Astronomy & Astrophysics manuscript no. output
April 7, 2026

A plug-and-play approach with fast uncertainty quantification for weak-lensing mass mapping

H. Leterme[1,2,⋆], A. Tersenov[2,3,⋆⋆], J. Fadili[1], and J.-L. Starck[2,3]

[1] Université Caen Normandie, ENSICAEN, CNRS, Normandie Univ, GREYC UMR 6072, F-14000 Caen, France
[2] Université Paris-Saclay, Université Paris Cité, CEA, CNRS, AIM, 91191, Gif-sur-Yvette, France
[3] Institutes of Computer Science and Astrophysics, Foundation for Research and Technology Hellas (FORTH), Heraklion, Greece

April 7, 2026

ABSTRACT

Context. Upcoming stage-IV surveys such as Euclid and Rubin will deliver vast amounts of high-precision data, opening new opportunities to constrain cosmological models with unprecedented accuracy. A key step in this process is the reconstruction of the dark matter distribution from noisy weak-lensing shear measurements.
Aims. Current deep-learning-based mass-mapping methods achieve high reconstruction accuracy, but either require retraining a model for each new observed sky region (limiting practicality) or rely on slow Markov chain Monte Carlo sampling. Efficient exploitation of future survey data therefore calls for a new method that is accurate, flexible, and fast at inference. In addition, an uncertainty quantification with coverage guarantees is essential for a reliable cosmological parameter estimation.
Methods. We introduce PnPMass, a plug-and-play approach for weak-lensing mass mapping. The algorithm produces point estimates by alternating between a gradient descent step with a carefully chosen data fidelity term and a denoising step implemented with a single deep-learning model trained on simulated data corrupted by Gaussian white noise. We also propose a fast sampling-free uncertainty quantification scheme based on moment networks, with calibrated error bars obtained through conformal prediction to ensure coverage guarantees. Finally, we benchmark PnPMass against model-driven and data-driven mass-mapping techniques.
Results. PnPMass achieves a performance close to that of the currently best deep-learning methods while offering fast inference. It converges in just a few iterations, and it requires only a single training phase, regardless of the noise covariance of the observations. It therefore combines flexibility, efficiency, and reconstruction accuracy while delivering tighter error bars than existing approaches, making it well suited for upcoming weak-lensing surveys.

Key words. cosmology: observations – methods: statistical – gravitational lensing: weak

1. Introduction

In the coming years, next-generation observational instruments such as the Euclid space telescope and the Vera C. Rubin Observatory will deliver galaxy catalogs with unprecedented precision and sky coverage. These surveys are expected to enhance our understanding of large-scale structures in the Universe and to refine existing models of its formation.

A central task in analyzing data from such surveys is mass mapping through weak gravitational lensing. The aim is to reconstruct the distribution of dark matter from noisy shear measurements, which capture the distortion of galaxy shapes caused by inhomogeneous mass distribution along the line of sight (Kaiser & Squires 1993). The resulting mass maps are then used to constrain the plausible ranges of cosmological parameters. An accurate mass mapping is crucial for optimizing cosmological inference. Recent work by Tersenov et al. (2025) showed that replacing the classical Kaiser-Squires method (Kaiser & Squires 1993) with a more advanced algorithm such as MCALens (Starck et al. 2021) can significantly improve the figure of merit (FoM) in realistic weak-lensing pipelines. Moreover, it is essential to rigorously quantify the uncertainty because errors in mass reconstruction can propagate to the estimation of cosmological parameters and might amplify discrepancies.

In recent years, deep-learning-based approaches have achieved the currently best reconstruction accuracy in weak-lensing mass mapping. Two notable examples are DeepMass (Jeffrey et al. 2020), which directly predicts a point estimate from a noisy shear map, and DeepPosterior (Remy et al. 2023), a Bayesian method that samples from the posterior distribution of the mass map. Despite their performances, both methods face limitations that are incompatible with upcoming stage-IV surveys such as Euclid and Rubin, where vast amounts of data are set to be delivered. DeepMass offers fast inference because a single forward pass through the network suffices to produce a point estimate from an initial computationally inexpensive estimate derived from a noisy observation. However, as an end-to-end method, the network is sensitive to factors such as varying noise levels and masks, and it must therefore be retrained for each new observation. DeepPosterior, by contrast, is more flexible and does not require retraining, but inference is computationally expensive since multiple samples must be drawn to obtain a single point estimate.

We introduce PnPMass, a novel deep-learning-based method for weak-lensing mass mapping that like DeepMass generates accurate point estimates without posterior sampling and like DeepPosterior does not require retraining for each new observation. Our approach builds on the plug-and-play (PnP) framework (Venkatakrishnan et al. 2013), which enables efficient reconstruction by integrating deep priors into physical models. PnP

⋆ hubert.leterme@cea.fr
⋆⋆ atersenov@physics.uoc.gr

Article published by EDP Sciences, to be cited as https://doi.org/10.1051/0004-6361/202557652

Table 1: Qualitative comparison of the mass-mapping methods.

	Accurate	Flexible	Fast rec.	Fast UQ
DeepMass	✓	✗*	✓	✓
DeepPosterior	✓	✓	✗	✗
PnPMass (ours)	✓	✓	✓	✓

Notes. *Requires specific retraining for each new observation.

methods have become widespread in computational imaging and physical modeling (Kamilov et al. 2023) and have seen growing theoretical support in recent years (Ryu et al. 2019; Terris et al. 2020; Hurault et al. 2022; Ebner & Haltmeier 2024), including convergence guarantees to a fixed point under appropriate conditions. Our key insight is to show that with appropriate algorithmic design, particularly regarding the data-fidelity term, it is possible to train a denoiser on white Gaussian noise with a standard deviation randomly drawn from a given interval, regardless of the noise characteristics of the actual observation. This allows for a high flexibility across datasets and survey conditions while maintaining fast and accurate reconstructions.

Additionally, we propose an uncertainty quantification (UQ) method for PnPMass that does not rely on sampling. Specifically, we adapted the moment network approach by Jeffrey & Wandelt (2020), originally developed for end-to-end methods such as DeepMass, to our PnP framework. To obtain statistically calibrated uncertainty estimates, we further applied conformalized quantile regression (CQR) (Romano et al. 2019) to derive per-pixel error bars with provable coverage guarantees.

Our experiments, conducted on the κTNG simulated dataset (Osato et al. 2021), demonstrate that PnPMass achieves competitive reconstruction accuracy while remaining flexible. The model is trained only once, unlike DeepMass, and it only requires a few iterations at inference (significantly fewer than those needed by DeepPosterior for sampling). Furthermore, our UQ method, which requires only one additional iteration to compute per-pixel intervals, produces smaller error bars than DeepMass. A qualitative comparison of PnPMass with DeepMass and DeepPosterior is provided in Table 1.

This paper is organized as follows. In Sect. 2 we review the mass-mapping problem and existing data-driven reconstruction methods. In Sect. 3 we present the PnPMass framework. In Sect. 4 we introduce our UQ pipeline, including the method based on moment networks in Sect. 4.1, and CQR-based calibration in Sect. 4.2. Sections 5 and 6 detail our experimental setup and results. We conclude and discuss future directions in Sect. 7.

2. Background on weak-lensing mass mapping

We denote deterministic vectors with bold lower-case Greek or Latin letters (e.g. $\boldsymbol{\kappa}$), while random vectors are represented with bold sans-serif capital letters (e.g. K). Deterministic matrices are indicated by standard bold capital letters (e.g. $\mathbf{A}$). Furthermore, indexing is done using brackets ($\boldsymbol{\kappa}[k]$, $\mathbf{K}[k]$, or $\mathbf{A}[k, l]$).

2.1. The mass-mapping problem

This section provides a brief overview of the weak-lensing mass-mapping problem, for which a comprehensive review has been proposed by Kilbinger (2015). The objective is to recover a convergence map $\boldsymbol{\kappa} \in \mathbb{R}^{K^2}$, which causes isotropic dilations of background galaxies, from a shear map $\boldsymbol{\gamma} \in \mathbb{C}^{K^2}$, which induces anisotropic stretching. Both fields are discretized over a square grid of size $K \times K$, with $K \in \mathbb{N}$, and represented as flattened one-dimensional vectors. In the weak-lensing regime, the convergence is proportional to the projected mass along the line of sight between the background galaxies and the observer (Kaiser & Squires 1993). Reconstructing the convergence field is thus equivalent to estimating the projected mass distribution.

In practice, only variations in $\boldsymbol{\kappa}$ around its mean value can be recovered because of the mass-sheet degeneracy. Similarly, only the variations in $\boldsymbol{\gamma}$ around its mean value contribute to the reconstruction. We therefore assume, without loss of generality, that both $\boldsymbol{\kappa}$ and $\boldsymbol{\gamma}$ are zero-centered. Under this assumption, a first-order relation between shear and convergence can be expressed as

$$\boldsymbol{\gamma} = \mathbf{A}\boldsymbol{\kappa}, \tag{1}$$

where $\mathbf{A} \in \mathbb{C}^{K^2 \times K^2}$ is a linear operator, defined as

$$\mathbf{A} := \mathbf{F}^* \mathbf{P} \mathbf{F}, \tag{2}$$

where $\mathbf{F}$ and its adjoint (Hermitian, or conjugate, transpose) $\mathbf{F}^* \in \mathbb{C}^{K^2 \times K^2}$ represent the two-dimensional discrete Fourier and inverse Fourier transforms, respectively. Moreover, $\mathbf{P} \in \mathbb{C}^{K^2 \times K^2}$ is a diagonal matrix satisfying $\mathbf{P}[0, 0] = 0$, and for all $(k_1, k_2) \in \{0, \dots, K-1\}^2 \setminus \{(0, 0)\}$,

$$\mathbf{P}\left[Kk_1 + k_2, Kk_1 + k_2\right] := \frac{(k_1 + ik_2)^2}{k_1^2 + k_2^2}. \tag{3}$$

Because $\mathbf{P}$ is zero at the zero frequency, the operator $\mathbf{A}$ is not invertible. Its kernel is the subspace of constant vectors, whereas its image is the subspace of zero-mean vectors. The pseudo-inverse of $\mathbf{A}$ is equal to its adjoint $\mathbf{A}^*$, which shares the same kernel and image. Therefore, the convergence map can be recovered via

$$\boldsymbol{\kappa} = \mathbf{A}^* \boldsymbol{\gamma}, \tag{4}$$

yielding an exact reconstruction when both $\boldsymbol{\gamma}$ and $\boldsymbol{\kappa}$ are restricted to the subspace of zero-mean vectors.

A direct observation of the shear, however, is impossible in practice. Instead, a noisy estimator can be obtained by binning a galaxy catalog at the desired resolution and averaging the corresponding galaxy ellipticities within each pixel. Then, (1) becomes

$$\boldsymbol{\gamma} = \mathbf{M}\mathbf{A}\boldsymbol{\kappa} + \boldsymbol{n}, \tag{5}$$

where $\mathbf{M} \in \{0, 1\}^{K^2 \times K^2}$ is a diagonal matrix encoding the observational mask (e.g., survey boundaries or contamination from bright stars in the foreground), and $\boldsymbol{n} \in \mathbb{C}^{K^2}$ is a realization of proper complex Gaussian noise N with zero-mean and diagonal covariance matrix $\boldsymbol{\Sigma} \in \mathbb{R}^{K^2 \times K^2}$. For any pixel $k \in \{1, \dots, K^2\}$ where $\mathbf{M}[k, k] = 1$, the variance is given by

$$\boldsymbol{\Sigma}[k, k] = \sigma_{\text{ell}}^2 / N_k, \tag{6}$$

where $\sigma_{\text{ell}}^2 > 0$ is the variance of intrinsic ellipticities, and N_k is the number of galaxies observed in pixel k.

For pixels $k \in \{1, \dots, K^2\}$ such that $\mathbf{M}[k, k] = 0$ (i.e., within masked regions), we adopted a strategy similar to that proposed by Starck et al. (2021); Remy et al. (2023): the noise variance $\boldsymbol{\Sigma}[k, k]$ was artificially set to a very high value in order to reduce the signal-to-noise ratio (S/N) to near zero. Then, (5) can be approximated by

$$\boldsymbol{\gamma} = \mathbf{A}\boldsymbol{\kappa} + \boldsymbol{n}. \tag{7}$$

In the remainder of the paper, pixels outside and within the mask are referred to as active pixels and masked pixels, respectively.

2.2. Overview of mass-mapping methods

2.2.1. Classical approaches

The most simple method for the mass-mapping problem was proposed by Kaiser & Squires (1993) and is referred to as the Kaiser-Squires (KS) solution. It consists of applying (4) to the noisy shear map $\boldsymbol{\gamma}$, followed by Gaussian smoothing to reduce the noise. More recent approaches have proposed to adopt a variational framework by solving minimization problems of the form

$$\hat{\boldsymbol{\kappa}} \in \operatorname*{argmin}_{\boldsymbol{\kappa}' \in \mathbb{R}^{K^2}} f_{\boldsymbol{\gamma}}(\mathbf{A}\boldsymbol{\kappa}') + g(\boldsymbol{\kappa}'), \tag{8}$$

where $f_{\boldsymbol{\gamma}}(\mathbf{A}\boldsymbol{\kappa}')$ represents a data fidelity term measuring the resemblance between $\boldsymbol{\gamma}$ and $\mathbf{A}\boldsymbol{\kappa}'$, while g denotes a handcrafted regularization function penalizing unexpected solutions. Notable examples include iterative Wiener filtering (Bobin et al. 2012), a forward-backward splitting algorithm with ℓ^2 regularization corresponding to a Gaussian prior with a predefined power spectrum; the GLIMPSE2D algorithm (Lanusse et al. 2016), which enforces sparsity in a wavelet dictionary; and the MCALens algorithm (Starck et al. 2021), which models the solution as a combination of a Gaussian and a sparse component in a wavelet dictionary.

2.2.2. Data-driven approaches

Deep-learning models have demonstrated significantly improved reconstruction accuracy over classical approaches by learning the distribution of convergence maps from large simulation datasets. From a Bayesian perspective, where the convergence map $\boldsymbol{\kappa}$ is viewed as the realization of a random vector $\mathbf{K}$ with an unknown distribution, the inverse problem (7) becomes

$$\boldsymbol{\Gamma} = \mathbf{A}\mathbf{K} + \mathbf{N}, \tag{9}$$

where $\boldsymbol{\Gamma}$ denotes the random vector from which $\boldsymbol{\gamma}$ is drawn.

One of the deep-learning-based approaches is DeepMass (Jeffrey et al. 2020). It takes the noisy shear map $\boldsymbol{\gamma}$ as input, computes a naive estimate such as the KS or iterative Wiener solution, passes it through a deep neural network (typically, a UNet), and outputs an enhanced reconstruction $\hat{\boldsymbol{\kappa}}_{\mathrm{dm}}$. The model is trained on simulated pairs of convergence maps and their corresponding noisy shear maps, $(\boldsymbol{\gamma}_i, \boldsymbol{\kappa}_i)_{i=1}^n$, following the forward model (7).

While DeepMass achieves high reconstruction accuracy, it has a major limitation: it requires retraining for each new observation $\boldsymbol{\gamma}$, as the noise covariance $\boldsymbol{\Sigma}$ depends on the galaxy number density in each pixel. As a result, the training data must be adapted accordingly, which reduces the flexibility of the method.

