lens, in the form of kinematic measurements, is needed to ensure a precise and unbiased inference. Traditionally, such kinematic information has been included in the inference after the image modeling, using spherical Jeans approximations to match the measured velocity dispersion integrated within an aperture. However, as spatially resolved kinematic measurements become available via IFU data, more sophisticated dynamical modeling is necessary. Such kinematic modeling is expensive, and constitutes a computational bottleneck that we aim to overcome with our Stellar Kinematics Neural Network (SKiNN). SKiNN emulates axisymmetric modeling using a neural network, quickly synthesizing from a given mass model a kinematic map that can be compared to the observations to evaluate a likelihood. With a joint lensing plus kinematic framework, this likelihood constrains the mass model at the same time as the imaging data. We show that SKiNN's emulation of a kinematic map is accurate to a considerably better precision than can be measured (better than 1% in almost all cases). Using SKiNN speeds up the likelihood evaluation by a factor of ~ 200 . This speedup makes dynamical modeling economical, and enables lens modelers to make effective use of modern data quality in the JWST era.

Key words. gravitational lensing: strong – galaxies: kinematics and dynamics – methods: numerical – cosmological parameters

1. Introduction

Gravitational lensing is a powerful tool that can measure the mass distributions of galaxies, and even offers a method to measure the Hubble parameter, H_0 , which is independent of the distance ladder (Refsdal 1964). In this context, the lens is typically an early-type galaxy (ETG) with a time-variable source, from which time delays between the multiple images can be used alongside a model of the lensing potential to provide a measure of distance. However, lensing degeneracies can introduce systematic uncertainties in the lens mass distribution which must be accounted for to recover an accurate H_0 . The most critical of such degeneracies is the mass-sheet degeneracy (MSD; Falco et al. 1985), which expresses a specific transformation of the mass distribution that leaves all imaging observables invariant, but affects the time delays. This leads to a measure of H_0 which is dependent on the choice of model. One must turn to external information to break the degeneracy and decide which model to keep. One such form of external information is the stellar

kinematics of the lens galaxy, which must be measured and modeled in conjunction with the lensing model if one wishes to derive an accurate value of H_0 (Treu & Koopmans 2002a,b; Koopmans et al. 2003).

The historically established method to use kinematics to break the MSD has consisted of combining lens models with a single aperture measurement of the galaxy's velocity dispersion (e.g., Suyu et al. 2010; Sonnenfeld et al. 2012; Wong et al. 2017; Birrer et al. 2019; Rusu et al. 2020). First, a lens model is performed, which provides a mass model and light model of the lens galaxy. From this model, one can use spherical Jeans approximations (Binney & Tremaine 1987) to estimate a predicted aperture velocity dispersion. Joining these observations together is performed by comparing the predicted velocity dispersion value to the observed value, and combining this kinematic χ^2 together with the lens model likelihood to determine the parameters that match both observations with the maximum total likelihood. Typically this kinematic constraint is included in post-processing, with only a single aperture constraint to help

decide between the already converged lensing-inferred results near the maximum of the imaging likelihood. The main limitation of this approach is that the velocity dispersion is not considered jointly with the lens model to help guide the sampling of the likelihood. Typically, a single aperture measurement only weakly favors a given lens model over another, with relatively little constraining power and a relatively low contribution to the total likelihood.

As the quality of data improves, the constraining power of kinematic information becomes more valuable. Telescopes are increasingly capable of measuring spatially resolved velocity dispersions in galaxies. Through integral field unit (IFU) spectrographs such as the Multi Unit Spectroscopic Explorer (MUSE; Bacon et al. 2010), the *Keck* Cosmic Web Imager (KCWI; Morrissey et al. 2012), and the *James Webb* Space Telescope (JWST) NIRSpec IFU (Yildirim et al. 2020), pixelated maps of velocity measurements for lens systems are becoming available, providing constraints on the mass over a range of radii. If combined with lens models, this information can break the MSD and improve the precision of cosmological measurements (Birrer & Treu 2021; Yıldırım et al. 2023; Shajib et al. 2023).

The challenge is that these data require more sophisticated dynamical models than the spherical Jeans models historically used for this task. Beyond spherical Jeans, the next level of generalization is to allow the model to be axisymmetric. Implementation of this model is possible using the Jeans Anisotropic Multiple Gaussian expansion (JAM; Cappellari 2008) method, which decomposes a mass profile using multiple Gaussian expansion (MGE), deprojects the Gaussian components given an inclination, and calculates the $v_{\text{rms}} = \sqrt{v_{\text{rot}}^2 + \sigma_v^2}$ in the sky plane, where v_{rot} is the rotational velocity and σ_v is the velocity dispersion.

JAM has been used to study the structure of nearby ETGs (Cappellari et al. 2011). Spatially resolved kinematic data measured by the SAURON survey has shown that ETGs come in two distinct kinematic classes: fast rotators and slow rotators (Cappellari et al. 2007; Emsellem et al. 2007). Fast rotators are well matched by an oblate axisymmetric model, and as such are well modeled by JAM (Cappellari 2016). Slow rotators, meanwhile, typically have position angles of their light distributions that are misaligned with the kinematic axis, which implies they tend to have prolate or triaxial shapes (Weijmans et al. 2014; Loubser et al. 2022). As such, the use of JAM carries the implicit assumption that these axes are aligned for a given lens galaxy, which Krajnović et al. (2011) found to be true for approximately 90% of nearby ETGs. Using JAM would provide a natural way to self-consistently model lens systems, except that the calculation is much more expensive than spherical Jeans. Combining more computationally expensive dynamical modeling with the already-expensive lens modeling in a joint inference framework is at present prohibitive without significant computational resources.

Current methods exist that are capable of combining spatially resolved kinematics and lensing information to varying degrees. Barnabè & Koopmans (2007) first created a joint kinematics+lensing modeling code with a goal of studying galaxy structure (see also Barnabè et al. 2009, 2012), and as such it has not been used for H_0 inference. van de Ven et al. (2010) self-consistently compared a lens model of imaging data with an axisymmetric kinematic model of resolved kinematic data, but the models were fit separately. More recently Yıldırım et al. (2020, 2023) implemented a joint lensing and dynamics framework using JAM which is capable of time-delay cosmography,

but is computationally very demanding to fit simultaneously the many lensing and kinematic measurements.

One source of inspiration for this work is that the booming field of machine learning (ML) has helped solve similar problems in related fields (e.g., see review by Huertas-Company & Lausasse 2023). Neural networks (NNs) have been used to replace expensive solver operations in cosmological applications (Albers et al. 2019; Bonici et al. 2022), to replace stellar population synthesis in spectral modeling (Alsing et al. 2020), to extract physical properties from velocity measurements of galaxies (Dawson et al. 2021), and even to speed up gravitational lens modeling itself (Hezaveh et al. 2017; Perreault Levasseur et al. 2017; Pearson et al. 2019; Park et al. 2021; Schuldt et al. 2021, 2023; Biggio et al. 2023). This work applies a similar strategy by using a NN to emulate the expensive kinematics computation step for use within a joint lensing+dynamics modeling framework. Rather than constructing a NN to learn both the kinematics and lensing physics, our NN emulates only the kinematics. This strategy also allows the NN to be implementable in a modular fashion with existing lens modeling frameworks. This ensures both pieces (the lens model and the emulated kinematic model) retain physical meaning independently, as we can both check that realistic kinematics maps are created and leverage existing lens modeling software to handle the lensing physics.

Having originally unveiled the prototype at NeurIPS 2022 (Gomer et al. 2022), in this work we present the Stellar Kinematics Neural Network (SKiNN), which replaces the kinematics computation for a joint lens+kinematics modeling framework. SKiNN is built to emulate JAM, producing a high-resolution velocity map of a galaxy, given a parametric description of the mass and light of the lensing galaxy as input. Figure 1 shows a schematic of such a joint framework, wherein mass and light profiles are modeled through lens modeling and kinematic modeling to evaluate a joint likelihood. We highlight SKiNN's role in orange, which calculates the velocity map associated with a particular model. SKiNN is open source and available for use as a python package¹.

In this work we show that SKiNN is capable of emulating JAM to a high accuracy, and does so at a greatly increased speed. Although SKiNN can be incorporated into virtually any existing lens modeling code, in this work we demonstrate its usage in *lenstronomy*² (Birrer & Amara 2018; Birrer et al. 2021). However, SKiNN is fully differentiable by construction and thus its usage is optimal within fully differentiable lens modeling codes (Gu et al. 2022; Galan et al. 2022; Biggio et al. 2023). This work represents a proof of concept, and as such we restrict ourselves to a training set constructed via relatively simple mass and light models (see Sect. 3), with the mindset that the SKiNN method could be expanded to more general training sets via transfer learning. The positive results obtained with SKiNN in this context suggest that generalizing its approach could pave the way for proper utilization of high-quality data expected from the next generation of telescopes.

This paper is organized as follows. Section 2 reviews the details of JAM, Sect. 3 describes the creation of the training set and SKiNN architecture, Sect. 4 quantifies the performance of SKiNN, and Sect. 5 describes the implementation into a joint framework, Sect. 6 discusses these results, and finally Sect. 7 concludes this work.

¹ <https://github.com/mattgomer/SKiNN>

² <https://github.com/lenstronomy/lenstronomy>

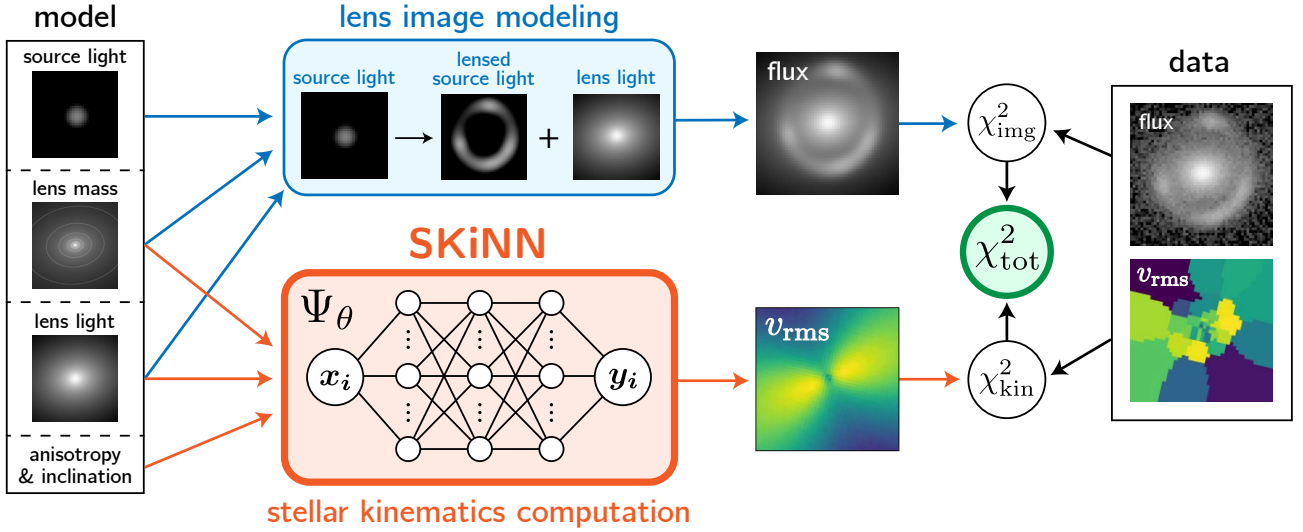


Fig. 1. SKiNN’s role in a joint modeling framework. SKiNN takes, as input, a parametric description of the lens mass profile, lens light profile, anisotropy of the stellar orbits in the lens galaxy, and lens inclination and outputs a v_{rms} map used to evaluate a kinematic likelihood (χ_{kin}^2). This likelihood is then combined with the lensing likelihood (χ_{img}^2) to give a total likelihood (χ_{tot}^2), which is then sampled. In the case of a variable lensed source with measured time delays, a time-delay likelihood (χ_{TD}^2 , not shown in the figure) can be added to the inference (see Sect. 5).

2. Jeans Anisotropic Multiple Gaussian expansion (JAM)

To model the kinematics of galaxies, one must implement some simplifying assumptions, chief among them being the assumption that the galaxy is in a steady state. Under this assumption, the dynamics are completely described by the gravitational potential and the 6D distribution function (DF) $f(\mathbf{x}, \mathbf{v})$, describing the positions and velocities of the tracer population (Cappellari 2016). The problem of reconstructing the DF and the mass distribution from line-of-sight data is inherently underconstrained, worsened by the nonuniqueness of deprojection (Gerhard & Binney 1996). To reduce the dimensionality of the problem, a triaxial shape, axisymmetry, or spherical symmetry is often imposed.

Noting these difficulties, there are three main methods to model dynamics: Schwarzschild modeling, in which a large collection of orbits are computed to represent the whole galaxy (Schwarzschild 1979); N -body simulations, in which particles are simulated to match the density and other observables (Syer & Tremaine 1996); and using the stellar hydrodynamics equations developed by Jeans (1922) to describe the DF within a gravitational potential. For this work, the third method is ideal to predict second-order velocity moments from a mass distribution without the expense of large simulations.

The Jeans equations are derived by multiplying the velocity moments by the collisionless Boltzmann equation, which assumes a steady-state system operating solely under the force of gravity (Binney & Tremaine 1987). Within a cylindrical coordinate system under axisymmetry, the two equations are expressed as:

$$\frac{\overline{v v_R^2} - \overline{v v_\phi^2}}{R} + \frac{\partial(\overline{v v_R^2})}{\partial R} + \frac{\partial(\overline{v v_R v_z})}{\partial z} = -\nu \frac{\partial \Phi}{\partial R} \quad (1)$$

$$\frac{\overline{v v_R v_z}}{R} + \frac{\partial(\overline{v v_z^2})}{\partial z} + \frac{\partial(\overline{v v_R v_z})}{\partial R} = -\nu \frac{\partial \Phi}{\partial z}, \quad (2)$$

where v_R , v_z , and v_ϕ describe the velocity components in cylindrical coordinates, ν is the tracer density (zeroth velocity moment),

and Φ is the gravitational potential, using the notation that $\overline{v v_j v_j} = \int v_i v_j f(\mathbf{x}, \mathbf{v}) d^3 \mathbf{v}$.

Cappellari (2008) introduced JAM as a way to efficiently solve these equations. Two more assumptions are implemented: (1) the velocity ellipsoid is aligned with the coordinate system, and (2) the anisotropy $\beta_z = 1 - \overline{v_z^2}/\overline{v_R^2}$ is a constant spatially. This parameter represents the degree to which stellar orbits are radially aligned, axially aligned, or isotropic. These assumptions reduce Eqs. (1) and (2) to:

$$\frac{\beta_z \overline{v v_z^2} - \overline{v v_\phi^2}}{R} + \frac{\partial(\beta_z \overline{v v_z^2})}{\partial R} = -\nu \frac{\partial \Phi}{\partial R} \quad (3)$$

$$\frac{\partial(\overline{v v_z^2})}{\partial z} = -\nu \frac{\partial \Phi}{\partial z}. \quad (4)$$

These equations allow the tracer density and gravitational potential to yield the second-order velocity moments. From here, they must be projected given an inclination angle i (where $i = 90^\circ$ is edge-on) to give the observed velocity $v_{\text{rms}}^2 = v_{\text{rot}}^2 + \sigma_v^2$ (for projection integrals, see Cappellari 2008), where v_{rot} represents the observed mean LOS velocity often associated with rotation and σ_v represents the velocity dispersion. This inclination angle i is the second parameter introduced (along with β_z) to describe an axisymmetric velocity model.

For a general mass profile, solving these equations to return a v_{rms} can require computationally expensive numerical integrals. However, for a two-dimensional Gaussian profile, the integral can be performed analytically. Leveraging this, JAM uses the Multiple Gaussian Expansion technique (MGE; Emsellem et al. 1994; Cappellari 2002) to describe a particular profile as a sum of many elliptical Gaussian profiles, then efficiently calculates the v_{rms} of the whole profile by summing the contributions of the Gaussian components. Requiring the two parameters β_z and i , the final result is a v_{rms} map which can be compared with observations.

The MGE method offers a physically realistic deprojection interpretation, producing oblate axisymmetric 3D densities, with

axis ratio

$$q_{3D}^2 = \frac{q^2 - \cos^2 i}{\sin^2 i}, \quad (5)$$

where q refers to the 2D projected axis ratio. The deprojection is still not unique (Rybicki 1987), although the nonuniqueness has been found to have only a marginal effect on the dynamics of realistic elliptical galaxies (so-called konus densities; see van den Bosch 1997). The effect worsens for lower inclinations, and the axisymmetric oblate interpretation is only defined if $\cos^2 i < q^2$, as beyond this point the deprojected 3D axis ratio becomes imaginary. This leads to a flattest possible q for a given Gaussian component below which deprojection is not physical.

In summary, JAM solves the axisymmetric Jeans equations and projects the velocity distribution functions along the line of sight using MGE. The end result is a map of v_{rms} which we will use to build the training set for SKiNN.

3. Stellar Kinematics Neural Network (SKiNN)

The goal of SKiNN is to mimic JAM and thus construct a high-resolution map of v_{rms} in the plane of the sky. The input for SKiNN is a list of specific values for the 8 parameters in Table 1 which describe the mass and light distributions of the lensing galaxy, as well as its inclination and anisotropy.

As a proof of concept, the present version of SKiNN is restricted to a single class of mass profiles and light profiles, although in principle the method can be expanded in the future to more general lens models (discussed in Sect. 6). At present, SKiNN is compatible with a Power-law Elliptical Mass Distribution (PEMD; Barkana 1998) and elliptical Sérsic light profiles (Sérsic 1963). These profiles are widely used models in lens modeling. The PEMD mass distribution is expressed in terms of convergence (dimensionless surface density) as

$$\kappa_{\text{PEMD}}(x, y) = \theta_E^{\gamma-1} \left(\frac{3-\gamma}{1+q_M} \right) \left(x^2 + \frac{y^2}{q_M^2} \right)^{\frac{1-\gamma}{2}}, \quad (6)$$

with three parameters: θ_E sets the mass normalization which in the circular case corresponds to the Einstein radius where $\int_0^{2\pi} \int_0^{\theta_E} \kappa(x, y) r dr d\phi = \pi \theta_E^2$; γ represents the slope of the profile with $\gamma = 2$ corresponding to isothermal; and q_M represents the axis ratio of the mass distribution. The Sérsic profile has a 2D light distribution expressed as

$$I(x, y) = A \exp \left[-k \left\{ \left(\frac{\sqrt{x^2 + \left(\frac{y}{q_L} \right)^2}}{R_{\text{Sérsic}}} \right)^{1/n_{\text{Sérsic}}} - 1 \right\} \right]. \quad (7)$$

Three parameters are relevant for the kinematics calculation: $R_{\text{Sérsic}}$ sets the effective radius within which half of the light is contained; $n_{\text{Sérsic}}$ sets the shape of the profile; and q_L represents the axis ratio of the light. The normalization A does not matter for the kinematics calculation, and k is a constant set to ensure the half-light property of $R_{\text{Sérsic}}$ (Sérsic 1963).

3.1. Training set construction

Care is necessary in constructing the training set for the NN. To simply allow all parameters to take any value would waste time

Table 1. Settings for the creation of training sets for the NN.