As an alternative, DeepPosterior (Remy et al. 2023) generates samples from the posterior $p_{\mathbf{K}|\boldsymbol{\Gamma}=\boldsymbol{\gamma}}(\cdot)$ by using a Markov chain Monte-Carlo (MCMC) algorithm. In this approach, the prior $p_{\mathbf{K}}(\cdot)$ is implicitly learned via neural score matching. Then, averaging over the samples yields a point estimate $\hat{\boldsymbol{\kappa}}_{\mathrm{dp}}$. This method directly allows for UQ by computing sample statistics. In contrast to the previous approach, DeepPosterior only needs to be trained once, regardless of the noise properties. However, inference is slow because many samples are required to produce a single estimate.

Other methods have been proposed, but have similar limitations. Shirasaki et al. (2019, 2021); Aoyama et al. (2025) introduced mass-mapping techniques based on conditional generative adversarial networks (GANs) or diffusion models. Recently, Whitney et al. (2025) developed a fast sampling method based on conditional Wasserstein GANs, which significantly reduced the generation time per sample compared to DeepPosterior. Despite their competitive performances, the above methods are limited in the same way as DeepMass: the training phase is specific to a given observation, and the methods therefore lack flexibility. More precisely, given a newly observed sky area or another galaxy survey, these models must be retrained relatively to the new configuration, with a specific mask $\mathbf{M}$ and per-pixel noise level $\boldsymbol{\Sigma}$. Our approach specifically addresses this issue.

3. Proposed approach based on plug-and-play

In this section, we introduce PnPMass, a PnP-based algorithm specifically tailored for the mass-mapping problem. We cast the problem into a real-valued formulation, where we define

$$\tilde{\boldsymbol{\gamma}} := \begin{pmatrix} \operatorname{Re}\boldsymbol{\gamma} \\ \operatorname{Im}\boldsymbol{\gamma} \end{pmatrix} \in \mathbb{R}^{2K^2}, \qquad \tilde{\boldsymbol{n}} := \begin{pmatrix} \operatorname{Re}\boldsymbol{n} \\ \operatorname{Im}\boldsymbol{n} \end{pmatrix} \in \mathbb{R}^{2K^2}, \tag{10}$$

$$\tilde{\mathbf{A}} := \begin{pmatrix} \operatorname{Re}\mathbf{A} \\ \operatorname{Im}\mathbf{A} \end{pmatrix} \in \mathbb{R}^{2K^2 \times K^2}, \quad \tilde{\boldsymbol{\Sigma}} := \frac{1}{2}\begin{pmatrix} \boldsymbol{\Sigma} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\Sigma} \end{pmatrix} \in \mathbb{R}^{2K^2 \times 2K^2}. \tag{11}$$

Then, (7) becomes

$$\tilde{\boldsymbol{\gamma}} = \tilde{\mathbf{A}}\boldsymbol{\kappa} + \tilde{\boldsymbol{n}}, \tag{12}$$

where the noise $\tilde{\boldsymbol{n}}$ is a realization of $\tilde{\mathbf{N}} \sim \mathcal{N}(\mathbf{0}, \tilde{\boldsymbol{\Sigma}})$.

In Sect. 3.1 we introduce a fixed-point iteration composed of a forward step, involving a linear operator $\tilde{\mathbf{B}}$ and a step size $\tau > 0$, followed by a backward step using a deep denoiser $\mathcal{D}_{\hat{\boldsymbol{\theta}}}$. Sect. 3.2 establishes properties on $\mathcal{D}_{\hat{\boldsymbol{\theta}}}$ that enable an accurate recovery of the convergence map $\boldsymbol{\kappa}$ from the noisy shear input $\boldsymbol{\gamma}$. In Sect. 3.3 we show that a suitable choice of $\tilde{\mathbf{B}}$ allows the denoiser to be trained on white Gaussian noise, regardless of the noise covariance matrix $\tilde{\boldsymbol{\Sigma}}$. Sect. 3.4 then derives bounds on τ to guarantee convergence to a fixed point. Additionally, Sect. 3.5 details properties required for the denoiser, including fixed-point conditions and noise-level awareness. Finally, in Sect. 3.6, we present a variant of PnPMass that incorporates prior knowledge of the Gaussian structure of convergence maps outside peak regions dominated by large matter aggregates.

3.1. Fixed-point iteration

Given an observation $\tilde{\boldsymbol{\gamma}} \in \mathbb{R}^{2K^2}$ and a step-size $\tau > 0$, we introduce a forward operator $\mathcal{F}_{\tilde{\boldsymbol{\gamma}}}(\cdot, \tau) : \mathbb{R}^{K^2} \to \mathbb{R}^{K^2}$, defined for any input $\boldsymbol{\kappa}'$ by

$$\mathcal{F}_{\tilde{\boldsymbol{\gamma}}}(\boldsymbol{\kappa}', \tau) := \boldsymbol{\kappa}' + \tau\tilde{\mathbf{B}}(\tilde{\boldsymbol{\gamma}} - \tilde{\mathbf{A}}\boldsymbol{\kappa}'), \tag{13}$$

where $\tilde{\mathbf{B}} \in \mathbb{R}^{K^2 \times 2K^2}$ denotes a linear operator to be defined. We also consider a deep-learning-based denoiser $\mathcal{D}_{\hat{\boldsymbol{\theta}}} : \mathbb{R}^{K^2} \to \mathbb{R}^{K^2}$ with trained parameters $\hat{\boldsymbol{\theta}}$. The PnP method, inspired by standard operator splitting methods in optimization theory for solving inverse problems (see Starck et al. 2015 and Appendix F) consists of iterating the operator $\mathcal{H}_{\tilde{\boldsymbol{\gamma}}}(\cdot, \tau)$ taken as the composition of the forward operator and the trained denoiser,

$$\mathcal{H}_{\tilde{\boldsymbol{\gamma}}}(\cdot, \tau) := \mathcal{D}_{\hat{\boldsymbol{\theta}}} \circ \mathcal{F}_{\tilde{\boldsymbol{\gamma}}}(\cdot, \tau). \tag{14}$$

Toward studying the properties of this operator, we state our main assumptions below.

(H.1) $\mathcal{D}_{\tilde{\theta}} : (\ker \tilde{\mathbf{B}}\tilde{\mathbf{A}})^{\perp} \to (\ker \tilde{\mathbf{B}}\tilde{\mathbf{A}})^{\perp}$;
(H.2) $\mathcal{D}_{\tilde{\theta}}$ is non-expansive on $(\ker \tilde{\mathbf{B}}\tilde{\mathbf{A}})^{\perp}$, i.e., β-Lipschitz continuous on $(\ker \tilde{\mathbf{B}}\tilde{\mathbf{A}})^{\perp}$ with $\beta \leq 1$;
(H.3) $\tilde{\mathbf{B}}\tilde{\mathbf{A}}$ is symmetric positive semidefinite;
(H.4) $\operatorname{ran} \tilde{\mathbf{B}} \subset (\ker \tilde{\mathbf{B}}\tilde{\mathbf{A}})^{\perp}$;
(H.5) Let $\lambda_{\min}$ and $\lambda_{\max} = \|\tilde{\mathbf{B}}\tilde{\mathbf{A}}\|$ denote the minimum and maximum nonzero eigenvalues of $\tilde{\mathbf{B}}\tilde{\mathbf{A}}$, respectively. We also introduce

$$\rho^{\star} := \frac{\lambda_{\max} - \lambda_{\min}}{\lambda_{\max} + \lambda_{\min}}. \tag{15}$$

For fixed $\rho \in [\rho^{\star}, 1[$, the step size τ satisfies

$$\frac{1-\rho}{\lambda_{\min}} \leq \tau \leq \frac{1+\rho}{\lambda_{\max}}. \tag{16}$$

Under the above assumptions, the operator $\mathcal{H}_{\tilde{\boldsymbol{\gamma}}}(\cdot, \tau)$ enjoys the following properties:

Proposition 1. *Under assumptions (H.1)–(H.5), the operator $\mathcal{H}_{\tilde{\boldsymbol{\gamma}}}(\cdot, \tau)$ admits a unique fixed point $\hat{\boldsymbol{\kappa}} \in (\ker \tilde{\mathbf{B}}\tilde{\mathbf{A}})^{\perp}$. Moreover, the sequence of iterates $(\boldsymbol{\kappa}^{(k)})_{k\in\mathbb{N}}$ defined by*

$$\boldsymbol{\kappa}^{(k+1)} = \mathcal{H}_{\tilde{\boldsymbol{\gamma}}}(\boldsymbol{\kappa}^{(k)}, \tau), \tag{17}$$

starting at $\boldsymbol{\kappa}^{(0)} \in (\ker \tilde{\mathbf{B}}\tilde{\mathbf{A}})^{\perp}$, converges linearly to $\hat{\boldsymbol{\kappa}}$ with a convergence rate ρ.

Proof. See Appendix A.

3.2. Which noise should be used to train the denoiser ?

Under the above assumptions, $\hat{\boldsymbol{\kappa}}$ obeys the fixed-point equation

$$\hat{\boldsymbol{\kappa}} = \mathcal{D}_{\tilde{\theta}}\big(\hat{\boldsymbol{\kappa}} + \tau\tilde{\mathbf{B}}(\tilde{\boldsymbol{\gamma}} - \tilde{\mathbf{A}}\hat{\boldsymbol{\kappa}})\big). \tag{18}$$

Assessing whether the fixed point $\hat{\boldsymbol{\kappa}}$ lies sufficiently close to the true convergence map $\boldsymbol{\kappa}$ is a challenging question that falls beyond the scope of this paper. A mathematical framework is nevertheless sketched in Appendix B, thereby constraining the noise level at which the denoiser $\mathcal{D}_{\tilde{\theta}}$ must be trained.

Our intuition can be summarized as follows. Assuming the residual $\hat{\boldsymbol{r}} := \hat{\boldsymbol{\kappa}} - \boldsymbol{\kappa}$ is small enough, we obtain by applying the forward step to the fixed point $\hat{\boldsymbol{\kappa}}$ the approximation

$$\hat{\boldsymbol{\kappa}}_0 := \mathcal{F}_{\tilde{\boldsymbol{\gamma}}}(\hat{\boldsymbol{\kappa}}, \tau) = \boldsymbol{\kappa} + \hat{\boldsymbol{r}} + \tau\tilde{\mathbf{B}}(-\tilde{\mathbf{A}}\hat{\boldsymbol{r}} + \tilde{\boldsymbol{n}}) \tag{19}$$

$$\approx \boldsymbol{\kappa} + \tau\tilde{\mathbf{B}}\tilde{\boldsymbol{n}}. \tag{20}$$

Therefore, application of the operator $\mathcal{H}_{\tilde{\boldsymbol{\gamma}}}$ introduced in (14) to the fixed point $\hat{\boldsymbol{\kappa}}$ yields

$$\hat{\boldsymbol{\kappa}} = \mathcal{D}_{\tilde{\theta}}(\hat{\boldsymbol{\kappa}}_0) \approx \mathcal{D}_{\tilde{\theta}}(\boldsymbol{\kappa} + \tau\tilde{\mathbf{B}}\tilde{\boldsymbol{n}}). \tag{21}$$

Thus, $\mathcal{D}_{\tilde{\theta}}$ acts as a denoiser on the true convergence map $\boldsymbol{\kappa}$ corrupted by a Gaussian noise $\boldsymbol{n}_0 := \tau\tilde{\mathbf{B}}\tilde{\boldsymbol{n}} \in \mathbb{R}^{K^2}$, which is an outcome of

$$\mathbf{N}_0 := \tau\tilde{\mathbf{B}}\tilde{\mathbf{N}} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_0), \quad \text{with} \quad \boldsymbol{\Sigma}_0 := \tau^2\tilde{\mathbf{B}}\tilde{\boldsymbol{\Sigma}}\tilde{\mathbf{B}}^{\top}. \tag{22}$$

In practice, we trained the denoiser by minimizing the empirical quadratic risk,

$$\hat{\boldsymbol{\theta}} \in \operatorname*{argmin}_{\boldsymbol{\theta}} \frac{1}{n}\sum_{i=1}^{n} \Big\|\mathcal{D}_{\boldsymbol{\theta}}\big(\boldsymbol{\kappa}_i + \boldsymbol{n}_{0,i}\big) - \boldsymbol{\kappa}_i\Big\|^2, \tag{23}$$

using a dataset of n training samples. The ground-truth convergence maps $\boldsymbol{\kappa}_i$ were generated from simulations based on the ΛCDM model, and the noisy inputs were computed from $\boldsymbol{\kappa}_i$ by adding noise $\boldsymbol{n}_{0,i}$ drawn according to (22).

3.3. Choice of the operator $\tilde{\mathbf{B}}$

One of the main objectives of this paper is to implement a mass-mapping method whose training phase is independent of the training noise covariance matrix $\tilde{\boldsymbol{\Sigma}}$. Clearly, given our discussion in Sect. 3.2, the covariance matrix $\boldsymbol{\Sigma}_0$ in (22) must be independent of $\tilde{\boldsymbol{\Sigma}}$. To this end, we used the degree of freedom offered by the choice of $\tilde{\mathbf{B}}$. More precisely, we propose to make the following choice:

$$\tilde{\mathbf{B}} := \tilde{\mathbf{A}}^{\top}\tilde{\boldsymbol{\Sigma}}^{-1/2}, \tag{24}$$

in which case (22) and (13) become

$$\boldsymbol{\Sigma}_0 = \tau^2\tilde{\mathbf{A}}^{\top}\tilde{\mathbf{A}} \quad \text{and} \quad \mathcal{F}_{\tilde{\boldsymbol{\gamma}}}(\boldsymbol{\kappa}', \tau) := \boldsymbol{\kappa}' + \tau\tilde{\mathbf{A}}^{\top}\tilde{\boldsymbol{\Sigma}}^{-1/2}(\tilde{\boldsymbol{\gamma}} - \tilde{\mathbf{A}}\boldsymbol{\kappa}'). \tag{25}$$

We caution here that strictly speaking, $\boldsymbol{\Sigma}_0$ is not a covariance matrix as it is only semidefinite positive with $\ker \boldsymbol{\Sigma}_0 = \ker \mathbf{A}$, the subspace of constant vectors. This is not a concern, however, as we are only interested in random perturbations along $(\ker \tilde{\mathbf{B}}\tilde{\mathbf{A}})^{\perp} = (\ker \mathbf{A})^{\perp}$, where $\boldsymbol{\Sigma}_0$ is symmetric positive definite.

In Appendix C we show that with the choice (24), $\tilde{\mathbf{B}}\tilde{\mathbf{A}} = \tilde{\mathbf{A}}^{\top}\tilde{\boldsymbol{\Sigma}}^{-1/2}\tilde{\mathbf{A}}$ verifies assumptions (H.3) and (H.4). We even have the equality $\operatorname{ran} \tilde{\mathbf{B}} = (\ker \tilde{\mathbf{B}}\tilde{\mathbf{A}})^{\perp}$.