Parameter	Description	Training set bounds
θ_E	Einstein radius	[0.5, 2'']
γ	2D profile slope (mass)	[1.5, 2.5]
q_M	Axis ratio (mass)	[0.6, 1.0]
q_L	Axis ratio (light)	[0.6, 1.0]
$R_{\text{Sérsic}}$	Sérsic radius (light)	[0.5 θ_E , θ_E]
$n_{\text{Sérsic}}$	Sérsic index (light)	[2, 4]
β_z	Anisotropy	[-0.4, 0.4]
i	Inclination	[arccos(0.6), 90°]
Map resolution	0.02''	
Map size	11''	

Notes. Parameters are sampled uniformly within the range indicated by square brackets.

training over nonphysical solutions, and could worsen the accuracy over physical solutions. On the other hand, the applicability of the end product is limited to the range of the training set, so one must be sure to allow relevant parameters to vary over the whole range of interest to a modeler. We detail here our rationale for how our training set is created.

The training set consists of a paired set of labels and images. The input label \mathbf{x} is a list of 8 parameters used to describe a lens system listed in Table 1. The image \mathbf{y} is a corresponding v_{rms} map of the lens created using JAM as detailed further in this section³. Our training set consists of 4000 randomly generated pairs, our validation set uses 1000 additional random pairs, and our test set uses another 4000 additional random pairs.

We constructed the training set by drawing these parameters uniformly from the ranges indicated in Table 1. The ranges from which these parameters are drawn were chosen based on the priors for the Time Delay Lens Modeling Challenge (TDLMC; Ding et al. 2021), as they reflect typical properties of observed lensed quasars. We allowed for a varying slope through the parameter γ and for the light distributions to have different ellipticities than the mass through the parameters q_M and q_L . Meanwhile, we assumed that the centroid positions and position angles of the mass and light align. These assumptions are based on the observed lens population, for which the mass and light models generally have mostly aligned position angles and centroid positions, but they may have different axis ratios (Shajib et al. 2019). Anisotropy is motivated by the prior used by Yıldırım et al. (2020), and additionally by the JAM models of SAURON observations of early-type galaxies (Cappellari et al. 2007). Minimum inclination was set by the flattest Gaussian of the MGE decomposition.

While the ranges of parameters are described above, numerical limitations lead us to choose a few more specifications for our training set. A singular PEMD mass distribution (i.e., with a vanishing core radius) introduces numerical effects at the lens center. The PEMD mass profile diverges in the center which, in turn, will lead to unphysical v_{rms} maps with very high v_{rms} values in the center. To avoid this, we set a core radius of $r_c = 0.08''$

³ JAM allows one to assign a different q and different β_z for each component of the MGE, creating the capacity to allow these quantities to change with radius. For this work, we set the values of β_z to be constant for each training lens. While we do not directly restrict q to be constant for each MGE component, the profiles we use to build the training set use a constant q , and as such the axis ratio for a given profile ends up not changing significantly with radius.

which is the minimum value for which the test maps produce central v_{rms} values of $<500 \text{ km s}^{-1}$. We consider this a reasonable threshold for numerical effects seeing as ETGs at intermediate redshift rarely have v_{rms} above this value (two cases in a sample of 90 galaxies; Derkenne et al. 2021). In addition, we set a lower bound on the axis ratio q_{MGE} of the Gaussians that are determined in the MGE routine, because single Gaussian components can have a very low q_{MGE} although the sum of all Gaussians follows well the input light and mass profiles. Since the minimum inclination angle is set by the flattest Gaussian of the MGE decomposition, those low q_{MGE} Gaussians would artificially skew the distribution of inclination angles toward 90° . We use $q_{\text{MGE}} \in [0.6, 1]$, which is the same range from which we draw the axis ratio of the mass and light for the training set.

With the parameters described above, the training set was constructed as follows. From a given input label, a PEMD mass distribution and Sérsic light profile is defined. We then use the GLEE lens modeling software (Gravitational Lens Efficient Explorer; Suyu & Halkola 2010; Suyu et al. 2012) to create the light and mass profiles that JAM requires as inputs. These profiles are then fed into JAM, which outputs the v_{rms} image, which we set to have a 551×551 pixel size at $0.02''$ resolution. The creation of each v_{rms} map takes about 21 seconds on a AMD Ryzen Threadripper 3970X CPU. Approximately 90% of this time comes from the MGE decomposition, for which a faster method is possible in the case of constant ellipticity (Shajib 2019), but JAM does not use this method in order to allow for a changing axis ratio with each MGE component.

We create v_{rms} maps with much higher resolution and larger field of view than typical data quality for this construction, with the intention that the map will eventually be rotated and interpolated down to data resolution. Additionally, for a general lens, the values of each pixel must be renormalized by the angular diameter distance to the lens. These steps are not necessary to evaluate the loss function of the NN, and so we postpone their discussion until Sect. 5.

3.2. Architecture

SKiNN can be seen as a function $\Psi_\theta : \mathbb{R}^8 \rightarrow \mathbb{R}^{d \times d}$, mapping an 8-dimensional vector of galaxy mass and light parameters into a $d \times d$ map of v_{rms} in the plane of the sky. Here, θ represents the set of all trainable parameters of the network. Given a training dataset $\mathcal{D} = \{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^N$ where $\mathbf{x} \in \mathbb{R}^8$, $\mathbf{y} \in \mathbb{R}^{d \times d}$ and N is the size of the dataset, the training process consists in finding an optimal set of parameters θ^* , such that a loss function \mathcal{L} , measuring the performance of Ψ on \mathcal{D} , is minimized. In this work, the standard mean-squared-error loss is chosen:

$$\mathcal{L} = \frac{1}{Nd^2} \sum_{i=1}^N \sum_{j=1}^{d^2} (\Psi_\theta(\mathbf{x}_i) - \mathbf{y}_i)^2, \quad (8)$$

which we optimize using the Adam optimizer routine (Kingma & Ba 2014).

While the original prototypes for SKiNN used convolutional architectures (Gomer et al. 2022), the current version of SKiNN is based on the Conditionally Independent Pixel Synthesis architecture (CIPS; Anokhin et al. 2020). Rather than using convolutions, this architecture uses the coordinates of each pixel as well as the parameter vector. The architecture comprises two main components: a mapping network \mathcal{M} and a generator \mathcal{G} . The mapping network takes as input the parameter vector \mathbf{x} and outputs a vector \mathbf{w} , called the style vector. The pixel coordinates are processed by a positional encoding e , which ultimately

results in a significant improvement in the output image quality. The generator takes these encodings as inputs and generates the values corresponding to each pixel, where the style vector \mathbf{w} is used to condition the generator by modulating its weights as indicated in Eq. (2) in Anokhin et al. (2020). Overall, the final image is obtained by passing all the pixel coordinates as well as the parameter vector to the model, that is, $\Psi_\theta(\mathbf{x}_i) = \mathcal{G}(e(\text{mgrid}(d, d)) | \mathcal{M}(\mathbf{x}))$, where $\text{mgrid}(d, d)$ is the meshed grid of pixel coordinates spanning from 0 to d for each coordinate. For more information about the architectural details of the model, we refer the interested reader to Anokhin et al. (2020). The CIPS architecture has been shown to produce photorealistic color images with more realistic power spectra than competing image generators such as StyleGANv2. While our output has only one v_{rms} channel, rather than three color channels, we find that the CIPS architecture results in improved accuracy over our earlier convolutional models. We note that the CIPS architecture requires a graphics processing unit (GPU), and as such a GPU is a hardware requirement for SKiNN.

To facilitate faster training, we exploit the symmetry of the model. Specifically, the NN is trained on only one fourth of the image ($d = 226$), and when generating a full map, the output is mirrored fourfold, creating the 551×551 image. This increase in speed allows us to use the more updated CIPS architecture without a significant loss in training time. Our last step in the output is to set any negative pixels to zero. These negative pixels only happen on rare occasions for individual pixels in the NN output, but would be nonphysical to interpret as v_{rms} .

4. Performance

4.1. Accuracy

We gauge SKiNN's accuracy using the test set. Figure 2 shows both a typical example emulation of a v_{rms} image (top row) and an example of a poor emulation (bottom row) which we consider to be a worst-case scenario. This scenario can arise in rare cases because for some parts of the parameter space, there can be segments of the map where the JAM truth has pixel values of 0 km s^{-1} . We discuss this case in more detail below. Because of this, relative residuals are often not the best metric to quantify error, and so instead we will use the nonrelative error (third column in Fig. 2) to quantify our performance. Figure 3 shows the residuals for 35 systems. We have selected the first ten systems, including our worst-case system (boxed in black in Fig. 3) to explore in the context of joint lensing+kinematics inference in Sect. 5, while the remaining 25 examples are randomly selected from the test set. The 31st (bottom left) system in Fig. 3 is another example with zero-valued pixels, albeit a less egregious case than our worst-case example.

Aside from the two instances with zero-valued pixels, every single pixel across the remaining 33 images has an error of less than 10 km s^{-1} , with typical error significantly less than that. To give an estimate of the accuracy of each image, we calculate the error averaged over each image,

$$\mathcal{E} = \frac{1}{d^2} \sum \Psi_\theta(\mathbf{x}) - \mathbf{y}, \quad (9)$$

as well as the absolute error averaged over each image,

$$|\mathcal{E}| = \frac{1}{d^2} \sum |\Psi_\theta(\mathbf{x}) - \mathbf{y}|. \quad (10)$$

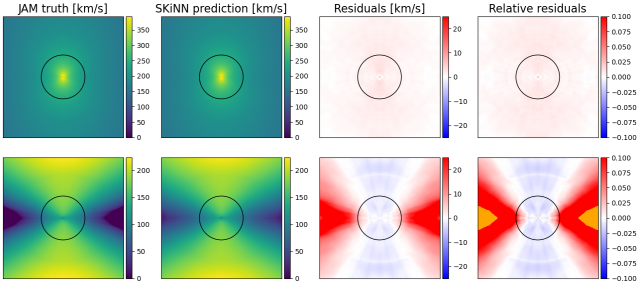


Fig. 2. Accuracy of SKiNN emulation of v_{rms} maps in a typical case (top) and in a particularly poor case (bottom). The first two columns give the truth and predicted v_{rms} maps. Residual differences (third column, km s^{-1}) and relative residuals, evaluated as $(\text{prediction} - \text{truth})/\text{truth}$ (fourth column) are also plotted. Where pixels have zero values, relative residuals are shown in orange. The black circle corresponds to a $2''$ radius, inside which real data are most constraining.

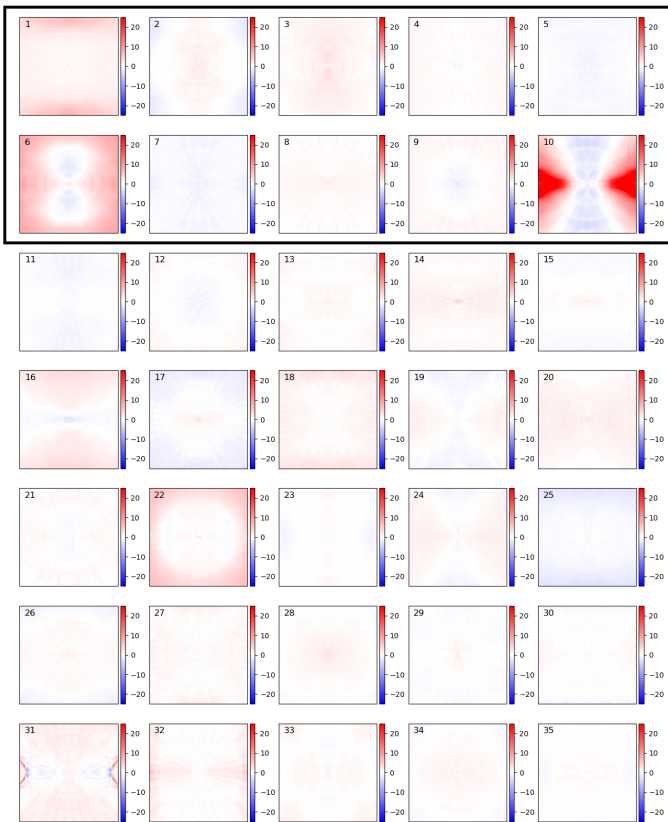


Fig. 3. Example residuals (SKiNN prediction – JAM truth, km s^{-1}) are shown for several randomly selected systems. The first ten systems (boxed in black) are later used for joint inference in Sect. 5. The third and tenth systems are those from Fig. 2.

These measures give an idea of the average error of a typical pixel across an image. We plot these measures for each image in our set of 4000 test images in Fig. 4. Most systems are very well emulated: the mean error averaged over each image \mathcal{E} is centered on 0.16 km s^{-1} , with a spread of approximately 0.88 km s^{-1} , indicating that SKiNN does not introduce a bias by overpredicting or underpredicting the image values. The image-averaged absolute error $|\mathcal{E}|$ has a median of 0.6 km s^{-1} , with the 95th percentile corresponding to an error of 2 km s^{-1} . We find that the distributions of these measures of error are quite similar when considering only the pixels in the innermost $2''$ (black circles in Fig. 2), indicating that the central region, where real data are most sensitive,

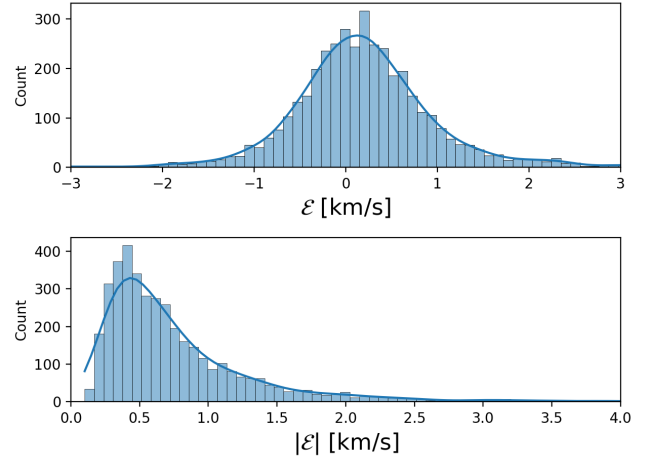


Fig. 4. Error evaluated as $(\text{prediction} - \text{truth})$, averaged across each v_{rms} image in the test set. Top: mean error averaged over each image (Eq. (9)). Bottom: mean absolute error averaged over each image (Eq. (10)).

is equally well emulated. With the knowledge that typical maps of lensing ETGs have $v_{\text{rms}} \sim 200 \text{ km s}^{-1}$, this indicates the emulated images are accurate to better than 1% in nearly all cases. Considering that real observations of v_{rms} have an approximate precision of $6\text{--}7 \text{ km s}^{-1}$ (Cappellari et al. 2011), we consider this an excellent emulation of JAM.

While most systems are very well emulated, there is a tail to the absolute error distribution, and so we looked for the systems with the highest error to see if we could diagnose any weaknesses in the emulation. This led us to discover systems like the worst-case system in the bottom row of Fig. 2, where the truth maps have pixels with a value of zero in the outer regions. These systems comprise approximately 5% of the training and test sets. In the inner regions of the maps, these systems still provide quite good emulations, even in the worst case, indicating that the NN has learned well how to handle these inner features. Diagnosing the cause of the zero-valued pixels, we find that these cases are confined to a region of parameter space with high β_z and high q_L , as shown in Fig. 5. This region of parameter space is unlikely to be relevant for real ETGs, which typically have $\beta_z < 0.7\epsilon$ where $\epsilon = 1 - q$ (Cappellari 2016). SKiNN is less successful with fitting these systems than with the full set, likely because the sharp dropoff to zero must be captured precisely to avoid substantial residuals. The absolute error for these systems is about twice that of the full set, with median 1.4 km s^{-1} and 95th percentile of 3.9 km s^{-1} .

We visually inspect the mean absolute error over the whole parameter space to check to see if there are any other regions in which SKiNN performs better or worse. We find that SKiNN performs slightly worse than average for systems with high β_z or low γ . More quantitatively, in the regions where $\beta_z > 0.3$ or $\gamma < 1.6$, the median absolute error is approximately 0.9 km s^{-1} with 95th percentile of 3.1 km s^{-1} . These regions are also unlikely to be sampled by realistic galaxies based on the β_z constraints of Cappellari (2016) and the fact that ETGs have nearly isothermal slopes (Shajib et al. 2019, 2021).

Altogether, SKiNN emulates JAM to within $\sim 1 \text{ km s}^{-1}$ in most cases, with some parts of the parameter space having errors in the recreation $\sim 5 \text{ km s}^{-1}$. While this is still sufficient for most applications, we note that a user can always use SKiNN to find an approximate solution and afterwards use JAM to confirm the accuracy of converged minimum, which still saves considerable

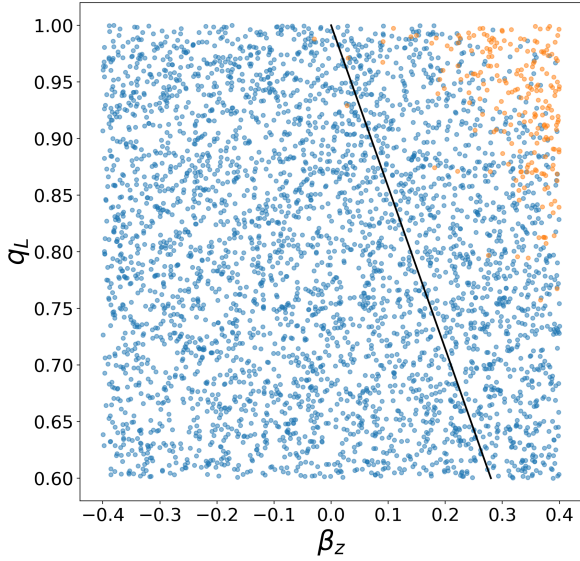


Fig. 5. Scatter plot where each point represents the input parameters for each of the 4000 test-set images. Maps with zero-valued pixels (orange) are predominately confined to a small region of parameter space with high β_z and q_L , compared to those with nonzero pixel values (blue). The black line indicates the cutoff of Cappellari (2016), above which ETGs have not been observed.

computational time. In Sect. 5, we will show that using SKiNN to recover the input parameters results in an accurate constraint for the ten systems in the black box in Fig. 3 shown above, including the worst-case scenario with zero-valued pixels.

4.2. Speed

Using the same machine as before, emulation of a v_{rms} image with SKiNN takes approximately 100 ms using an NVIDIA GeForce GTX 1660 Super GPU, which is about 200 times faster than generating the same image using JAM. This does require the upfront cost to build the training and test sets, which took approximately 2.5 days, as well as the training cost, which took approximately 3 days. However, these costs only need to be performed once, and have already been done for the current setup, for which we make the trained weights publicly available as part of the SKiNN python package. This speed increase makes it possible to sample v_{rms} kinematics within a Markov chain Monte Carlo (MCMC), which we implement and demonstrate in the following section.

5. Joint inference

In a joint implementation, the lens light and mass models are used to generate models of both the imaging data and binned kinematic data which are respectively compared to the observed image of the lensing system and the kinematics of the lensing galaxy (Fig. 1). Each comparison leads to a likelihood estimation, and the summed log likelihood is optimized, hence constraining the lens mass and light parameters by using both imaging and kinematic data. Some parameters, such as the source parameters (not shown in this work), are constrained only from lensing. On the other hand, some parameters, such as the inclination and anisotropy, are specific to the kinematic data and cannot be constrained by the lensing imaging information. Meanwhile, the mass profile and lens light profile are constrained by both types of data, reducing modeling degeneracies.