(25) shows that the residual $\tilde{\boldsymbol{\gamma}} - \tilde{\mathbf{A}}\boldsymbol{\kappa}'$ is multiplied by $\tilde{\boldsymbol{\Sigma}}^{-1/2}$, which corresponds to a noise-whitening step. This is in a stark contrast with the choice $\tilde{\mathbf{B}} := \tilde{\mathbf{A}}^{\top}\tilde{\boldsymbol{\Sigma}}^{-1}$, which yields $\mathcal{F}_{\tilde{\boldsymbol{\gamma}}}(\boldsymbol{\kappa}', \tau) := \boldsymbol{\kappa}' + \tau\tilde{\mathbf{A}}^{\top}\tilde{\boldsymbol{\Sigma}}^{-1}(\tilde{\boldsymbol{\gamma}} - \tilde{\mathbf{A}}\boldsymbol{\kappa}')$. This alternative choice is the one dictated by the standard variational (or Bayesian MAP) perspective with a data fidelity $\frac{1}{2}\big\|\tilde{\boldsymbol{\Sigma}}^{-1/2}(\tilde{\boldsymbol{\gamma}} - \tilde{\mathbf{A}}\boldsymbol{\kappa}')\big\|^2$ as negative log-likelihood, whose gradient is $\tilde{\mathbf{A}}^{\top}\tilde{\boldsymbol{\Sigma}}^{-1}(\tilde{\boldsymbol{\gamma}} - \tilde{\mathbf{A}}\boldsymbol{\kappa}')$. This clearly shows that the widely adopted optimization or Bayesian perspective on PnP is not necessarily the most appropriate and might even be misleading in the PnP context (see also the discussion in Appendix F). The fixed-point perspective, on the other hand, offers more insight and flexibility.

In Appendix D we show that

$$\tilde{\mathbf{A}}^{\top}\tilde{\mathbf{A}} = \mathbf{I} - \frac{1}{K^2}, \tag{26}$$

(the constant $1/K^2$ is subtracted from all matrix elements). In practice, the image size K is sufficiently large to neglect the non-diagonal elements. Therefore, from (25) and (26), it is valid to take the approximation

$$\boldsymbol{\Sigma}_0 \approx \tau^2\mathbf{I}. \tag{27}$$

In turn, the denoiser $\mathcal{D}_{\boldsymbol{\theta}}$ is trained on zero-mean white Gaussian noise with a standard deviation equal to the step size τ. In Sect. 3.4 we discuss the selection of an appropriate range for τ.

3.4. Step-size selection

As discussed in Sect. 3.1, the existence and uniqueness of a fixed point as well as convergence of the PnP iteration (17) to this fixed point require that τ satisfies (H.5). When we substitute the expression for $\tilde{\mathbf{B}}$ from (24), condition (16) specializes to

$$\frac{1-\rho}{\lambda_{\min}} \leq \tau \leq \frac{1+\rho}{\lambda_{\max}}, \quad \text{where} \quad \lambda_{\max} = \big\|\tilde{\mathbf{A}}^{\top}\tilde{\boldsymbol{\Sigma}}^{-1/2}\tilde{\mathbf{A}}\big\|. \tag{28}$$

It is well known that $\lambda_{\max}$ can be accurately computed via the power iteration method. In Appendix E we also discuss how the power iteration can be used to efficiently compute $\lambda_{\min}$.

A simple estimate of the upper bound for τ can also be obtained via an upper bound of $\lambda_{\max}$. We observe that

$$\begin{aligned}\left\|\tilde{\mathbf{A}}^\top\tilde{\boldsymbol{\Sigma}}^{-1/2}\tilde{\mathbf{A}}\right\| &\leq \left\|\tilde{\mathbf{A}}^\top\tilde{\mathbf{A}}\right\|\left\|\tilde{\boldsymbol{\Sigma}}^{-1/2}\right\| \\ &\leq \left\|\tilde{\boldsymbol{\Sigma}}^{-1/2}\right\| = 1/\min_k \tilde{\sigma}_k,\end{aligned}$$

where we used (26), and $\tilde{\sigma}_k := \tilde{\boldsymbol{\Sigma}}[k,\, k]^{1/2}$ denotes the noise standard deviation at pixel k. Therefore,

$$\tau \leq (1+\rho)\min_k \tilde{\sigma}_k, \tag{29}$$

is a sufficient condition for the right-hand side of (28) to hold. Thus, the choice of the largest step size is essentially governed by the lowest noise level in the input shear map, that is, the pixel with the largest number of observed galaxies.

3.5. Denoiser training

3.5.1. Ensuring convergence to a fixed point

The denoiser ought to comply with (H.1) and (H.2) for Proposition 1 to hold. As discussed in Sect. 3.3 (see also Appendix C), with the choice of $\tilde{\mathbf{B}}$ given in (24), we have

$$(\ker\tilde{\mathbf{B}}\tilde{\mathbf{A}})^\perp := \ker(\tilde{\mathbf{A}})^\perp, \tag{30}$$

that is, the subspace of zero-mean convergence maps. In turn, (H.1) tells us that the denoiser must not introduce any bias, that is, the denoised version of a zero-mean map must also be zero-mean. This is a natural and expected property. It is consistent with the mass-sheet degeneracy property discussed in Sect. 2.1 and can be enforced by explicitly appending a mean-centering layer at the output of the denoiser.

The non-expansiveness assumption (H.2) can be encouraged during training by introducing a regularization term that controls the spectral norm of the denoiser Jacobian (Pesquet et al. 2021). However, this strategy is computationally very expensive, and we simply chose to ignore this constraint in practice. A theoretical understanding of the training dynamics, including a potential implicit bias toward non-expansive denoisers, would be valuable. This avenue is beyond the scope of this work and is left for future research.

3.5.2. Noise-level-aware denoiser

According to the discussions in Sects. 3.2, 3.3 and 3.4, the denoiser $\mathcal{D}_{\hat{\boldsymbol{\theta}}}$ was trained by solving (23), where $\boldsymbol{n}_{0,i}$ are sampled from a zero-mean white Gaussian distribution with variance τ^2. The bound (29) shows that the noise level τ used for training depends on the noise covariance matrix $\tilde{\boldsymbol{\Sigma}}$ of the mass-mapping problem. At this stage, the training phase therefore remains dependent on $\tilde{\boldsymbol{\Sigma}}$. However, this dependence now reduces to a single scalar τ, whereas the full (diagonal) covariance matrix was required for training DeepMass.

We aim to push this one step forward by making the training phase completely independent of the covariance matrix. One simple strategy would be to select a low value for τ to cover a broad range of survey conditions. However, as our experiments confirm, the choice of τ strongly affects the reconstruction quality.

We therefore adopted a more flexible approach in which the model was enabled to adapt to different noise conditions during inference. To this end, the network was trained on a dataset $(\boldsymbol{\kappa}_{0,i},\, \boldsymbol{\kappa}_i)_{i=1}^n$, where each input $\boldsymbol{\kappa}_{0,i}$ was obtained by corrupting the ground truth $\boldsymbol{\kappa}_i$ by noise $\boldsymbol{n}_{0,i}$ drawn from $\mathcal{N}(\mathbf{0},\, \sigma_i^2)$, with σ_i being randomly selected among a wide range of noise levels. For higher reconstruction accuracy, we used a noise-level-aware model, following an approach initially proposed by Zhang et al. (2022). In this context, the denoiser network takes the noise standard deviation σ as an additional entry, that is, $\mathcal{D}_{\hat{\boldsymbol{\theta}}}(\boldsymbol{\kappa}',\, \sigma)$, where $\sigma > 0$ denotes the noise standard deviation at which the model is expected to operate.

During inference with PnPMass, according to (27), the intermediate outputs are propagated though the network with noise standard deviation τ. Then, the fixed-point iteration introduced in (14) now reads

$$\mathcal{H}_{\tilde{\boldsymbol{\gamma}}}(\cdot,\, \tau) := \mathcal{D}_{\hat{\boldsymbol{\theta}}}(\cdot,\, \tau) \circ \mathcal{F}_{\tilde{\boldsymbol{\gamma}}}(\cdot,\, \tau), \tag{31}$$

where now the forward and denoising steps both depend on the step size τ. This is again reminiscent of the forward-backward splitting (FBS) algorithm, as discussed in Appendix F.

3.5.3. Training on multiple cosmologies

Generating training data via simulation requires selecting a set of cosmological parameters, which implicitly shapes the prior distribution from which convergence map samples are drawn. If the chosen parameters diverge significantly from the true unknown cosmology that is to be inferred, the trained denoiser may struggle to generalize and thus fail to produce accurate reconstructions. One way to overcome this is to generate training data across a distribution of plausible cosmologies, enhancing the network robustness to distributional shifts.

3.6. PnPMass on non-Gaussian residuals

In the spirit of DeepMass (Jeffrey et al. 2020), the PnPMass algorithm can be enhanced by considering a priori knowledge about the underlying physics and data structure. To do this, we used a hypothesis made by Starck et al. (2021) for MCALens: convergence maps can be considered the superposition of a Gaussian and a non-Gaussian field (hereafter referred to as a residual field),

$$\boldsymbol{\kappa} = \boldsymbol{\kappa}^{\mathrm{g}} + \boldsymbol{\kappa}^{\mathrm{ng}}, \tag{32}$$

where $\boldsymbol{\kappa}^{\mathrm{g}}$, the Gaussian component, can be estimated from the noisy shear map $\boldsymbol{\gamma}$ using an iterative Wiener filtering method (see Sect. 2.2.1), assuming the power spectrum $\mathbf{P}_{\mathrm{g}}$ is known, coined $\mathcal{W}_{\mathbf{P}_{\mathrm{g}}}$,

$$\boldsymbol{\kappa}^{\mathrm{g}} := \mathcal{W}_{\mathbf{P}_{\mathrm{g}}}(\tilde{\boldsymbol{\gamma}}). \tag{33}$$

Then, the inverse problem (12) can be rewritten as

$$\tilde{\boldsymbol{\gamma}}^{\mathrm{ng}} = \tilde{\mathbf{A}}\boldsymbol{\kappa}^{\mathrm{ng}} + \tilde{\boldsymbol{n}}, \tag{34}$$

where $\tilde{\boldsymbol{\gamma}}^{\mathrm{ng}}$ denotes the residual shear map,

$$\tilde{\boldsymbol{\gamma}}^{\mathrm{ng}} := \tilde{\boldsymbol{\gamma}} - \tilde{\mathbf{A}}\boldsymbol{\kappa}^{\mathrm{g}}. \tag{35}$$

Then, PnPMass can be used to estimate $\boldsymbol{\kappa}^{\mathrm{ng}}$.

In order for this approach to remain consistent with the previous theoretical framework, the denoiser must be trained on a suitable dataset. Specifically, we considered the training set $(\boldsymbol{\kappa}_{0,i},\, \boldsymbol{\kappa}_i)_{i=1}^n$ introduced in Sect. 3.2. Then, for each $i \in \{0, \ldots, n\}$, we computed the Gaussian component $\boldsymbol{\kappa}_i^{\mathrm{g}}$ from the noisy input $\boldsymbol{\kappa}_{0,i}$ using a Wiener filter with a power spectrum $\mathbf{P}_{\mathrm{g}}$. Finally, we obtained the residual training set $(\boldsymbol{\kappa}_{0,i}^{\mathrm{ng}},\, \boldsymbol{\kappa}_i^{\mathrm{ng}})_{i=1}^n$ by subtracting $\boldsymbol{\kappa}_i^{\mathrm{g}}$ from the noisy input and the ground truth.

4. Uncertainty quantification

In Sect. 4.1 we introduced a fast UQ method for PnPMass. This approach is based on moment networks that were originally proposed by Jeffrey & Wandelt (2020) to quantify uncertainty in end-to-end deep-learning models such as DeepMass.

To the best of our knowledge, this is the first UQ method within the PnP framework that does not rely on ensembles or posterior sampling, both of which can be computationally demanding. Ensemble-based approaches estimate the uncertainty by training and evaluating multiple networks and by capturing variability across the model parameters (Shi et al. 2021; Terris et al. 2025). Posterior sampling methods, in contrast, are Bayesian UQ methods that seek to approximate the posterior distribution of the reconstructed images based on some implicit or explicit priors. For example, Laumont et al. (2022) combined implicit PnP priors with Langevin diffusion to draw posterior samples. Alternatively, Ekmekci & Cetin (2021) proposed a Bayesian extension of the PnP framework by placing a prior over the denoiser parameters and propagating the uncertainty through the reconstruction pipeline.

A sampling-free method was introduced by Postels et al. (2019) to approximate uncertainty in Monte Carlo dropout networks. This approach could in principle be adapted to the Bayesian PnP framework proposed by Ekmekci & Cetin (2021) as a way to reduce computational cost. This line of research targets uncertainties in the model design and parameters (commonly referred to as epistemic uncertainty) and is left for future work. In contrast, we assumed a perfect model design and training, and we instead quantified irreducible uncertainty arising from data noise, known as aleatoric uncertainty. We refer to Abdar et al. (2021) for a broader discussion of the distinction between epistemic and aleatoric uncertainty.

Next, in Sect. 4.2, we describe the CQR algorithm (Romano et al. 2019), which was adapted for mass mapping in a previous work (Leterme et al. 2025). This method serves as a postprocessing step that can be applied to any reconstruction method with an initial uncertainty estimate, providing per-pixel non-asymptotic marginal coverage guarantees without relying on prior assumptions about the data distribution.

4.1. Variance estimation with moment networks

In Sect. 4.1.1 we adapt the main principles of posterior variance estimation using moment networks, as described by Jeffrey & Wandelt (2020), to DeepMass specifically. Then, in Sect. 4.1.2, we extend this approach to the PnP framework and provide theoretical insights supporting its use. Finally, in Sect. 4.1.3, we explain how the error bars are obtained from the estimated posterior variance.

4.1.1. Moment networks for DeepMass

Given a simulated dataset $(\tilde{\boldsymbol{\gamma}}_i, \boldsymbol{\kappa}_i)_{i=1}^n$, where the pairs $(\tilde{\boldsymbol{\gamma}}_i, \boldsymbol{\kappa}_i)$ are independent and identically-distributed samples of the ground-truth convergence maps and observed noisy shear maps, DeepMass attempts to solve the mass-mapping problem by learning a deep neural network $\mathcal{M}_{\boldsymbol{w}}$ on $(\tilde{\boldsymbol{\gamma}}_i, \boldsymbol{\kappa}_i)_{i=1}^n$ according to

$$\hat{\boldsymbol{w}} \in \operatorname*{argmin}_{\boldsymbol{w}} \frac{1}{n} \sum_{i=1}^{n} \left\| \mathcal{M}_{\boldsymbol{w}}(\tilde{\boldsymbol{\gamma}}_i) - \boldsymbol{\kappa}_i \right\|^2. \tag{36}$$

As $n \to \infty$, (36) is expected to approach a minimizer of the population risk, that is,

$$\hat{\boldsymbol{w}} \in \operatorname*{argmin}_{\boldsymbol{w}} \mathbb{E}\left[\left\| \mathcal{M}_{\boldsymbol{w}}(\tilde{\boldsymbol{\Gamma}}) - \mathbf{K} \right\|^2 \right], \tag{37}$$

where the expectation is to be understood with respect to the joint distribution of $(\mathbf{K}, \tilde{\boldsymbol{\Gamma}})$.