5.1. Kinematic likelihood

In this section, we describe the steps needed to transform the output from SKiNN into a binned kinematic data and evaluate a likelihood. SKiNN outputs a high-resolution v_{rms} image at a fiducial cosmological distance, which must be rescaled, rotated, sampled, and binned to compare with the observed kinematic data.

The first step necessary is to rescale the map from the fiducial cosmological distances to those of the system. Rescaling with cosmological distance is a straightforward multiplicative factor on the values of each v_{rms} pixel (Birrer et al. 2020; Yıldırım et al. 2023), scaling as

$$v_{\text{rms}} \propto \sqrt{\frac{D_{\Delta t}}{D_d(1+z_d)}}, \quad (11)$$

where D_d is the distance to the deflector, z_d is the deflector redshift, and the time-delay distance $D_{\Delta t} = (1+z_d)\frac{D_d D_s}{D_{ds}}$ can be expressed in terms of the distance to the source D_s and the distance between the deflector and source D_{ds} . In a joint framework, $D_{\Delta t}$ is constrained by the lensing model of imaging and time delays, allowing the kinematics to constrain D_d through Eq. (11). Our training set uses fiducial distances corresponding to a lens redshift of $z_d = 0.5$ and source redshift of $z_s = 2$ for a flat universe with $H_0 = 72 \text{ km s}^{-1} \text{ Mpc}^{-1}$ and $\Omega_m = 0.32$, resulting in $D_d = 1216 \text{ Mpc}$ and $D_{\Delta t} = 2887 \text{ Mpc}$.

Once rescaled according to the lens distances, the SKiNN image must be resampled to match the observed data. SKiNN-generated images are originally aligned with the x -axis centered on the origin, and as such must be first rotated and translated to align with the light distribution before binning down to the lower resolution of the observed data. Finally, the generated image is binned according to the same bins as the observed data, which uses Voronoi binning (Cappellari & Copin 2003) to construct bins of approximately the same signal to noise ratio (S/N). In each bin, the luminosity-weighted v_{rms} is calculated using the modeled light distribution. We are left with a list of predicted v_{pred} values in each of the data bins, which can be compared with observed v_{data} to evaluate a likelihood, which to within a normalization constant is expressed as:

$$\log \mathcal{L}_{\text{kin}} = -\frac{1}{2} \chi_{\text{kin}}^2 = -\frac{1}{2} (v_{\text{pred}} - v_{\text{data}})^T C^{-1} (v_{\text{pred}} - v_{\text{data}}), \quad (12)$$

where C is the covariance matrix giving the precision to which v_{rms} can be measured in each bin. This likelihood term is added to the lensing likelihood to construct a joint likelihood which can be maximized to recover a best-fit model of the lensing+kinematic data.

We have implemented this joint likelihood in `lenstronomy`, which is already capable of recovering a likelihood of a lens model given imaging data. The addition of SKiNN allows `lenstronomy` to evaluate a kinematic likelihood by using SKiNN to construct a v_{rms} map from the eight parameters in \mathbf{x} and then applying the rescaling, translation, and rotation to allow for sampling over the center position, position angle, and cosmological distances. The five parameters which dictate this transformation are listed in Table 2. All in all, a given map from which a kinematic likelihood can be evaluated is produced using the 13 parameters in Tables 1 and 2.

5.2. Testing the joint inference framework

To test the joint implementation of lensing+SKiNN, we created mock lensing and kinematic data which we fit both separately

Table 2. Parameters describing the translation, rotation, and rescaling from a SKiNN output map to data resolution.

Parameter	Description	Sample range
ϕ	Position angle	$[0, 90^\circ]$
x_{center}	x -coordinate of center	$[-0.15'', -0.15'']$
y_{center}	y -coordinate of center	$[-0.15'', -0.15'']$
$D_{\Delta t}$	Time-delay distance	$2887 \text{ Mpc} \pm 20\%$
D_d	Angular distance to the deflector	$1216 \text{ Mpc} \pm 20\%$

Notes. Parameters are sampled uniformly within the range indicated when constructing mock data.

and jointly to demonstrate the utility of SKiNN in a realistic test case. We created ten sets of mock imaging and kinematic data from the boxed systems in Fig 3. These systems were randomly selected, except for the tenth system, which as discussed in Sect. 4 was selected specifically to gauge our accuracy in the worst-case scenario. For a given system, a member of the test set provides the truth velocity map and the eight parameters in Table 1. We then randomly draw truth values for ϕ , x_{center} , y_{center} , $D_{\Delta t}$, and D_d according to the ranges in Table 2. For each system, we add an external shear with magnitude 0.02 and a position angle which is offset from the ellipse major axis by 30° . We show these mock observations in Fig. 6. Mock lensing imaging information is shown on the left. The binned v_{rms} image (far right) is constructed using the deflector light as a weight map for the unbinned v_{rms} , which itself comes from JAM but has been rotated and rescaled according to the truth distance values for a given system. The remainder of this section discusses the creation of these mock data in more detail.

For the mock kinematic data, a light map is needed to define the binning scheme. We construct a mock image of the Sérsic light distribution, where the goal of this mock is to have a realistic binning scheme with ~ 20 – 50 bins to test the SKiNN implementation rather than to perfectly calibrate to the real noise levels of any particular telescope. That said, our brightness and noise settings correspond to setting the integrated brightness to a magnitude 19 galaxy with a 200s exposure time using a zero point of 26 magnitude and read out noise of $21 e^-/s$ with the magnitude of the sky background set to 20 mag. The v_{rms} mock image is created using a 55×55 pixel grid with a resolution of $0.05''$. We evaluate the luminosity-weighted v_{rms} using the test image, which was created using JAM. To mimic the effects of atmospheric seeing, we convolve the high-resolution JAM v_{rms} as well as the light during the weighting step using a Gaussian PSF with a FWHM of $0.1''$ before binning. We apply Voronoi binning to the data with a target $S/N \sim 15$ for each bin. Finally, for each v_{rms} bin, we independently add noise drawn from a Gaussian distribution with a width depending on the bin S/N : the width is set to 10% at $S/N = 10$ and narrows with increasing S/N to a minimum width of 5% when the bin $S/N \geq 40$. This scatter added to the v_{rms} values is intended to represent the imperfect accuracy to which the velocity can be measured from the spectrum of a given bin. The covariance matrix in Eq. (12) is therefore a diagonal matrix where the entry for each bin is the variance of each Gaussian. These binned data are taken to be our observed v_{rms} data, shown in the far right column of Fig. 6.

The lensing imaging data are constructed in a similar fashion to the previously mentioned mock light map used for the kinematics weighting. We use the same exposure times and noise settings, but add in arcs from a lensed source. To create these

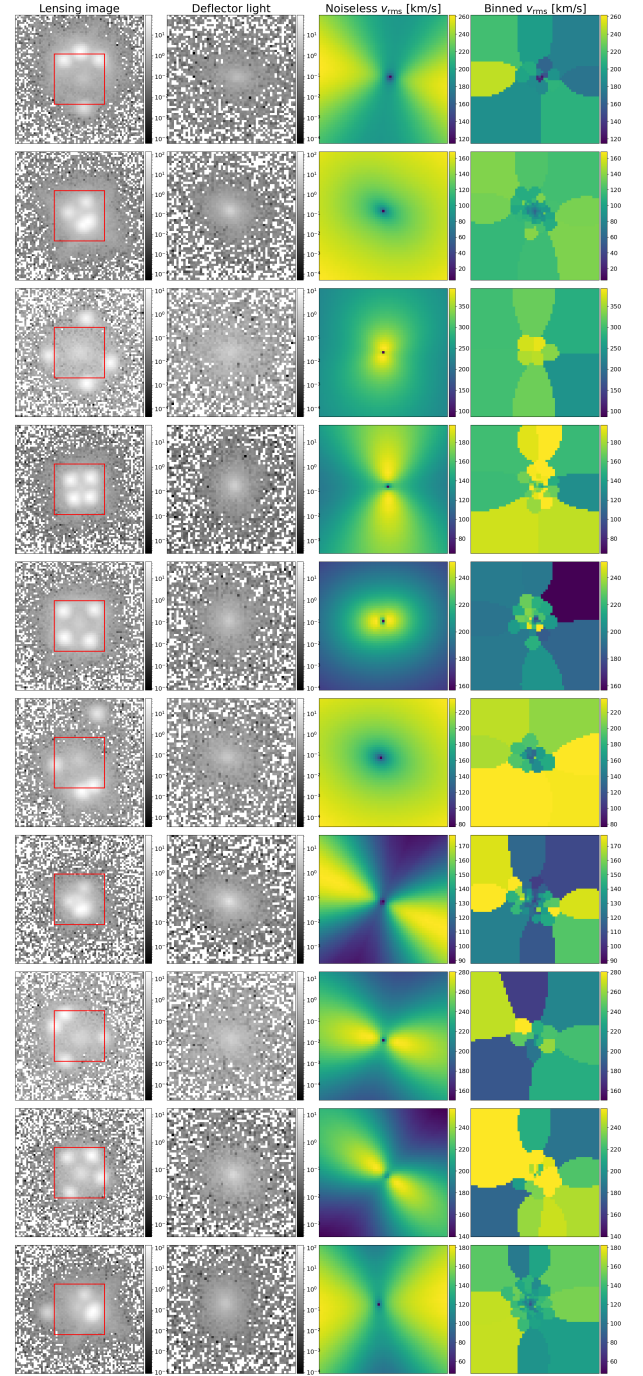


Fig. 6. Mock observations for the 10 systems used for joint inference. From left to right: the lensing image used for lens modeling (in combination with time delays); the deflector light only, with the lensing arcs removed, used to define the kinematics binning and weighting; the noiseless v_{rms} map created by JAM, translated, rotated, and sampled at the data resolution; and the binned v_{rms} with noise added, i.e. the input for the kinematics modeling. The red square in the lensing imaging indicates the tighter field of view of the remaining three panels.

arcs, we randomly draw a source position from within the region of the source plane capable of producing four lensed images. In this position, we place a point source with an intrinsic magnitude of 21.75 in the center of an extended Sérsic source with $R_{\text{Sérsic}} = 0.1$ and intrinsic magnitude of 22.5. Unlike the kinematic light map, we change the image cutout to a 70×70 pixel cutout with a resolution of $0.1''$. Since the lensing imaging

includes point sources, we use the PSF from the TDLMC (Ding et al. 2021), which is more realistic than a Gaussian PSF for imaging data. These changes allow for a larger cutout than the velocity map to ensure that all the lensed images are observed.

In addition to the imaging information, the lens data include mock relative time-delay measurements of the multiple images. These measurements are given a relative uncertainty of 2% or 1 day, whichever is higher, and are used to evaluate a time-delay likelihood which is added into the total likelihood to sample $D_{\Delta t}$.

In the joint framework, the kinematic data and the lens imaging data are combined together along with the time-delay data to optimize a total log likelihood by adding their respective log likelihoods (i.e., treating observations as independent and multiplying their likelihoods), which to within the normalization of L is equivalent to a summed χ^2 :

$$\log L_{\text{tot}} = -\frac{1}{2}\chi_{\text{tot}}^2 = -\frac{1}{2}(\chi_{\text{img}}^2 + \chi_{\text{kin}}^2 + \chi_{\text{TD}}^2). \quad (13)$$

However, we note that the log likelihood of lens imaging data is preponderant over the kinematic and time-delay likelihoods. Indeed, images encompass a lot of pixels (~ 5000) while the number of kinematic bins is restricted to the order of ~ 30 and the time-delay likelihood is restricted to 3 observations at most. When images are taken with the best telescopes, that is with higher resolution, the relative contribution of the imaging data gets even higher (e.g., TDCOSMO systems; Millon et al. 2020). The optimal way to combine likelihoods of different magnitudes may require rescaling individual component likelihoods and is a general problem which is beyond the scope of this work. In our case, we simply combine likelihoods with no rescaling as in Eq. (13) and show that the inclusion of kinematics still improves the overall constraint.

5.3. Results of the joint framework test

For each of the systems in Fig. 6, we model the lensing image and the kinematic image individually as well as jointly.

The lens modeling is performed on the mock images and time delays using a PEMD+external shear model. Strictly speaking this is more optimistic for a lens model than we should expect from real systems because we know in this case that the true mass distribution is the same as our model, meaning we will artificially break the MSD by giving the lens model the correct profile. As such, our lensing inference will reflect a precision for the slope γ which is overestimated relative to a more realistic case. Nonetheless we will show that the kinematic information helps constrain γ to break the MSD in the more natural way by measuring the mass distribution directly.

In all cases, we use an MCMC to sample the posterior parameter space near the truth label. This assumes that a blind inference would find a maximum likelihood for parameters near the truth value from which to start an MCMC, which may not always be true. However, seeing as we ultimately find that the MCMC converges to a region surrounding the truth value, we are satisfied that this maximum would be recovered from a blind starting point, and felt our computational resources were better spent on exploring more systems than on converging from an arbitrary starting point.

We plot the MCMC results for one system in Fig. 7. Sampling using the lensing data alone (blue) provides constraints on most parameters, with no ability to constrain D_d , β_z , or i , since these three parameters are not sampled in a lensing-only model. As such, we have replaced them with uniform distributions for visualization in Fig. 7. The kinematics-only result (orange) typically cannot constrain individual parameters such as θ_E as tightly

as the lensing information can, but it is able to probe all of the parameters of interest. When sampled jointly (green), the resulting posteriors narrow around the truth values, indicating the kinematics constraints have helped the lensing constraints inform the mass distribution. In addition, we also find the external shear closely matches the input for all systems (not plotted). The recovered shear has a value of 0.021 ± 0.003 for both the lensing-only and joint results, indicating the kinematic information does not bias the recovery of the shear. Further testing may be required for an input shear parallel to or orthogonal to the major axis of the lens.

In Fig. 8 we plot the results of the MCMC for the worst-case scenario system with zero-valued pixels. The results are quite similar, possibly because the weighting of the light distribution favors the central regions, as represented by the black circle in Fig. 2. This makes the outer regions where numerical effects are present less relevant. One possible bias in the kinematics-only result is that the truth centroid position (x_{center} and y_{center}) lies just outside the $\sim 1\sigma$ level of the posterior. Fortunately the centroid positions are well constrained by imaging information, and the joint inference strongly favors the correct centroid positions. Noting this, one could also consider for a specific modeling context to fix the centroid positions to the values determined by the lens model. For our tests, we find that fixing these centroid positions does not significantly change the other parameter posterior distributions.

6. Discussion

We have shown that SKiNN offers a fast way to emulate JAM to high accuracy. The speed increase has made it possible to jointly model lensing and kinematics for several systems on a single GPU. Here we summarize the joint inference results of the 10 systems and discuss our outlook for SKiNN in the future.

Seeing as the full cornerplots are somewhat cumbersome to show for all 10 systems, we show the 1D histograms for the 13 parameters, normalized to the truth values in Fig. 9. These plots are constructed by stacking together all the chains of $\Delta x = (x - \text{truth})$ for each parameter x , and plotting all 10 systems as one distribution. We note that this visualization can be lacking for parameters with truth values near the edge of the prior: some distributions, such as $D_{\Delta t}$, β_z , and i were uniform distributions in the corner plots, but when stacked together can result in non-centered distributions. This is because $D_{\Delta t}$ for example cannot go below zero so it is possible to be considerably offset in the positive direction but impossible to be offset in the negative direction. Other distributions, such as γ , may result in narrowed distributions due to the prior width limiting the maximum amount by which a fit can possibly be offset (0.5 on average in the case of γ). Despite these limitations, this visualization serves to demonstrate the precision with which data of this quality is able to constrain the truth.

A caveat that bears repeating is that since this test used a PEMD lens model to fit a PEMD mass distribution, the MSD is artificially broken due to having the external knowledge of the functional form of the mass model. As such, the uncertainty of the lensing-only γ result is underestimated. This makes it all the more important that the kinematics-only γ result be centered on the truth to show that the kinematics helps to break this degeneracy, which is the result we find in Fig. 9.

From the plots, it is clear that most parameters are constrained by lensing alone, with the inclusion of kinematics helping only slightly. Some parameters are constrained even better by lensing alone than by a joint inference such that it may

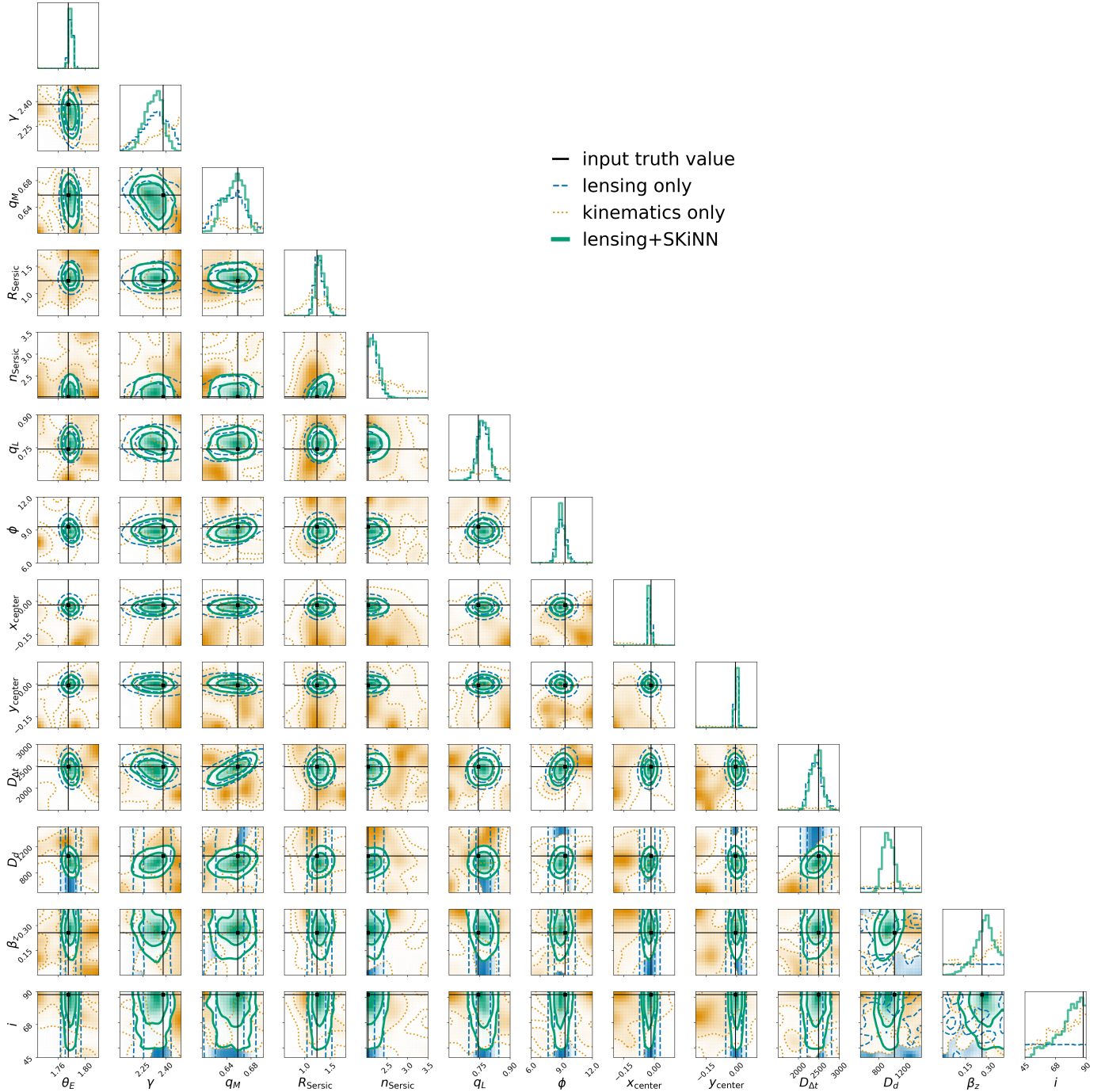


Fig. 7. Corner plot for the MCMC sampling of input parameters using SKiNN for the first example in Fig. 2.