DeepMass adopts a Bayesian perspective and attempts to interpret (37) as the solution to

$$\min_{\mathcal{D} \text{ is } \mu_{\tilde{\boldsymbol{\Gamma}}|\mathbf{K}}\text{-measurable}} \mathbb{E}\left[\left\| \mathcal{D}(\tilde{\boldsymbol{\Gamma}}) - \mathbf{K} \right\|^2 \right], \tag{38}$$

where $\mu_{\tilde{\boldsymbol{\Gamma}}|\mathbf{K}}$ is the conditional distribution of $\tilde{\boldsymbol{\Gamma}}$ given $\mathbf{K}$. The closed-form solution to (38) is known as the posterior conditional mean estimator,

$$\widehat{\mathcal{D}}(\tilde{\boldsymbol{\gamma}}) = \mathbb{E}[\mathbf{K} \,|\, \tilde{\boldsymbol{\Gamma}} = \tilde{\boldsymbol{\gamma}}]. \tag{39}$$

The key assumption in DeepMass is that the following approximation holds:

$$\widehat{\mathcal{D}}(\tilde{\boldsymbol{\gamma}}) \approx \mathcal{M}_{\hat{\boldsymbol{w}}}(\tilde{\boldsymbol{\gamma}}). \tag{40}$$

Intuitively, this can make sense if the architecture of the neural network $\mathcal{M}_{\boldsymbol{w}}$, seen as a parameterized class of functions, is expressive enough, in such a way that the two optimization problems (37) and (38) are close.

This reasoning can be extended to estimate higher-order moments, including the posterior conditional variance per pixel,

$$\mathbb{V}[\mathbf{K} \,|\, \tilde{\boldsymbol{\Gamma}} = \tilde{\boldsymbol{\gamma}}] := \mathbb{E}\left[\left(\mathbf{K} - \mathbb{E}[\mathbf{K} \,|\, \tilde{\boldsymbol{\Gamma}} = \tilde{\boldsymbol{\gamma}}] \right)^2 \,\middle|\, \tilde{\boldsymbol{\Gamma}} = \tilde{\boldsymbol{\gamma}} \right], \tag{41}$$

which is known to be the solution to

$$\min_{\mathcal{V} \text{ is } \mu_{\tilde{\boldsymbol{\Gamma}}|\mathbf{K}}\text{-measurable}} \mathbb{E}\left[\left\| \mathcal{V}(\tilde{\boldsymbol{\Gamma}}) - (\mathbf{K} - \widehat{\mathcal{D}}(\tilde{\boldsymbol{\Gamma}})^2 \right\|^2 \right]. \tag{42}$$

To this end, we considered a new neural network model $\mathcal{G}_{\boldsymbol{\omega}}$ trained on a dataset $(\tilde{\boldsymbol{\gamma}}_i, \boldsymbol{\nu}_i)_{i=1}^n$ by solving

$$\hat{\boldsymbol{\omega}} \in \operatorname*{argmin}_{\boldsymbol{\omega}} \frac{1}{n} \sum_{i=1}^{n} \left\| \mathcal{G}_{\boldsymbol{\omega}}(\tilde{\boldsymbol{\gamma}}_i) - \boldsymbol{\nu}_i \right\|^2, \tag{43}$$

where we defined the target $\boldsymbol{\nu}_i := (\boldsymbol{\kappa}_i - \mathcal{M}_{\hat{\boldsymbol{w}}}(\tilde{\boldsymbol{\gamma}}_i))^2$. Therefore, similarly to what we have argued for (37), (43) is an empirical version of

$$\hat{\boldsymbol{\omega}} \in \operatorname*{argmin}_{\boldsymbol{\omega}} \mathbb{E}\left[\left\| \mathcal{G}_{\boldsymbol{\omega}}(\tilde{\boldsymbol{\Gamma}}) - \left(\mathbf{K} - \mathcal{M}_{\hat{\boldsymbol{w}}}(\tilde{\boldsymbol{\Gamma}}) \right)^2 \right\|^2 \right], \tag{44}$$

Comparing (42) and (44), combining (40) and (41), and arguing as above, we can reasonably assume that

$$\mathbb{V}[\mathbf{K} \,|\, \tilde{\boldsymbol{\Gamma}} = \tilde{\boldsymbol{\gamma}}] \approx \mathcal{G}_{\hat{\boldsymbol{\omega}}}(\tilde{\boldsymbol{\gamma}}). \tag{45}$$

The corresponding model is referred to as an order-2 moment network.

4.1.2. Adaptation to PnPMass

The moment network approach is not directly applicable to the PnP framework for several reasons. First, PnP, which is iterative, does not necessarily have a Bayesian interpretation, let alone a posterior conditional mean (see Sect. 3.3 and Appendix F). We also recall that we did not wish to train any network on a dataset that depends on the noise covariance matrix $\tilde{\boldsymbol{\Sigma}}$ for flexibility reasons. However, we describe below that within the setting presented in Sect. 3, the posterior conditional variance can be estimated with an order-2 moment network trained on zero-mean white Gaussian noise.

Let $\mathcal{D}_{\hat{\theta}}$ be the denoiser network that has been trained by solving (23) on a training dataset $(\boldsymbol{\kappa}_{0,i}, \boldsymbol{\kappa}_i)_{i=1}^n$, where we have denoted $\boldsymbol{\kappa}_{0,i} := \boldsymbol{\kappa}_i + \boldsymbol{n}_{0,i}$, with $\boldsymbol{n}_{0,i}$ being sampled from $\mathcal{N}(\mathbf{0}, \tau_i^2 \mathbf{I})$ conditionally on τ_i (see (22) and (27)). The noise standard deviations τ_i are samples from the uniform distribution on an interval $[\tau_{\min}, \tau_{\max}]$ containing the bounds in (28). As discussed in Sect. 3.5.2, τ is also an input of the denoiser network.

We now consider the random vector

$$\mathbf{K}_0 := \mathbf{K} + \mathbf{N}_0, \tag{46}$$

where $\mathbf{N}_0$ has been introduced in (22). In the context in which we consider the step size τ to be an outcome of $\mathsf{T} \sim \mathcal{U}([\tau_{\min}, \tau_{\max}])$, $\mathbf{N}_0$ is now a mixture of zero-mean Gaussians with uniform mixing over T. The noisy inputs $\boldsymbol{\kappa}_{0,i}$ from the training set are then assumed to be drawn independently from the same distribution as $\mathbf{K}_0$. With a similar reasoning as in Sect. 4.1.1, we can infer that given τ and a noisy input $\boldsymbol{\kappa}_0$,

$$\mathcal{D}_{\hat{\theta}}(\boldsymbol{\kappa}_0, \tau) \approx \mathbb{E}\Big[\mathbf{K} \,\Big|\, \mathbf{K}_0 = \boldsymbol{\kappa}_0, \mathsf{T} = \tau\Big]. \tag{47}$$

For the posterior conditional variance, we consider a model $\mathcal{G}_{\hat{\omega}}$ trained on $(\boldsymbol{\kappa}_{0,i}, \boldsymbol{v}_i)_{i=1}^n$, with

$$\boldsymbol{v}_i := \left(\boldsymbol{\kappa}_i - \mathcal{D}_{\hat{\theta}}(\boldsymbol{\kappa}_{0,i}, \tau_i)\right)^2, \tag{48}$$

and again following Sect. 4.1.1, we obtain

$$\mathcal{G}_{\hat{\omega}}(\boldsymbol{\kappa}_0, \tau) = \mathbb{V}\Big[\mathbf{K} \,\Big|\, \mathbf{K}_0 = \boldsymbol{\kappa}_0, \mathsf{T} = \tau\Big]. \tag{49}$$

Now, we argue that after reaching a sufficient number of iterations, the outputs of $\mathcal{D}_{\hat{\theta}}$ and $\mathcal{G}_{\hat{\omega}}$ approximate the posterior conditional mean and variance, respectively, given a noisy observation $\tilde{\boldsymbol{\Gamma}} = \tilde{\boldsymbol{\gamma}}$. To this end, let $\hat{\boldsymbol{\kappa}}_0 := \mathcal{F}_{\tilde{\boldsymbol{\gamma}}}(\hat{\boldsymbol{\kappa}}, \tau)$ denote the output of the forward step applied to the fixed point, satisfying (19). In Sect. 3.1 we informally established conditions on the training noise to favor small residuals $\hat{\boldsymbol{r}}$. Therefore, we consider (20) as a reasonable approximation,

$$\hat{\boldsymbol{\kappa}}_0 \approx \boldsymbol{\kappa} + \boldsymbol{n}_0, \quad \text{with} \quad \boldsymbol{n}_0 := \tau \tilde{\mathbf{B}} \tilde{\boldsymbol{n}} \in \mathbb{R}^{K^2}. \tag{50}$$

For a number of iterations $N_{\text{it}} \geq 1$, denote $\mathcal{H}_{\tilde{\boldsymbol{\gamma}}}^{N_{\text{it}}}(\cdot, \tau)$ the N_{it}-th composition of the PnPMass fixed-point operator $\mathcal{H}_{\tilde{\boldsymbol{\gamma}}}^{N_{\text{it}}}(\cdot, \tau)$. Clearly, the PnPMass iteration (17) also reads

$$\boldsymbol{\kappa}^{(N_{\text{it}})} = \mathcal{H}_{\tilde{\boldsymbol{\gamma}}}^{N_{\text{it}}}(\boldsymbol{\kappa}^{(0)}, \tau). \tag{51}$$

We denote $\mathcal{P}(\tilde{\boldsymbol{\gamma}}, \tau) := \mathcal{F}_{\tilde{\boldsymbol{\gamma}}}(\boldsymbol{\kappa}^{(N_{\text{it}})}, \tau)$ the output of the $(N_{\text{it}} + 1)$-th forward step. For N_{it} large enough, we have from Proposition 1 that $\boldsymbol{\kappa}^{(N_{\text{it}})} \approx \hat{\boldsymbol{\kappa}}$, which combined with (19) and (20) yields

$$\mathcal{P}(\tilde{\boldsymbol{\gamma}}, \tau) \approx \mathcal{F}_{\tilde{\boldsymbol{\gamma}}}(\hat{\boldsymbol{\kappa}}, \tau) = \hat{\boldsymbol{\kappa}}_0 \approx \boldsymbol{\kappa} + \boldsymbol{n}_0, \tag{52}$$

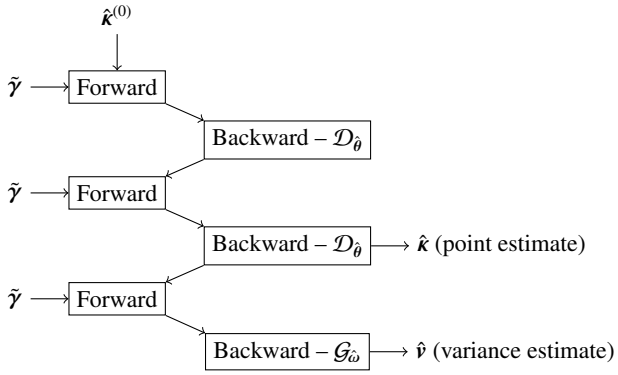


Fig. 1: Schematic representation of the PnPMass algorithm. Two iterations are shown for the point estimate (for illustration purposes only; more iterations are typically required in practice), followed by one additional iteration for the variance.

and therefore,

$$\mathcal{P}(\tilde{\boldsymbol{\Gamma}}, \mathsf{T}) \approx \mathbf{K}_0, \tag{53}$$

where $\mathbf{K}_0$ has been introduced in (46). Plugging this into (47) and (49), replacing $\boldsymbol{\kappa}_0$ by $\hat{\boldsymbol{\kappa}}_0$, we obtain

$$\mathcal{D}_{\hat{\theta}}(\hat{\boldsymbol{\kappa}}_0, \tau) \approx \mathbb{E}\Big[\mathbf{K} \,\Big|\, \mathcal{P}(\tilde{\boldsymbol{\Gamma}}, \mathsf{T}) = \mathcal{P}(\tilde{\boldsymbol{\gamma}}, \tau), \mathsf{T} = \tau\Big] \tag{54}$$

and

$$\mathcal{G}_{\hat{\omega}}(\hat{\boldsymbol{\kappa}}_0, \tau) \approx \mathbb{V}\Big[\mathbf{K} \,\Big|\, \mathcal{P}(\tilde{\boldsymbol{\Gamma}}, \mathsf{T}) = \mathcal{P}(\tilde{\boldsymbol{\gamma}}, \tau), \mathsf{T} = \tau\Big]. \tag{55}$$

It is then sufficient that $\mathcal{P}(\cdot, \tau)$ is injective to deduce that

$$\mathcal{D}_{\hat{\theta}}(\hat{\boldsymbol{\kappa}}_0, \tau) \approx \mathbb{E}\Big[\mathbf{K} \,\Big|\, \tilde{\boldsymbol{\Gamma}} = \tilde{\boldsymbol{\gamma}}\Big] \tag{56}$$

and

$$\mathcal{G}_{\hat{\omega}}(\hat{\boldsymbol{\kappa}}_0, \tau) \approx \mathbb{V}\Big[\mathbf{K} \,\Big|\, \tilde{\boldsymbol{\Gamma}} = \tilde{\boldsymbol{\gamma}}\Big]. \tag{57}$$

The overall algorithm implementing the PnPMass iteration is summarized in Fig. 1 and Algorithm 1.

4.1.3. Uncalibrated pre-estimation of per-pixel error bars

Our goal is to provide per-pixel error bars with marginal and non-asymptotic coverage guarantees. More precisely, given a desired miscoverage rate $\alpha \in]0, 1[$, we wish to estimate $\hat{\boldsymbol{\kappa}}^-$ and $\hat{\boldsymbol{\kappa}}^+$ such that for every $k \in \{1, \ldots, K^2\}$,

$$\mathbb{P}\Big\{\mathbf{K}[k] \notin [\hat{\boldsymbol{\kappa}}^-[k], \hat{\boldsymbol{\kappa}}^+[k]] \,\Big|\, \tilde{\boldsymbol{\Gamma}} = \tilde{\boldsymbol{\gamma}}\Big\} \leq \alpha. \tag{58}$$

To this end, we started with a rough estimate based on a Gaussian assumption. This was then calibrated (see Sect. 4.2). More precisely, we started with the following quantiles $\hat{\boldsymbol{\kappa}}^-$ and $\hat{\boldsymbol{\kappa}}^+$ from a Gaussian distribution of mean $\hat{\boldsymbol{\kappa}}$ and variance $\hat{\boldsymbol{v}}$, which are provided by Algorithm 1:

$$\hat{\boldsymbol{\kappa}}^{\pm}[k] := \hat{\boldsymbol{\kappa}}[k] \pm \Phi^{-1}\left(1 - \frac{\alpha}{2}\right) \hat{\boldsymbol{\sigma}}[k], \tag{59}$$

where Φ is the cumulative distribution function (CDF) of the normal distribution $\mathcal{N}(0, 1)$, and $\hat{\boldsymbol{\sigma}} = \sqrt{\hat{\boldsymbol{v}}}$ is the standard deviation estimate. If the conditional distribution of $\mathbf{K}[k]$, given $\tilde{\boldsymbol{\gamma}}$, were $\mathcal{N}(\hat{\boldsymbol{\kappa}}[k], \hat{\boldsymbol{v}}[k])$, then (58) would be satisfied by construction. This is not the case, however, and hence the need of a calibration step that we describe below.