be preferable to simply fix them to the lensing result: namely θ_E , x_{center} , and y_{center} . Meanwhile the anisotropy and inclination are mostly informed by the kinematics, with the lensing helping only slightly. The notable exception to systems being constrained predominately by one form of information or the other is D_d , which clearly requires joint information to constrain. The kinematic constraint can only recover a combination of D_d and $D_{\Delta t}$, a degeneracy which is broken by the lensing measure of $D_{\Delta t}$. The precise constraint of these cosmological distances is critical for a reliable recovery of H_0 . We show that for all systems the combination of lensing and kinematic data is consistently

able to break the degeneracy and recover accurate cosmological distances.

Despite the significant speed increase SKiNN has over JAM, it is still the bottleneck, taking approximately 90% of the time for a each likelihood evaluation. A typical joint fit in this test undergoes nearly 10^6 likelihood evaluations, each of which takes approximately 100 ms using a single GPU (~ 28 GPU-hours per system). It is important to note, however that if the same test were done using JAM, we estimate it would recover the same results but take ~ 5800 CPU-hours per system, a prohibitive cost for modelers without the use of cluster computing.

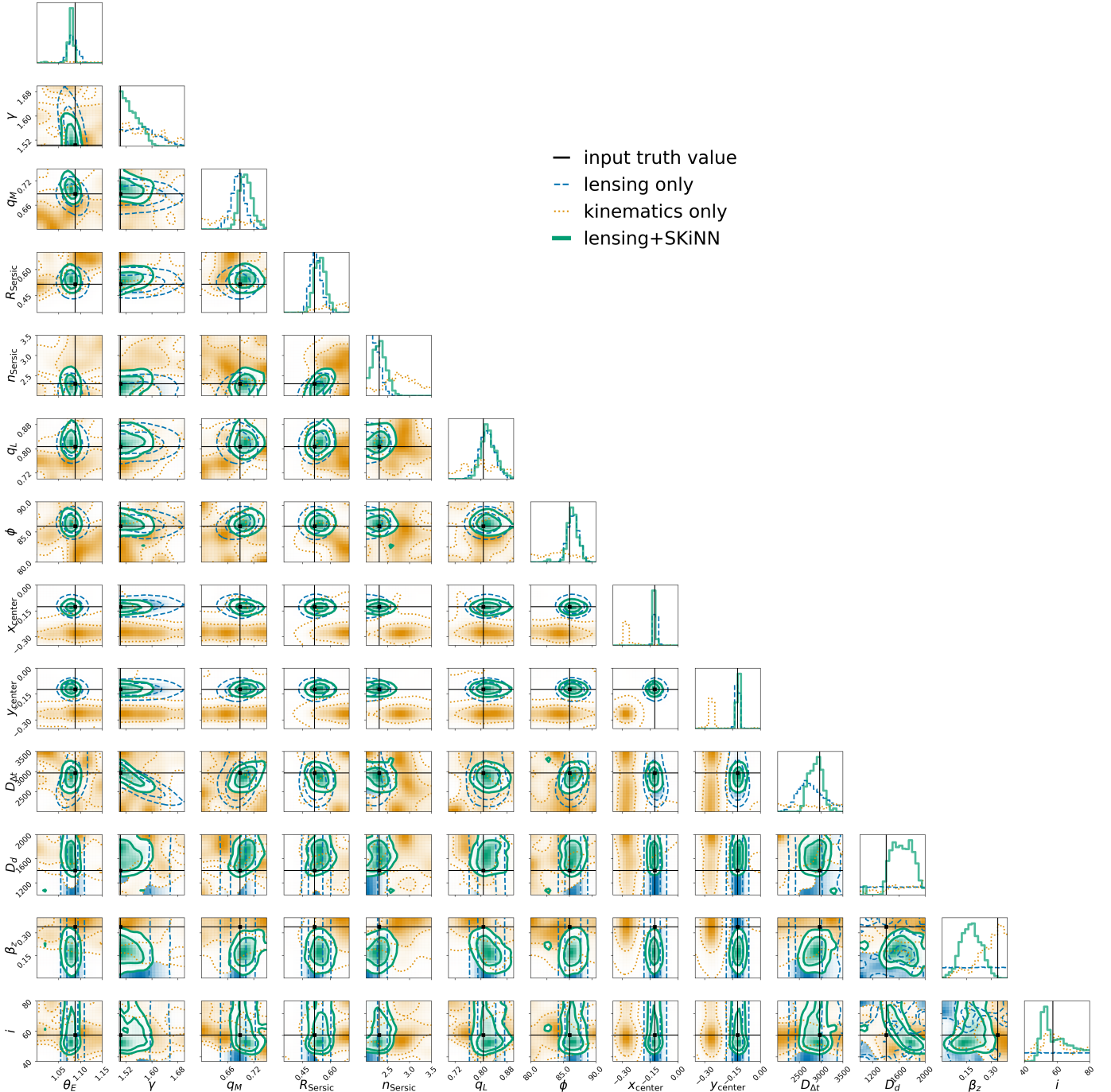


Fig. 8. Corner plot for the MCMC sampling of input parameters using SKiNN for the worst-case system with zero-valued pixels (second example in Fig. 2).

The application of SKiNN in its current form is limited to the assumptions used to create the training set. Constructed with lensed quasar systems in mind, the training set used in this work may not be suitable for all applications, depending on the expected range of parameter values (for example demanding that $R_{\text{Sersic}} < \theta_E$, which is not always true for lower redshift galaxy-galaxy lenses). Being an emulation of JAM, SKiNN inherits JAM's limitations: the mass is assumed to be oblate, axisymmetric, and with a deprojectable axis ratio. SKiNN is limited at present to a constant-anisotropy power-law mass model and a single Sersic light profile, with the position angles and centroids

aligned. Even with these limitations, we note that SKiNN can quickly find an approximate solution within its allowed parameter space, which one could use as a starting point for more sophisticated dynamical modeling, saving significant time.

We anticipate that generalization of the SKiNN method is possible, as we have previously generalized from an isothermal prototype to the power-law mass profile discussed in this work. Such generalization requires recreating (or supplementing) the existing training set and retraining the neural network. For example, one could expand the method beyond the limitations of the JAM model if one had access to a large training set created from

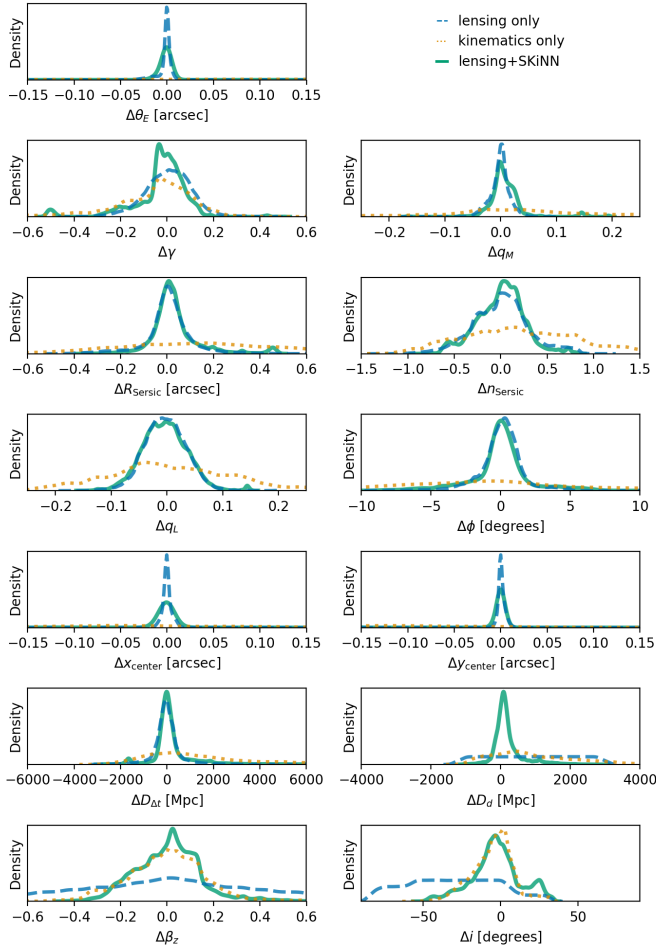


Fig. 9. Recovery of the 13 parameters from Tables 1 and 2 shown as 1D distributions. Plotted results are stacked from those of the ten systems in Sect. 5.

N -body simulations or some other method of emulating galaxy kinematics. In such a case, one could speed up training time by using transfer learning starting from the weights learned in this work.

Thinking more broadly, it may even be possible to modify the method of SKiNN to input a mass and light map instead of a parametric description, providing great flexibility, but this would require an update to the architecture as well as retraining, and is beyond the scope of this current work. However, there is reason to be optimistic about such modifications: we have already iterated on the design of SKiNN from the NeurIPS version (Gomer et al. 2022) to a new architecture and a larger training set, indicating that similar iterations are possible in the future.