Algorithm 1 PnPMass with uncertainty quantification

Require: Noise covariance matrix $\boldsymbol{\Sigma}$;
Require: Power spectrum $\mathbf{P}_{\mathrm{g}}$ (residual version only, Sect. 3.6);
Require: Step-size τ and number of iterations N_{it};
Require: Denoiser $\mathcal{D}_{\hat{\theta}}$ trained on white Gaussian noise;
Require: Order-2 moment network $\mathcal{G}_{\hat{\omega}}$.

input: Noisy shear map $\tilde{\boldsymbol{\gamma}}$.
if residual version **then**
  Compute Gaussian component: $\boldsymbol{\kappa}^{\mathrm{g}} \leftarrow \mathcal{W}_{\mathbf{P}_{\mathrm{g}}}(\tilde{\boldsymbol{\gamma}})$;
  Update input (residual): $\tilde{\boldsymbol{\gamma}} \leftarrow \tilde{\boldsymbol{\gamma}} - \tilde{\mathbf{A}}\boldsymbol{\kappa}^{\mathrm{g}}$.
end if

initialize: $\boldsymbol{\kappa}^{(0)} = \mathbf{0}$.
for $i = 1, \dots, N_{\mathrm{it}}$ **do**
  Forward step: $\boldsymbol{\kappa}_0^{(i)} \leftarrow \mathcal{F}_{\tilde{\boldsymbol{\gamma}}}(\boldsymbol{\kappa}^{(i-1)}, \tau)$;
  Backward step: $\boldsymbol{\kappa}^{(i)} \leftarrow \mathcal{D}_{\hat{\theta}}(\boldsymbol{\kappa}_0^{(i)}, \tau)$.
end for
Set point estimate: $\hat{\boldsymbol{\kappa}} \leftarrow \boldsymbol{\kappa}^{(N_{\mathrm{it}})}$.

Forward step: $\hat{\boldsymbol{\kappa}}_0 \leftarrow \mathcal{F}_{\tilde{\boldsymbol{\gamma}}}(\hat{\boldsymbol{\kappa}}, \tau)$;
Backward step (variance estimate): $\hat{\boldsymbol{\nu}} \leftarrow \mathcal{G}_{\hat{\omega}}(\hat{\boldsymbol{\kappa}}_0, \tau)$.

if residual version **then**
  Update point estimate: $\hat{\boldsymbol{\kappa}} \leftarrow \hat{\boldsymbol{\kappa}} + \boldsymbol{\kappa}^{\mathrm{g}}$.
end if
return $(\hat{\boldsymbol{\kappa}}, \hat{\boldsymbol{\nu}})$.

4.2. Calibration with conformal predictions

In Sect. 4.1 the pre-estimates of the per-pixel error bars were based on several assumptions entailing several approximations, as listed below.

- (47) and (49) state that the networks $\mathcal{D}_{\hat{\theta}}$ and $\mathcal{G}_{\hat{\omega}}$ are only approximations of the posterior mean and variance given $\mathbf{K}_0 = \boldsymbol{\kappa}_0$, respectively. As discussed there, the approximation quality depends on factors such as the complexity of the neural networks used and the quality and diversity of the training data. The method therefore primarily captures aleatoric uncertainty, while epistemic uncertainty, associated with model expressiveness or limited knowledge, remains unaddressed.
- (52) is yet another approximation of the point estimate which, as we argued there, relies on the assumption that the fixed point $\hat{\boldsymbol{\kappa}}$ of PnPMass lies close enough to $\boldsymbol{\kappa}$. In practice, this is not necessarily true, and thus bias, that is, nonzero residuals in (52), may still be present. This is especially true near masked regions where the S/N is very low.
- The per-pixel error bars have been derived under a Gaussian assumption for the posterior distribution, making them only rough uncertainty estimates.

For all these reasons, the miscoverage inequality as stated in (58) is not guaranteed to hold. In this section, we present a calibration method directly inspired by CQR (Romano et al. 2019), for which we can obtain (marginal) coverage guarantees that are valid for finite samples (non-asymptotic) and do not depend on any assumption on the models or the data distribution (distribution free). The choice of this method over other calibration approaches such as risk-controlling prediction sets (Angelopoulos et al. 2022), based on concentration inequalities, is motivated by the fact that it explicitly avoids overconservative error bands, making it suitable in a context in which the tightest possible error bars are required. We refer to Leterme et al. (2025) for a detailed discussion of this topic.

In Sect. 4.2.1 we review the general principles of CQR for mass mapping. Then, we propose in Sect. 4.2.2 a method for minimizing the size of the error bars without the need for a model selection on a validation set.

4.2.1. General principles

Let $(\tilde{\boldsymbol{\gamma}}_i, \boldsymbol{\kappa}_i)_{i=n+1}^{n+m}$ denote a calibration set of size $m \in \mathbb{N}$. For each pair $(\tilde{\boldsymbol{\gamma}}_i, \boldsymbol{\kappa}_i)$, we denote by $(\hat{\boldsymbol{\kappa}}_i^-, \hat{\boldsymbol{\kappa}}_i^+)$ the lower and upper bounds obtained using (59), from which a vector of conformity scores, denoted by $\boldsymbol{\lambda}_i \in \mathbb{R}^{K^2}$, can be computed as follows. For each pixel $k \in \{1, \dots, K^2\}$,

$$\boldsymbol{\lambda}_i[k] = \max\left(\hat{\boldsymbol{\kappa}}_i^-[k] - \boldsymbol{\kappa}_i[k],\, \boldsymbol{\kappa}_i[k] - \hat{\boldsymbol{\kappa}}_i^+[k]\right). \tag{60}$$

Then, we introduce the calibration vector $\boldsymbol{\lambda}$, obtained by taking, in each pixel k, the $(1-\alpha)(1+1/m)$-th empirical quantile of $(\boldsymbol{\lambda}_i[k])_{i=n+1}^{n+m}$. The term $(1+1/m)$, which accounts for finite-sample correction, imposes a lower bound to the target error rate: $\alpha \geq 1/(m+1)$. The calibrated lower and upper bounds are finally defined as

$$\hat{\boldsymbol{\kappa}}_{\mathrm{cqr}}^- := \hat{\boldsymbol{\kappa}}^- - \boldsymbol{\lambda} \qquad \text{and} \qquad \hat{\boldsymbol{\kappa}}_{\mathrm{cqr}}^+ := \hat{\boldsymbol{\kappa}}^+ + \boldsymbol{\lambda}. \tag{61}$$

We denote by $\hat{\mathbf{K}}_{\mathrm{cqr}}^-$ and $\hat{\mathbf{K}}_{\mathrm{cqr}}^+$ the random vectors from which $\hat{\boldsymbol{\kappa}}_{\mathrm{cqr}}^-$ and $\hat{\boldsymbol{\kappa}}_{\mathrm{cqr}}^+$ are drawn. Their random nature comes from that of $\tilde{\boldsymbol{\Gamma}}$, from which the noisy observation is drawn, as well as from $(\tilde{\boldsymbol{\Gamma}}_i, \mathbf{K}_i)_{i=n+1}^{n+m}$, from which the calibration set is drawn. If $(\tilde{\boldsymbol{\Gamma}}_i, \mathbf{K}_i)_{i=n+1}^{n+m} \cup \{(\tilde{\boldsymbol{\Gamma}}, \mathbf{K})\}$ are exchangeable (e.g., independent and identically distributed), and the conformity scores are almost surely distinct, then we have the following (marginal) miscoverage guarantee at pixel k:

$$\alpha - \frac{1}{m+1} \leq \mathbb{P}\left\{\mathbf{K}[k] \notin \left[\hat{\mathbf{K}}_{\mathrm{cqr}}^-[k],\, \hat{\mathbf{K}}_{\mathrm{cqr}}^+[k]\right]\right\} \leq \alpha. \tag{62}$$

By averaging over pixels, we deduce that the expected proportion of pixels whose mass estimates fall outside the corresponding predicted error interval also verifies the bounds in (62).

The bound in (62) presents a major difference compared to (58). Whereas the latter is conditional given $\tilde{\boldsymbol{\gamma}}$, the former is marginalized over all possible outcomes of $\tilde{\boldsymbol{\Gamma}}$. Consequently, some inputs may more likely lead to errors than others. Conformal prediction with conditional guarantees is a very challenging problem and has only been solved in specific cases (Gibbs et al. 2025). We leave this for future work. In addition, the coverage guarantee provided in (62) is also marginalized over the distribution of all calibration sets. Therefore, accidentally picking an out-of-distribution calibration set may lead to a higher error rate than expected. This effect can be mitigated by increasing the size of the calibration set.

The calibration procedure is not independent of the covariance matrix $\boldsymbol{\Sigma}$, unlike the training phase. However, we point out that the former is cheaper by several orders of magnitude than the latter, computationally speaking, as detailed in Sect. 6.4.

4.2.2. Minimizing the size of the error bars

We can easily show that the miscoverage guarantee from (62) still holds if the pre-calibration bounds $\hat{\boldsymbol{\kappa}}^{\pm}$, introduced in (59), are replaced by

$$\hat{\boldsymbol{\kappa}}_{\mu}^{\pm}[k] := \hat{\boldsymbol{\kappa}}[k] \pm \mu\, \Phi^{-1}\left(1 - \frac{\alpha}{2}\right) \hat{\boldsymbol{\sigma}}[k] \tag{63}$$

for any multiplicative constant $\mu > 0$. The calibration vector is then recomputed for each μ, and is denoted by $\boldsymbol{\lambda}_\mu$. The mean error bar size can then be minimized over μ:

$$\mu^* \in \operatorname*{argmin}_{\mu>0} \mu\epsilon + \lambda_\mu, \tag{64}$$

where $\epsilon > 0$ denotes the mean half-size of the precalibration error bars (averaged over all test images and all pixels), and $\lambda_\mu \in \mathbb{R}$ denotes the mean value of the calibration vector $\boldsymbol{\lambda}_\mu$, which is a non-increasing function of μ. This procedure is computationally inexpensive because $\hat{\boldsymbol{\kappa}}_i$ and $\hat{\boldsymbol{\sigma}}_i$ are computed only once. For each candidate μ, only the conformity scores $\boldsymbol{\lambda}_{\mu,i}$ and their corresponding quantiles must be recomputed in order to obtain $\boldsymbol{\lambda}_\mu$.

5. Experimental settings

5.1. Datasets

In order to generate the training, validation, calibration, and test sets, the ground-truth convergence maps $\boldsymbol{\kappa}_i$ were obtained using the κTNG dataset of mock convergence maps, as described below. The mass maps were generated from the full redshift distribution, following the same procedure as Leterme et al. (2025).

For the training and validation sets, the noisy inputs $\boldsymbol{\kappa}_{0,i}$ were obtained from $\boldsymbol{\kappa}_i$ by adding a white Gaussian noise with a standard deviation uniformly distributed between 0 and 0.2. The choice for this particular interval is explained in Sect. 5.2.1. At each training epoch, a new noise realization was drawn, with a newly selected noise level. For the calibration and test sets, the noisy shear maps $\boldsymbol{\gamma}_i$ were computed from $\boldsymbol{\kappa}_i$ using (7). The covariance matrix $\boldsymbol{\Sigma}$ was obtained by binning the weak-lensing shear catalog from Schrabback et al. (2010) (bright catalog only) at the κTNG resolution (0.29 arcmin per pixel), and then applying (6). The intrinsic standard deviation σ_{ell} was estimated from the measured ellipticities and set to 0.39.

This catalog is based on the Cosmic Evolution Survey (COSMOS, Scoville et al. 2007) obtained with the Hubble Space Telescope, as well as photometric redshift measurements from Mobasher et al. (2007). It contains an average of 32 galaxies per square arcminute over a wide range of redshifts, which is consistent with what is expected from forthcoming surveys such as Euclid. This number excludes the galaxies with redshifts above 2.6 for the sake of consistency with the κTNG dataset. It also ignores the masked pixels, without any observed galaxy. An example of a mock convergence map $\boldsymbol{\kappa}_i$ from κTNG, and the corresponding noisy shear map $\boldsymbol{\gamma}_i$, is provided in Appendix H, Fig. H.1.

The κTNG cosmological hydrodynamic simulations (Osato et al. 2021) include realizations of $5 \times 5\,\mathrm{deg}^2$ convergence maps for various source redshifts up to $z = 2.6$, at a 0.29 arcmin per pixel resolution, assuming a flat ΛCDM universe. Only one set of cosmological parameters was used to run the simulations: $H_0 = 67.74\,\mathrm{km \cdot s^{-1} \cdot Mpc^{-1}}$, $\Omega_\mathrm{b} = 0.0486$, $\Omega_\mathrm{m} = 0.3089$, $n_\mathrm{s} = 0.9667$, and $\sigma_8 = 0.8159$. For training and validation, we used 98 and 2 nearly independent realizations generated from a single lensing potential, respectively, that we randomly rotated and cropped to 384×384 pixels until we reached 70 560 images for training and 1 440 for validation. For calibration and testing, we used 57 and 43 nearly independent realizations generated from another lensing potential, respectively. The test set was then obtained by extracting 512 non-overlapping[1] crops of size 384×384 pixels, whereas the calibration set was obtained by following the augmentation procedure described above, with a total of 1 024 images.

[1] Accounting for the COSMOS boundaries.

5.2. Other implementation details

5.2.1. Parameters for PnPMass

With the specific value of $\boldsymbol{\Sigma}$ introduced in Sect. 5.1, the step size τ must be chosen in the interval]0, 0.176[to comply with (28). To derive these bounds, $\lambda_{\max}$ and $\lambda_{\min}$ were computed using the power iteration method with 100 iterations (see Appendix E). To assess the sensitivity of PnPMass to the step size, we tested several values of τ, set to 50%, 75%, and 100% of $\tau_{\max} := (1 + \rho)/\lambda_{\max}$, where ρ was taken arbitrarily close to 1.

According to (27), the denoiser must be able to operate at a noise level equal to the step size τ. For flexibility reasons, we chose to train the model on a realistic range of noise levels including the above interval, following the method described in Sect. 3.5.2. As detailed in Sect. 5.1, we used a uniform distribution between 0 and 0.2.

For the PnPMass variant on non-Gaussian residuals described in Sect. 3.6, the power spectrum $\mathbf{P}_\mathrm{g}$ was estimated on 2 048 ground-truth convergence maps from the training set. The Gaussian component $\boldsymbol{\kappa}^\mathrm{g}$ was then computed using an iterative Wiener filtering method with 12 iterations and a step-size of 1.61×10^{-2}.

5.2.2. Uncertainty quantification parameters

We selected a target error rate α such that $\Phi^{-1}(1 - \alpha/2) = 2$, see (59). This setting, referred to as 2σ confidence, corresponds to $\alpha \approx 4.55\%$. We note that higher confidence levels require larger calibration sets due to the finite-sample correction: the CQR algorithm selects the $(1 - \alpha)(1 + 1/m)$-th empirical quantile of the conformity scores, which by definition must remain below 100%. Consequently, a 2σ confidence requires at least $m = 21$ calibration samples, versus $m = 370$ at 3σ and $m = 15\,787$ at 4σ. In our experiments, we chose 1 024 calibration examples, which is far above the minimum requirement of 21 samples. According to (62), the probability of miscoverage after calibration with this setting ranges between 4.45% and 4.55%.

5.2.3. Denoiser training and PnPMass implementation

Our models are based on SUNet (Fan et al. 2022), a Swin transformer-based architecture with 7.2 millions of trainable parameters, which we adapted to incorporate noise-level awareness. Following an approach similar to that used by (Zhang et al. 2022) for DRUNet, we introduced an additional input channel that encodes the noise standard deviation. Since our models were trained with stationary noise, this channel was filled with the constant value σ_i for each input $\boldsymbol{\kappa}_{0,i}$.

We trained four models in total: two order-1 models $\mathcal{D}_{\hat{\boldsymbol{\theta}}}$ for the point estimate, and two order-2 models $\mathcal{G}_{\hat{\boldsymbol{\omega}}}$ for the variance (see Algorithm 1). In each case, we trained one model for standard PnPMass and another one for the variant on non-Gaussian residuals (see Sect. 3.6), using Adam as the optimizer and the squared ℓ^2 loss. Training was run for 100 epochs, with a batch size of 16. The learning rate was initialized at 10^{-3} and reduced by a factor of 10 four times during training. For the PnPMass variant on non-Gaussian residuals, the Gaussian components $\boldsymbol{\kappa}_i^\mathrm{g}$ were computed from $\boldsymbol{\kappa}_{0,i}$ using one-step Wiener filtering, which is feasible because the noise is stationary, with the power spectrum $\mathbf{P}_\mathrm{g}$ introduced in Sect. 5.2.1.

In $\mathcal{G}_{\hat{\boldsymbol{\omega}}}$, the final mean-centering layer was replaced with a rectified linear unit (ReLU) activation layer to enforce nonnegative outputs. To avoid vanishing gradients during training, however,

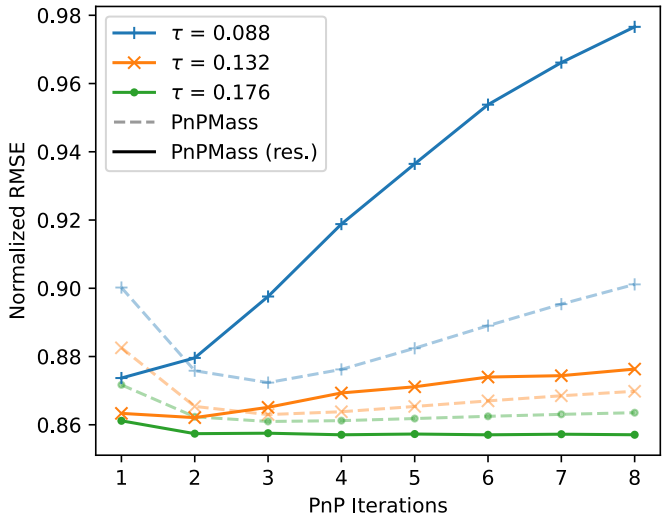


Fig. 2: Evolution of the mean RMSE along PnPMass iterations, computed on the κTNG test set (512 images) for various step sizes τ. The results have been normalized w.r.t. to the ℓ^2-norm of each image.

this layer was only applied at inference time. The training set was built from the same dataset as we used for training $\mathcal{D}_{\hat{\theta}}$, with the ground truth derived from (48). Our PnPMass implementation is based on the DeepInverse library (Tachella et al. 2025),[2] which is built on PyTorch.

6. Results and discussion

6.1. Reconstruction accuracy

6.1.1. PnPMass iterations

We ran PnPMass for eight iterations and plotted the evolution of the normalized root-mean-square error (RMSE) over the test set, restricted to active pixels, for various step sizes τ for the standard PnPMass and its variant on non-Gaussian residuals. The results are shown in Fig. 2. Throughout the discussion, we refer to this metric as the reconstruction accuracy.

The RMSE decreases over the first iterations before either stabilizing or increasing again, depending on the step size. Moreover, the reconstruction accuracy of PnPMass and its variant on non-Gaussian residuals increases as τ increases in the devised range. PnPMass on non-Gaussian residuals is notably more sensitive to the choice of τ than standard PnPMass. However, since the former incorporates prior knowledge of the Gaussian structure of the background mass distribution, it achieves the best reconstruction accuracy compared to the standard PnPMass at $\tau = \tau_{\max}$. Based on bootstrapping, the expected improvement after eight iterations lies between 6.04×10^{-3} and 6.88×10^{-3} with 95% confidence.

From now on, we focus on the results obtained after eight iterations, at $\tau = \tau_{\max}$. This choice was motivated by empirical observations because the reconstruction accuracy did not improve further beyond a few iterations.

[2] https://deepinv.github.io/

Table 2: Reconstruction accuracy.

Method	Normalized RMSE ($\times 10^{-1}$)
Wiener*	8.86 ± 0.34
MCALens†	8.74 ± 0.44
DeepMass‡	8.53 ± 0.48
PnPMass	8.64 ± 0.50
PnPMass (res.)	8.58 ± 0.49

Notes. For each test example, the RMSE was computed over active pixels and normalized with respect to the ℓ^2-norm of the ground-truth image. Then, the mean and standard deviation were evaluated across the test set (512 images). *Wiener was run for 12 iterations with a power spectrum estimated from 2 048 ground-truth κ-maps in the training set. †MCALens was run for 16 iterations (early stopping), with a 4.5σ detection threshold and a step size of 0.088 for the non-Gaussian component. ‡DeepMass was trained for 20 epochs on a UNet architecture with 280k parameters, following Jeffrey et al. (2020).

6.1.2. Benchmark against other approaches

We compared the performance of the Wiener, MCALens, DeepMass, and PnPMass algorithms in terms of reconstruction accuracy (RMSE on active pixels). The results are displayed in Table 2.

We found that our methods outperform the classical approaches (Wiener and MCALens), but fall slightly behind the reconstruction accuracy achieved by DeepMass. A likely explanation is that DeepMass has been fine-tuned for a specific mask $\mathbf{M}$ and noise covariance $\mathbf{\Sigma}$, whereas PnPMass is applicable across a wide range of configurations. However, as discussed in Sect. 2.2.2, this level of performance comes at the cost of retraining for each new observation, which limits its practicality. We therefore argue that PnPMass offers a very good trade-off between flexibility and reconstruction accuracy.

6.2. Uncertainty quantification results

For each mass-mapping method introduced above and each example in the κTNG test set, we computed the lower and upper bounds before and after calibration. Then, we measured the empirical miscoverage rate, that is, the percentage of active pixels falling outside the predicted bounds. For Wiener and MCALens, the error bars were simply initialized to 0 before calibration. Regarding DeepMass and PnPMass, the error bars before calibration were obtained using appropriate order-2 moment networks, as described in Sects. 4.1.1 and 4.1.2, respectively. We then applied calibration while minimizing the size of the error bars, as described in Sect. 4.2.2. The corresponding results, computed on the test set, are depicted in Figs. 3 and G.1a (Appendix G).

6.2.1. Accuracy versus error bar size

After calibration, all methods achieved the same miscoverage rate, as guaranteed by (62). However, the mean error bar size differed in the methods, as discussed below.

First, let us focus on the comparison of our approach with Wiener and MCALens. PnPMass in the standard and residual versions produces smaller error bars than the two model-driven methods (see the y-axis in Fig. 3). We hypothesize that the combination of higher reconstruction accuracy and adaptive pre-calibrated error bars obtained with order-2 moment networks helps to distinguish between smooth regions and peak areas. This might in turn result in a more refined uncertainty quantification by correctly assigning higher uncertainties near peaks. In

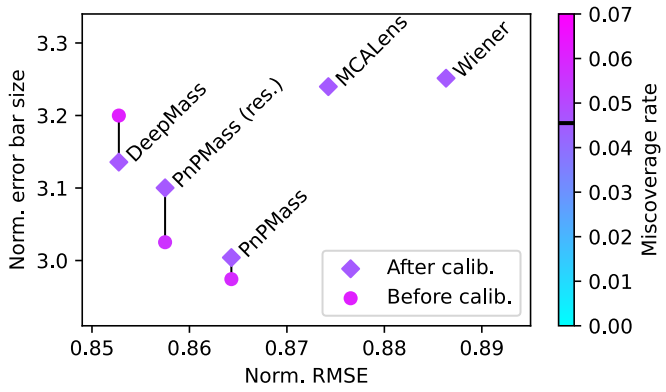


Fig. 3: Mean length of prediction intervals plotted against reconstruction accuracy before and after calibration. Both metrics have been normalized w.r.t. to the ℓ^2-norm of each image. The marker colors indicate empirical miscoverage rates, the target α being marked on the color bar. The metrics are computed over the κTNG test set (512 examples). All mass-mapping algorithms were run with the same parameters as in Table 2. For Wiener and MCALens, only the post-calibration miscoverage rate is displayed, since their error bars were initialized to 0.

contrast, Wiener and MCALens produce error bars that are independent of the input images: they are initialized at $\mathbf{0}$ and subsequently calibrated using a parameter that only depends on the pixel location.

More surprisingly, PnPMass also produces smaller error bars than DeepMass, but they are slightly less accurate. This result is statistically significant because it is observed for all 512 test images. To explain this phenomenon, we propose the following conjecture: DeepMass detects more peaks than PnPMass, which improves reconstruction accuracy, but is also more prone to hallucinations (i.e., false detections), which negatively affects the confidence level. In other words, the optimal precision-recall balance for peak detection depends on the metric that is to be prioritized. Consequently, a trade-off arises between reconstruction accuracy and error bar size. The same reasoning applies when we compare the standard and residual versions of PnPMass.

It is worth noting that error bars might in principle have been initialized at $\mathbf{0}$ for all mass-mapping methods, avoiding the need to train an order-2 moment network for DeepMass and PnPMass. However, this would ultimately result in larger calibrated error bars to compensate for the poorer initialization. A plot similar to Fig. 3, with zero-initialization for all methods, is shown in Appendix G, Fig. G.1b.

6.2.2. Effect of the calibration on the confidence intervals

For all methods based on moment networks, the pre-calibration error bars yield a miscoverage rate slightly above the target α. Interestingly, calibration reduced the error bar size for DeepMass, which may seem counterintuitive. This effect arises from the optimization strategy described in Sect. 4.2.2. Our interpretation is that while error bars were underestimated for most pixels, some regions, particularly around peaks, were overly conservative, leading to an overall overestimation that calibration was able to correct.

Table 3: Computation times for κTNG.

Method	Training Order 1	Training Order 2	Calibration	Inference
Wiener	–	–	9″	6″
MCALens	–	–	44″	24″
DeepMass‡	07:27′	46:33′	18″	7″
PnPMass	31:41′	47:20′	1′42″	50″
PnPMass (res.)	33:43′	43:44′	1′48″	54″

Notes. Training times were estimated for the first- and second-order moment networks on a single NVIDIA Quadro RTX 6000 GPU. Calibration and inference were performed on a single NVIDIA GeForce GTX 1080 Ti GPU, using 1 024 and 512 examples, respectively. Inference includes the computation of a point estimate, the initialization of error bars, and their calibration using the parameters computed at the previous stage. The training times are expressed in hours, not minutes. ‡The order-2 moment network for DeepMass was trained for 100 epochs, as the validation curve indicated that training was not yet complete after 20 epochs, in contrast to the order-1 network.

6.3. Mass-mapping visualization

We applied the two versions of PnPMass to the COSMOS shear field, which we binned at the κTNG resolution. A visual representation of the two reconstructions is shown in Fig. 4, together with DeepMass and MMGAN reconstructions.[3] Additional visuals are provided in Appendix H, Fig. H.2, including the position of known x-ray clusters, as well as uncertainty maps (standard deviation per pixel).

A visual example of a reconstructed convergence map from a κTNG simulation, along with its predicted standard deviation and bounds, is shown in Appendix H, Fig. H.3. While the post-calibration miscoverage rate remains constant on average, there are situations in which a given method may systematically fail to estimate uncertainty correctly. This is particularly noticeable for the Wiener solution around peak values: the blue spots in the bottom image indicate miscoverage. In contrast, our methods better account for peak values because the order-2 moment networks tends to predict a higher uncertainty in these regions. However, conformal prediction does not guarantee control of the miscoverage rate in specific situations such as peak values. One possible improvement would be to employ conditional conformal prediction (Gibbs et al. 2025), which we leave for future work.

6.4. Computation times

The computation times for training, calibration, and inference are reported in Table 3. While calibration and inference with PnPMass are slower than with DeepMass due to its iterative nature, these differences are negligible compared to training times, which are longer by several orders of magnitude than the inference times. This highlights one of the main advantages of our approach: unlike DeepMass and MMGAN, training is required only once.

7. Conclusion

We introduced PnPMass, a PnP-based mass-mapping method that combines the strengths of DeepMass and DeepPosterior, namely, flexibility (no need to retrain for each new observation)

[3] We did not reproduce MMGAN ourselves; the reconstruction was downloaded from the Zenodo repository provided by the authors.

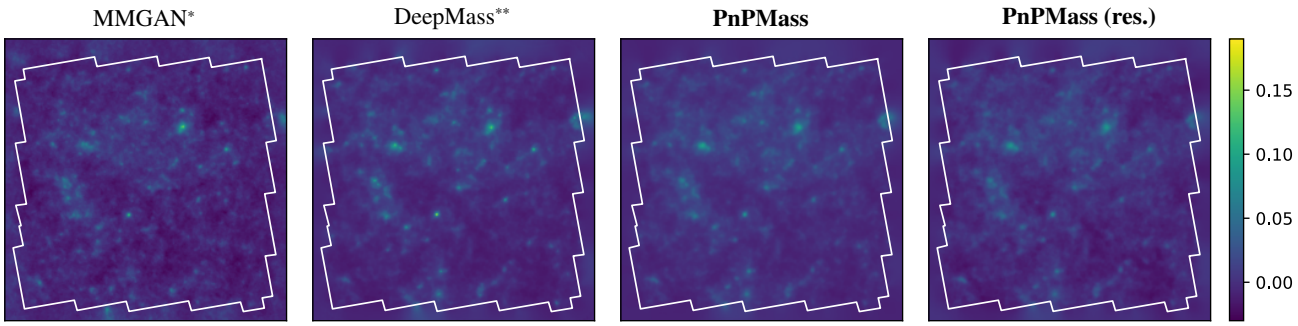


Fig. 4: Reconstruction of the COSMOS field with several deep-learning-based mass-mapping methods. For consistency with MMGAN (left), we concatenated the bright and faint catalogs before binning to the κTNG resolution. *Directly provided by Whitney et al. (2025). **DeepMass was retrained to take the noise level and mask from the bright + faint catalogs into account.

and fast inference, while maintaining competitive reconstruction accuracy. We further proposed a fast UQ strategy for PnPMass based on moment networks, followed by a conformal prediction to mitigate several sources of bias, including model uncertainty. Our results show that PnPMass yields the smallest calibrated error bars of all evaluated mass-mapping methods. Together, these contributions make our approach well suited to handle the large data volumes expected from upcoming stage-IV surveys such as Euclid and Rubin.

Several directions remain open to fully integrate our framework into weak-lensing pipelines. First, further investigations are needed to assess the robustness of our approach across different cosmologies. Our experiments focused on κTNG, a dataset simulated from a single set of cosmological parameters. A more realistic setting would involve datasets spanning a range of cosmological models. Second, although conformal prediction provides coverage guarantees on average, it may fail to control miscoverage around peak structures, which are particularly important for cosmological parameter inference (Tersenov et al. 2025). A promising avenue to address this limitation is to employ conditional conformal prediction (Gibbs et al. 2025), rather than the standard version used here. Finally, it will be crucial to assess how uncertainties estimated at the mass-mapping stage propagate to cosmological parameter inference and affect the resulting FoM. In particular, this analysis could clarify whether the residual variant should be preferred over the standard version of PnPMass, as optimal cosmological parameter estimation may depend on a trade-off between reconstruction accuracy and size of the confidence intervals.