7. Conclusion

We present SKiNN, which emulates v_{rms} maps for dynamical modeling in the context of joint modeling with strong gravitational lensing. SKiNN is trained to emulate JAM dynamical modeling, which it does to a high precision (better than 1% in nearly all cases), with no indications of a bias. SKiNN makes it possible to compute kinematic likelihoods approximately 200 times faster than with JAM. This speedup reduces the severity of the computational bottleneck that makes joint kinematic+lensing inference expensive, allowing us to sample

MCMC chains for a kinematic likelihood in a timely manner on a single GPU. We show that these sampling methods are able to recover the input parameters associated with the truth when provided mock data using JAM.

SKiNN is currently available for use as a python package and ready to use for time-delay cosmography applications. While SKiNN is currently implemented in `lenstronomy`, its modular nature makes it suitable to be implemented in other lens modeling codes. SKiNN is fully differentiable, and so its value can be further optimized if used in conjunction with differentiable modeling software. With updates to the training set and/or architecture, the method of SKiNN can likely be further generalized to a wider range of mass and light profiles.

This work represents a step forward in making modern dynamical methodology tractable for strong lens modeling, increasing the complexity from spherical Jeans to axisymmetric Jeans. This increased model complexity allows lens modelers to make proper use of upcoming spatially resolved kinematics from modern JWST-era telescopes.

Acknowledgements. This work originated in the Lensing Odyssey 2021 workshop, and so we would like to acknowledge the organizers and attendees for the fruitful discussions. This project has received funding from the European Research Council (ERC) under the European Union’s Horizon 2020 research and innovation programme (COSMICLENS: grant agreement No 787886; LENSNOVA grant agreement No 771776). S.E. and S.H.S. thank the Max Planck Society for support through the Max Planck Research Group and Max Planck Fellowship for SHS. This research is supported in part by the Excellence Cluster ORIGINS which is funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany’s Excellence Strategy – EXC-2094 – 390783311. A.G. thanks the Swiss National Science Foundation for support through the SNSF Postdoc.Mobility programme. G.V.’s research was made possible by the generosity of Eric and Wendy Schmidt by recommendation of the Schmidt Futures program.

References

- Albers, J., Fidler, C., Lesgourgues, J., Schöneberg, N., & Torrado, J. 2019, *J. Cosmol. Astropart.*, 2019, 028
- Alsing, J., Peiris, H., Leja, J., et al. 2020, *ApJS*, 249, 5
- Anokhin, I., Demochkin, K., Khakhulin, T., et al. 2020, arXiv e-prints [arXiv:2011.13775]
- Bacon, R., Accardo, M., Adjali, L., et al. 2010, *SPIE Conf. Ser.*, 7735, 773508
- Barkana, R. 1998, *ApJ*, 502, 531
- Barnabè, M., & Koopmans, L. V. E. 2007, *ApJ*, 666, 726
- Barnabè, M., Nipoti, C., Koopmans, L. V. E., Vegetti, S., & Ciotti, L. 2009, *MNRAS*, 393, 1114
- Barnabè, M., Dutton, A. A., Marshall, P. J., et al. 2012, *MNRAS*, 423, 1073
- Biggio, L., Vernardos, G., Galan, A., Peel, A., & Courbin, F. 2023, *A&A*, 675, A125
- Binney, J., & Tremaine, S. 1987, *Galactic Dynamics* (Princeton: Princeton University Press)
- Birrer, S., & Amara, A. 2018, *Phys. Dark Universe*, 22, 189
- Birrer, S., & Treu, T. 2021, *A&A*, 649, A61
- Birrer, S., Treu, T., Rusu, C. E., et al. 2019, *MNRAS*, 484, 4726
- Birrer, S., Shajib, A. J., Galan, A., et al. 2020, *A&A*, 643, A165
- Birrer, S., Shajib, A., Gilman, D., et al. 2021, *J. Open Source Softw.*, 6, 3283
- Bonici, M., Biggio, L., Carbone, C., & Guzzo, L. 2022, arXiv e-prints [arXiv:2206.14208]
- Cappellari, M. 2002, *MNRAS*, 333, 400
- Cappellari, M. 2008, *MNRAS*, 390, 71
- Cappellari, M. 2016, *ARA&A*, 54, 597
- Cappellari, M., & Copin, Y. 2003, *MNRAS*, 342, 345
- Cappellari, M., Emsellem, E., Bacon, R., et al. 2007, *MNRAS*, 379, 418
- Cappellari, M., Emsellem, E., Krajnović, D., et al. 2011, *MNRAS*, 413, 813
- Dawson, J. M., Davis, T. A., Gomez, E. L., & Schock, J. 2021, *MNRAS*, 503, 574
- Derkenne, C., McDermid, R. M., Poci, A., et al. 2021, *MNRAS*, 506, 3691
- Ding, X., Treu, T., Birrer, S., et al. 2021, *MNRAS*, 503, 1096
- Emsellem, E., Monnet, G., & Bacon, R. 1994, *A&A*, 285, 723
- Emsellem, E., Cappellari, M., Krajnović, D., et al. 2007, *MNRAS*, 379, 401
- Falco, E. E., Gorenstein, M. V., & Shapiro, I. I. 1985, *ApJ*, 289, L1

- Galan, A., Vernardos, G., Peel, A., Courbin, F., & Starck, J. L. 2022, *A&A*, **668**, A155
- Gerhard, O. E., & Binney, J. J. 1996, *MNRAS*, **279**, 993
- Gomer, M. R., Biggio, L., Ertl, S., et al. 2022, *Machine Learning and the Physical Sciences, NeurIPS 2022 Workshop*
- Gu, A., Huang, X., Sheu, W., et al. 2022, *ApJ*, **935**, 49
- Hezaveh, Y. D., Perreault Levasseur, L., & Marshall, P. J. 2017, *Nature*, **548**, 555
- Huertas-Company, M., & Lanas, F. 2023, *PASA*, **40**, e001
- Jeans, J. H. 1922, *MNRAS*, **82**, 122
- Kingma, D. P., & Ba, J. 2014, arXiv e-prints [arXiv:1412.6980]
- Koopmans, L. V. E., Treu, T., Fassnacht, C. D., Blandford, R. D., & Surpi, G. 2003, *ApJ*, **599**, 70
- Krajnović, D., Emsellem, E., Cappellari, M., et al. 2011, *MNRAS*, **414**, 2923
- Loubser, S. I., Lagos, P., Babul, A., et al. 2022, *MNRAS*, **515**, 1104
- Millon, M., Galan, A., Courbin, F., et al. 2020, *A&A*, **639**, A101
- Morrissey, P., Matuszewski, M., Martin, C., et al. 2012, *SPIE Conf. Ser.*, **8446**, 844613
- Park, J. W., Wagner-Carena, S., Birrer, S., et al. 2021, *ApJ*, **910**, 39
- Pearson, J., Li, N., & Dye, S. 2019, *MNRAS*, **488**, 991
- Perreault Levasseur, L., Hezaveh, Y. D., & Wechsler, R. H. 2017, *ApJ*, **850**, L7
- Refsdal, S. 1964, *MNRAS*, **128**, 307
- Rusu, C. E., Wong, K. C., Bonvin, V., et al. 2020, *MNRAS*, **498**, 1440
- Rybicki, G. B. 1987, *IAU Symp.*, **127**, 397
- Schuldt, S., Suyu, S. H., Meinhardt, T., et al. 2021, *A&A*, **646**, A126
- Schuldt, S., Cañameras, R., Shu, Y., et al. 2023, *A&A*, **671**, A147
- Schwarzschild, M. 1979, *ApJ*, **232**, 236
- Sérsic, J. L. 1963, *Boletín de la Asociación Argentina de Astronomía*, **6**, 41
- Shajib, A. J. 2019, *MNRAS*, **488**, 1387
- Shajib, A. J., Birrer, S., Treu, T., et al. 2019, *MNRAS*, **483**, 5649
- Shajib, A. J., Treu, T., Birrer, S., & Sonnenfeld, A. 2021, *MNRAS*, **503**, 2380
- Shajib, A. J., Mozumdar, P., Chen, G. C. F., et al. 2023, *A&A*, **673**, A9
- Sonnenfeld, A., Treu, T., Gavazzi, R., et al. 2012, *ApJ*, **752**, 163
- Suyu, S. H., & Halkola, A. 2010, *A&A*, **524**, A94
- Suyu, S. H., Marshall, P. J., Auger, M. W., et al. 2010, *ApJ*, **711**, 201
- Suyu, S. H., Hensel, S. W., McKean, J. P., et al. 2012, *ApJ*, **750**, 10
- Syer, D., & Tremaine, S. 1996, *MNRAS*, **282**, 223
- Treu, T., & Koopmans, L. V. E. 2002a, *ApJ*, **575**, 87
- Treu, T., & Koopmans, L. V. E. 2002b, *MNRAS*, **337**, L6
- van den Bosch, F. C. 1997, *MNRAS*, **287**, 543
- van de Ven, G., Falcón-Barroso, J., McDermid, R. M., et al. 2010, *ApJ*, **719**, 1481
- Weijmans, A.-M., de Zeeuw, P. T., Emsellem, E., et al. 2014, *MNRAS*, **444**, 3340
- Wong, K. C., Suyu, S. H., Auger, M. W., et al. 2017, *MNRAS*, **465**, 4895
- Yıldırım, A., Suyu, S. H., & Halkola, A. 2020, *MNRAS*, **493**, 4783
- Yıldırım, A., Suyu, S. H., Chen, G. C. F., & Komatsu, E. 2023, *A&A*, **675**, A21