Data availability

To ensure reproducibility, all software, scripts, and notebooks used in this study are available on GitHub, at `https://github.com/hubert-leterme/weaklensing_uq.git`.

Acknowledgements. This work was funded by the TITAN ERA Chair project (contract no. 101086741) within the Horizon Europe Framework Program of the European Commission, and the French Agence Nationale de la Recherche (ANR-22-CE31-0014-01 TOSCA and ANR-18-CE31-0009 SPHERES).

References

Abdar, M., Pourpanah, F., Hussain, S., et al. 2021, Inf. Fusion, 76, 243
Angelopoulos, A. N., Kohli, A. P., Bates, S., et al. 2022, in Proceedings of the 39th International Conference on Machine Learning (PMLR), 717–730
Aoyama, S. D., Osato, K., & Shirasaki, M. 2025, PASJ, submitted [arXiv:2505.00345]
Bobin, J., Starck, J.-L., Sureau, F., & Fadili, J. 2012, Adv. Astron., 2012, 703217
Ebner, A. & Haltmeier, M. 2024, IEEE Trans. Image Process., 33, 1476
Ekmekci, C. & Cetin, M. 2021, in Proceedings of the IEEE/CVF International Conference on Computer Vision, 4018–4027
Fan, C.-M., Liu, T.-J., & Liu, K.-H. 2022, in 2022 IEEE International Symposium on Circuits and Systems (ISCAS), 2333–2337
Finoguenov, A., Guzzo, L., Hasinger, G., et al. 2007, ApJS, 172, 182
Gibbs, I., Cherian, J. J., & Candès, E. J. 2025, J. R. Stat. Soc. Ser. B: Stat. Methodol., 87, 1100
Hurault, S., Leclaire, A., & Papadakis, N. 2022, in Proceedings of the 39th International Conference on Machine Learning (PMLR), 9483–9505
Jeffrey, N., Lanusse, F., Lahav, O., & Starck, J.-L. 2020, MNRAS, 492, 5023
Jeffrey, N. & Wandelt, B. D. 2020, in Third Workshop on Machine Learning and the Physical Sciences (NeurIPS 2020)
Kaiser, N. & Squires, G. 1993, ApJ, 404, 441
Kamilov, U. S., Bouman, C. A., Buzzard, G. T., & Wohlberg, B. 2023, IEEE Signal Proc. Mag., 40, 85
Kilbinger, M. 2015, Rep. Prog. Phys., 78, 086901
Lanusse, F., Starck, J.-L., Leonard, A., & Pires, S. 2016, A&A, 591, A2
Laumont, R., Bortoli, V. D., Almansa, A., et al. 2022, SIAM J. Imaging Sci., 15, 701
Leterme, H., Fadili, J., & Starck, J.-L. 2025, A&A, 694, A267
Mobasher, B., Capak, P., Scoville, N. Z., et al. 2007, ApJS, 172, 117
Osato, K., Liu, J., & Haiman, Z. 2021, MNRAS, 502, 5593
Pesquet, J.-C., Repetti, A., Terris, M., & Wiaux, Y. 2021, SIAM J. Imaging Sci., 14, 1206
Postels, J., Ferroni, F., Coskun, H., Navab, N., & Tombari, F. 2019, in Proceedings of the IEEE/CVF International Conference on Computer Vision, 2931–2940
Remy, B., Lanusse, F., Jeffrey, N., et al. 2023, A&A, 672, A51
Romano, Y., Patterson, E., & Candes, E. 2019, in Advances in Neural Information Processing Systems, Vol. 32 (Curran Associates, Inc.)
Ryu, E., Liu, J., Wang, S., et al. 2019, in Proceedings of the 36th International Conference on Machine Learning (PMLR), 5546–5557
Schrabback, T., Hartlap, J., Joachimi, B., et al. 2010, A&A, 516, A63
Scoville, N., Abraham, R. G., Aussel, H., et al. 2007, ApJS, 172, 38
Shi, G., Zhu, Y., Tremblay, J., et al. 2021, in 2021 IEEE International Conference on Robotics and Automation (ICRA), 5200–5207
Shirasaki, M., Moriwaki, K., Oogi, T., et al. 2021, MNRAS, 504, 1825
Shirasaki, M., Yoshida, N., & Ikeda, S. 2019, Phys. Rev. D, 100, 043527
Starck, J.-L., Donoho, D. L., Fadili, M. J., & Rassat, A. 2013, A&A, 552, A133
Starck, J.-L., Murtagh, F., & Fadili, J. 2015, Sparse Image and Signal Processing: Wavelets and Related Geometric Multiscale Analysis (Cambridge University Press)
Starck, J.-L., Themelis, K. E., Jeffrey, N., Peel, A., & Lanusse, F. 2021, A&A, 649, A99
Tachella, J., Terris, M., Hurault, S., et al. 2025, J. Open Source Software, 10, 8923
Terris, M., Repetti, A., Pesquet, J.-C., & Wiaux, Y. 2020, in IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)
Terris, M., Tang, C., Jackson, A., & Wiaux, Y. 2025, MNRAS, 537, 1608
Tersenov, A., Baumont, L., Starck, J.-L., & Kilbinger, M. 2025, A&A, 698, A25
Venkatakrishnan, S. V., Bouman, C. A., & Wohlberg, B. 2013, in 2013 IEEE Global Conference on Signal and Information Processing, 945–948
Whitney, J. J., Liaudat, T. I., Price, M. A., Mars, M., & McEwen, J. D. 2025, MNRAS, 542, 2464
Zhang, K., Li, Y., Zuo, W., et al. 2022, IEEE Trans. Pattern Anal. Mach. Intell., 44, 6360

Appendix A: Proof of Proposition 1

We start with consider this preparatory lemma.

Lemma 1. *Suppose that (H.3) and (H.4) hold. Then, for* $\rho \in [\rho^\star, 1[$*, the forward operator* $\mathcal{F}_{\tilde{\gamma}}(\cdot, \tau)$ *is* ρ*-contractive on* $(\ker \tilde{\mathbf{B}}\tilde{\mathbf{A}})^\perp$ *if, and only if, the step-size* τ *satisfies (16). The smallest contraction factor* $\rho = \rho^\star$ *is obtained for* $\tau = \tau^\star$*, with*

$$\tau^\star := \frac{2}{\lambda_{\max} + \lambda_{\min}}. \tag{A.1}$$

Proof. Denote for short the subspace $\mathbb{V} := (\ker \tilde{\mathbf{B}}\tilde{\mathbf{A}})^\perp$. By construction, and since (H.4) holds, we have:

$$\mathcal{F}_{\tilde{\gamma}}(\cdot, \tau) : \mathbb{V} \to \mathbb{V}. \tag{A.2}$$

Let $\boldsymbol{\kappa}'$ and $\boldsymbol{\kappa}'' \in \mathbb{V}$. The forward operator is an affine map, and we thus have

$$\mathcal{F}_{\tilde{\gamma}}(\boldsymbol{\kappa}', \tau) - \mathcal{F}_{\tilde{\gamma}}(\boldsymbol{\kappa}'', \tau) = (\mathbf{I} - \tau\tilde{\mathbf{B}}\tilde{\mathbf{A}})(\boldsymbol{\kappa}' - \boldsymbol{\kappa}''). \tag{A.3}$$

Since $\tilde{\mathbf{B}}\tilde{\mathbf{A}}$ is symmetric positive definite on the orthogonal to its kernel, the Lipschitz constant L_τ of $\mathcal{F}_{\tilde{\gamma}}(\cdot, \tau)$ on $\mathbb{V}$ is exactly given by

$$L_\tau = \max(|1 - \tau\lambda_{\min}|, |1 - \tau\lambda_{\max}|). \tag{A.4}$$

Consequently, $\mathcal{F}_{\tilde{\gamma}}(\cdot, \tau)$ is ρ-contractive on $\mathbb{V}$ if, and only if, $L_\tau \leq \rho$. Expanding this inequality, and considering that $\lambda_{\min} \leq \lambda_{\max}$, this is equivalent to (16). The lower bound $\rho^\star$ introduced in (15) ensures the non-emptiness of the interval. Moreover, setting $\rho = \rho^\star$ reduces the range of possible step-size values to $\tau = \tau^\star$ such as introduced in (A.1). Finally, having $\rho < 1$ provides the contractive property of $\mathcal{F}_{\tilde{\gamma}}(\cdot, \tau)$.

Let us turn to the proof of Proposition 1.

Proof. According to (H.1) and (A.2), both $\mathcal{D}_{\hat{\theta}}$ and $\mathcal{F}_{\tilde{\gamma}}(\cdot, \tau)$ map $\mathbb{V}$ to $\mathbb{V}$. Combining this with the non-expansiveness of $\mathcal{D}_{\hat{\theta}}$ on $\mathbb{V}$ (H.2), we have, for all $\boldsymbol{\kappa}'$ and $\boldsymbol{\kappa}'' \in \mathbb{V}$,

$$\left\|\mathcal{H}_{\tilde{\gamma}}(\boldsymbol{\kappa}', \tau) - \mathcal{H}_{\tilde{\gamma}}(\boldsymbol{\kappa}'', \tau)\right\| \leq \left\|\mathcal{F}_{\tilde{\gamma}}(\boldsymbol{\kappa}', \tau) - \mathcal{F}_{\tilde{\gamma}}(\boldsymbol{\kappa}'', \tau)\right\|. \tag{A.5}$$

Invoking now Lemma 1 and using the step-size choice in (H.5), we get

$$\left\|\mathcal{H}_{\tilde{\gamma}}(\boldsymbol{\kappa}', \tau) - \mathcal{H}_{\tilde{\gamma}}(\boldsymbol{\kappa}'', \tau)\right\| \leq \rho \left\|\boldsymbol{\kappa}' - \boldsymbol{\kappa}''\right\|. \tag{A.6}$$

Clearly, $\mathcal{H}_{\tilde{\gamma}}(\cdot, \tau)$ is a ρ-contraction on $\mathbb{V}$. Since $\mathbb{V}$ is obviously a complete metric space, we can now invoke the Banach-Picard fixed point theorem on $\mathcal{H}_{\tilde{\gamma}}(\cdot, \tau)$ to get that it has a unique fixed point $\hat{\boldsymbol{\kappa}} \in \mathbb{V}$. Moreover, using (H.1) and an induction argument, $(\boldsymbol{\kappa}^{(k)})_{k\in\mathbb{N}} \subset \mathbb{V}$, and thus, the sequence converges linearly to $\hat{\boldsymbol{\kappa}}$ at the rate ρ.

Appendix B: Insights into convergence properties

In this section, we sketch a mathematical framework for studying the convergence properties of our algorithm, which leads to constrain the noise level at which the denoiser must be trained, as explained in Sect. 3.2.

Inserting (12) into the fixed-point equation (18), we infer that

$$\hat{\boldsymbol{\kappa}} = \mathcal{D}_{\hat{\theta}}\Big(\boldsymbol{\kappa} + \hat{\boldsymbol{r}} + \tau\tilde{\mathbf{B}}(-\tilde{\mathbf{A}}\hat{\boldsymbol{r}} + \tilde{\boldsymbol{n}})\Big) = \mathcal{D}_{\hat{\theta}}\Big(\boldsymbol{\kappa} + (\mathbf{I} - \tau\tilde{\mathbf{B}}\tilde{\mathbf{A}})\hat{\boldsymbol{r}} + \boldsymbol{n}_0\Big), \tag{B.1}$$

where we denoted the residual $\hat{\boldsymbol{r}} := \hat{\boldsymbol{\kappa}} - \boldsymbol{\kappa}$ and $\boldsymbol{n}_0 = \tau\tilde{\mathbf{B}}\tilde{\boldsymbol{n}}$. Observe that $\boldsymbol{n}_0$ is a sample of $\mathbf{N}_0$ as defined in (22).

Clearly, $\mathcal{D}_{\hat{\theta}}$ acts as a denoiser on the true convergence map $\boldsymbol{\kappa}$ corrupted by the "noise" $(\mathbf{I} - \tau\tilde{\mathbf{B}}\tilde{\mathbf{A}})\hat{\boldsymbol{r}} + \boldsymbol{n}_0$. Intuitively, one would like $(\mathbf{I} - \tau\tilde{\mathbf{B}}\tilde{\mathbf{A}})\hat{\boldsymbol{r}}$ to be small enough so that the dominating part of the noise would be that from $\boldsymbol{n}_0$. This intuition is exact when $\mathbf{A} = \mathbf{I}$ (i.e. denoising), in which case, with the natural choice $\tilde{\mathbf{B}} = \mathbf{I}$ and $\tau = 1$, (B.1) reads $\hat{\boldsymbol{\kappa}} = \mathcal{D}_{\hat{\theta}}(\boldsymbol{\kappa} + \boldsymbol{n}_0)$ with $\boldsymbol{n}_0 = \tilde{\boldsymbol{n}}$. One should then train the denoiser with noise $\boldsymbol{n}_0$.

To get some insight into this intuition in the general case, let us now bound the residual $\hat{\boldsymbol{r}}$. Let $\boldsymbol{\zeta}$ be a sample of the zero-mean random noise $\mathbf{Z}$ used during training, and assume that the support of the distribution of $\mathbf{K} + \mathbf{Z}$ is contained in $(\ker \tilde{\mathbf{B}}\tilde{\mathbf{A}})^\perp$. Subtracting $\boldsymbol{\kappa}$ to both sides of (B.1), we have:

$$\begin{aligned}
\|\hat{\boldsymbol{r}}\| &\leq \left\|\mathcal{D}_{\hat{\theta}}\Big(\boldsymbol{\kappa} + (\mathbf{I} - \tau\tilde{\mathbf{B}}\tilde{\mathbf{A}})\hat{\boldsymbol{r}} + \boldsymbol{n}_0\Big) - \mathcal{D}_{\hat{\theta}}\Big(\boldsymbol{\kappa} + \boldsymbol{\zeta}\Big)\right\| + \left\|\mathcal{D}_{\hat{\theta}}\Big(\boldsymbol{\kappa} + \boldsymbol{\zeta}\Big) - \boldsymbol{\kappa}\right\| \\
&\leq \left\|(\mathbf{I} - \tau\tilde{\mathbf{B}}\tilde{\mathbf{A}})\hat{\boldsymbol{r}} + \boldsymbol{n}_0 - \boldsymbol{\zeta}\right\| + \left\|\mathcal{D}_{\hat{\theta}}\Big(\boldsymbol{\kappa} + \boldsymbol{\zeta}\Big) - \boldsymbol{\kappa}\right\| \qquad \text{(B.2)} \\
&\leq \rho\,\|\hat{\boldsymbol{r}}\| + \|\boldsymbol{n}_0 - \boldsymbol{\zeta}\| + \left\|\mathcal{D}_{\hat{\theta}}\Big(\boldsymbol{\kappa} + \boldsymbol{\zeta}\Big) - \boldsymbol{\kappa}\right\|, \qquad \text{(B.3)}
\end{aligned}$$

where we used (H.2) in (B.2), and the triangle inequality and Lemma 1 (Appendix A) in (B.3). We then deduce, after taking the expectation with respect to the joint distribution of $(\mathbf{K}, \mathbf{N}, \mathbf{Z})$, that

$$\begin{aligned}
\mathbb{E}\left[\left\|\hat{\mathbf{R}}\right\|\right] &\leq (1-\rho)^{-1}\left(\mathbb{E}\left[\left\|\mathbf{N}_0 - \mathbf{Z}\right\|\right] + \mathbb{E}\left[\left\|\mathcal{D}_{\hat{\theta}}(\mathbf{K} + \mathbf{Z}) - \mathbf{K}\right\|\right]\right) \\
&\leq (1-\rho)^{-1}\left(\mathbb{E}\left[\left\|\mathbf{N}_0 - \mathbf{Z}\right\|^2\right]^{\frac{1}{2}} + \mathbb{E}\left[\left\|\mathcal{D}_{\hat{\theta}}(\mathbf{K} + \mathbf{Z}) - \mathbf{K}\right\|^2\right]^{\frac{1}{2}}\right),
\end{aligned} \tag{B.4}$$

where we used Jensen's inequality.

We aim to minimize the expected norm of the residual such as introduced above. Ideally, the denoiser $\mathcal{D}_{\boldsymbol{\theta}}$ would be trained to minimize the population quadratic risk, which is nothing but the second term in the upper bound (B.4). In practice, however, training is performed using the empirical quadratic risk, which introduces an additional generalization error term that must be accounted for. Regarding the first term in (B.4), it captures the discrepancy between the noise in training and that in the PnP iteration. We have

$$\mathbb{E}\left[\|\mathbf{N}_0 - \mathbf{Z}\|^2\right] = \mathbb{E}\left[\|\mathbf{N}_0\|^2\right] + \mathbb{E}\left[\|\mathbf{Z}\|^2\right] - 2\nu\mathbb{E}\left[\|\mathbf{N}_0\|^2\right]^{1/2}\mathbb{E}\left[\|\mathbf{Z}\|^2\right]^{1/2},$$

where $\nu \in [-1, 1]$ is the correlation coefficient between $\mathbf{N}_0$ and $\mathbf{Z}$. Minimizing the above expression with respect to $\mathbb{E}\left[\|\mathbf{Z}\|^2\right]$ and ν gives that the optimal values are, respectively, $\mathbb{E}\left[\|\mathbf{N}_0\|^2\right]$ and 1. Clearly, the noise $\mathbf{Z}$ ought to have the same variance as $\mathbf{N}_0$ and has correlation 1 with it.

Appendix C: Properties of $\tilde{\mathbf{B}}\tilde{\mathbf{A}}$

Recall that $\tilde{\mathbf{B}}\tilde{\mathbf{A}} := \tilde{\mathbf{A}}^\top\tilde{\boldsymbol{\Sigma}}^{-1/2}\tilde{\mathbf{A}}$ with the choice made in (24). This is obviously a symmetric positive semidefinite matrix, hence (H.3) follows. Observe also that:

$$\operatorname{ran}\tilde{\mathbf{B}} = \operatorname{ran}\tilde{\mathbf{A}}^\top\tilde{\boldsymbol{\Sigma}}^{-1/2} = \operatorname{ran}\tilde{\mathbf{A}}^\top, \tag{C.1}$$

since $\tilde{\boldsymbol{\Sigma}}$ is invertible. On the other hand,

$$(\ker\tilde{\mathbf{B}}\tilde{\mathbf{A}})^\perp = (\ker\tilde{\mathbf{A}}^\top\tilde{\boldsymbol{\Sigma}}^{-1/2}\tilde{\mathbf{A}})^\perp = (\ker\tilde{\mathbf{A}}^\top\tilde{\mathbf{A}})^\perp = (\ker\tilde{\mathbf{A}})^\perp, \tag{C.2}$$

since $\tilde{\boldsymbol{\Sigma}}$ is invertible. As $\operatorname{ran}\tilde{\mathbf{A}}^\top = (\ker\tilde{\mathbf{A}})^\perp$, we conclude that (H.4) holds.

Appendix D: Proof of (26)

We embark from (11) to see that

$$\mathbf{A}^\top\mathbf{A} = \mathrm{Re}(\mathbf{A}^*\mathbf{A}),$$

where $\mathbf{A}^*$ is the adjoint matrix of $\mathbf{A}$. Using now (2), we get

$$\mathbf{A}^*\mathbf{A} = \mathbf{F}^*|\mathbf{P}|^2\mathbf{F}.$$

From (3), it follows that $|\mathbf{P}|^2$ is a diagonal matrix with

$$|\mathbf{P}[0,0]| = 0 \quad \text{and} \quad |\mathbf{P}[i,i]| = 1, \forall i \geq 1.$$

This entails that

$$\mathbf{A}^*\mathbf{A} = \mathbf{F}^*\mathbf{F} - \mathbf{F}[1,:]^\top\mathbf{F}[1,:],$$

where $\mathbf{F}[1,:]$ = is the first row of $\mathbf{F}$. Let $\mathbb{1}$ be the column vector of ones of size K^2. Since $\mathbf{F}[1,:] := \mathbb{1}^\top/K$ (the zero frequency component), and $\mathbf{F}$ is unitary, we get that

$$\mathbf{A}^*\mathbf{A} = \mathbf{I} - \mathbb{1}\mathbb{1}^\top/K^2 = \mathbf{I} - 1/K^2.$$

Appendix E: Computing $\lambda_{\max}$ and $\lambda_{\min}$

$\lambda_{\max}$ can be easily computed from the power iteration applied to $\tilde{\mathbf{A}}^\top\tilde{\boldsymbol{\Sigma}}^{-1/2}\tilde{\mathbf{A}}$ starting from a random uniform vector. We now turn to computing the smallest nonzero eigenvalue of $\tilde{\mathbf{A}}^\top\tilde{\boldsymbol{\Sigma}}^{-1/2}\tilde{\mathbf{A}}$. From Appendix C, we have $\tilde{\mathbf{A}}^\top\tilde{\boldsymbol{\Sigma}}^{-1/2}\tilde{\mathbf{A}} = \ker\mathbf{A} = \mathrm{ran}\,\mathbb{1}$, whose orthogonal is the subspace of zero-mean vectors. Let $\lambda_1 = 0 < \lambda_2 = \lambda_{\min} \leq \cdots \leq \lambda_{K^2} = \lambda_{\max}$, be the eigenvalues of $\tilde{\mathbf{A}}^\top\tilde{\boldsymbol{\Sigma}}^{-1/2}\tilde{\mathbf{A}}$ sorted in ascending order. Denote $\boldsymbol{u}_i$ the corresponding unit norm eigenvectors, where $\boldsymbol{u}_1 = \mathbb{1}^\top/K$. Define the matrix $\mathbf{C} := \lambda_{\max}\mathbf{I} - \tilde{\mathbf{A}}^\top\tilde{\boldsymbol{\Sigma}}^{-1/2}\tilde{\mathbf{A}}$, whose eigenvalues are $\lambda_{\max} - \lambda_i$, and eigenvectors are $\boldsymbol{u}_i$. The key observation is that $\mathbf{C}$ is again symmetric positive semidefinite and, orthogonally to $\mathrm{ran}\,\mathbb{1}$, the largest eigenvalue of $\mathbf{C}$ is $\lambda_{\max} - \lambda_{\min}$.

The idea now is to apply the power iteration to $\mathbf{C}$ with an appropriate initial vector. For instance, we draw a vector $\boldsymbol{x}_0$ form the uniform distribution (or any distribution which has a density with respect to the Lebesgue measure), in which case we can write

$$\boldsymbol{x}_0 = \sum_{i=1}^{K^2} c_i\boldsymbol{u}_i,$$

where, for all i, $c_i \neq 0$ with probability one. We then project $\boldsymbol{x}_0$ on $(\ker\mathbf{A})^\perp = (\mathrm{ran}\,\mathbb{1})^\perp$, that is,

$$\bar{\boldsymbol{x}}_0 = \boldsymbol{x}_0 - \frac{1}{K^2}\sum_{i=1}^{K^2}\boldsymbol{x}_0[i].$$

Since $(\ker\mathbf{A})^\perp$ is the span of $(\boldsymbol{u}_i)_{i\geq 2}$, we have

$$\bar{\boldsymbol{x}}_0 = \sum_{i=2}^{K^2} c_i\boldsymbol{u}_i.$$

Hence,

$$\mathbf{C}^k\bar{\boldsymbol{x}}_0 = c_2(\lambda_{\max} - \lambda_{\min})^k\boldsymbol{u}_2 + \sum_{i=3}^{K^2-1} c_i(\lambda_{\max} - \lambda_i)^k\boldsymbol{u}_i.$$

Applying the power iteration to $\mathbf{C}$ with initial vector $\bar{\boldsymbol{x}}_0$ allows computing $\lambda_{\max} - \lambda_{\min}$. Plugging the value of $\lambda_{\max}$ (computed using, e.g., power iteration) gives an estimate of $\lambda_{\min}$.

Appendix F: PnPMass and the FBS algorithm

We assume the existence of a proper convex, lower semicontinuous function g such that

$$\mathcal{D}_{\hat{\boldsymbol{\theta}}}(\boldsymbol{\kappa}', \tau) = \mathrm{prox}_{\tau g}(\boldsymbol{\kappa}'), \tag{F.1}$$

where the proximal operator is defined as

$$\mathrm{prox}_{\tau g} : \boldsymbol{\kappa}' \mapsto \underset{\boldsymbol{\kappa}''}{\mathrm{argmin}}\ \frac{1}{2}\left\|\boldsymbol{\kappa}'' - \boldsymbol{\kappa}'\right\|_2^2 + \tau g(\boldsymbol{\kappa}''). \tag{F.2}$$

Under this assumption, the fixed-point iteration (31) can be written as

$$\mathcal{H}_{\tilde{\boldsymbol{\gamma}}}(\boldsymbol{\kappa}', \tau) := \mathrm{prox}_{\tau g}\left[\boldsymbol{\kappa}' - \tau\nabla f_{\tilde{\boldsymbol{\gamma}}}(\tilde{\mathbf{A}}\,\cdot)(\boldsymbol{\kappa}')\right], \tag{F.3}$$

where we have considered the following data fidelity term

$$f_{\tilde{\boldsymbol{\gamma}}}(\tilde{\mathbf{A}}\boldsymbol{\kappa}') := \frac{1}{2}\left\|\tilde{\boldsymbol{\gamma}} - \tilde{\mathbf{A}}\boldsymbol{\kappa}'\right\|_{\tilde{\boldsymbol{\Sigma}}^{-1/2}}^2. \tag{F.4}$$

The operator (F.3) is nothing but the fixed-point operator of the well-known FBS algorithm. It consists of a forward step—gradient descent with respect to the data fidelity term $f_{\tilde{\boldsymbol{\gamma}}}(\tilde{\mathbf{A}}\,\cdot)$—and a backward step—proximal mapping with respect to the regularization term g. This is the reason why PnP-type algorithms with an FBS structure are usually referred to as PnP-FBS in the literature (Ryu et al. 2019; Ebner & Haltmeier 2024).

It is well-known that if the step-size satisfies $0 < \tau < 2/\left\|\tilde{\mathbf{A}}^\top\tilde{\boldsymbol{\Sigma}}^{-1/2}\tilde{\mathbf{A}}\right\|$, then the sequence of iterates of FBS with operator (F.3) converges to a minimizer $\hat{\boldsymbol{\kappa}}$ of the following minimization problem—similar in form to (8):

$$\hat{\boldsymbol{\kappa}} \in \underset{\boldsymbol{\kappa}'}{\mathrm{argmin}}\ f_{\tilde{\boldsymbol{\gamma}}}(\tilde{\mathbf{A}}\boldsymbol{\kappa}') + g(\boldsymbol{\kappa}'), \tag{F.5}$$

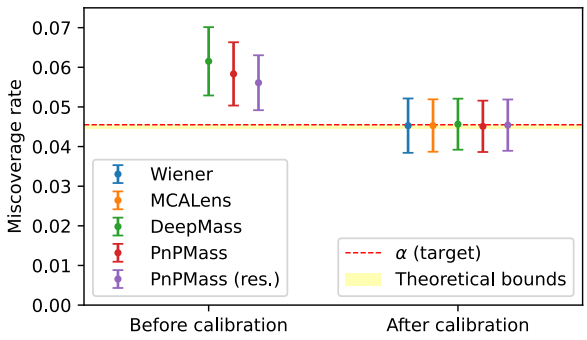
with the proviso that the set of minimizers is non-empty.

The FBS has been widely used for astronomical data analysis; see the comprehensive treatment in (Starck et al. 2015). The iterative Wiener filtering algorithm (Bobin et al. 2012) for mass mapping (see Sect. 2.2.1), is also an instance of the FBS algorithm.

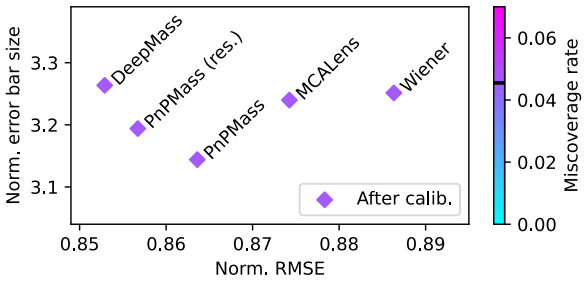
While some proximal operators can be seen as denoisers, the converse is clearly not true. Thus, there is no reason that a deep-learning based denoiser $\mathcal{D}_{\hat{\boldsymbol{\theta}}}(\cdot, \tau)$ to be interpreted as a proximal operator. At best, the denoiser network can be constrained during training to be firmly non-expansive, that is, to be the resolvent of a maximally monotone set-valued operator as proposed in (Terris et al. 2020). While this point of view brings interpretability, it is however not clear it leads to better denoising results. Consequently, PnP algorithms such as PnPMass are not expected to solve any optimization problem in general, hence the fixed-point perspective that we advocated in this paper.

It is worth recalling again (see Sect. 3.3) that the data fidelity term defined in (F.4) does not correspond to the negative log-likelihood typically used in Bayesian frameworks: the Mahalanobis norm would be defined with respect to $\tilde{\boldsymbol{\Sigma}}^{-1}$ rather than $\tilde{\boldsymbol{\Sigma}}^{-1/2}$. Furthermore, the regularization function g does not necessarily correspond to a negative log-prior; see Starck et al. (2013) for a thorough discussion. As a result, $\hat{\boldsymbol{\kappa}}$ should not be interpreted as a maximum a posteriori (MAP) estimate, even if (F.1) is satisfied.

Appendix G: Additional plots



(a) Empirical miscoverage rate over active pixels, before and after calibration (using 1 024 calibration examples). Error bars indicate the mean and standard deviation across the 512 test examples from the κTNG dataset. As in Fig. 3, Wiener and MCALens results are only displayed after calibration. The theoretical bounds (in yellow) come from (62).



(b) Mean length of prediction intervals plotted against reconstruction accuracy, after calibration. Unlike Fig. 3, all methods have been initialized with zero-valued error bars (no order-2 moment network). We observe that ignoring the pre-calibration step with order-2 moment networks result in larger error bars for PnPMass and DeepMass, by comparison with Fig. 3. Furthermore, DeepMass now produces larger error bars than Wiener or MCALens, despite being significantly more accurate.

Fig. G.1: Left: Miscoverage rate before and after calibration. Right: Zero-initialization error bars.

Appendix H: Visual representations

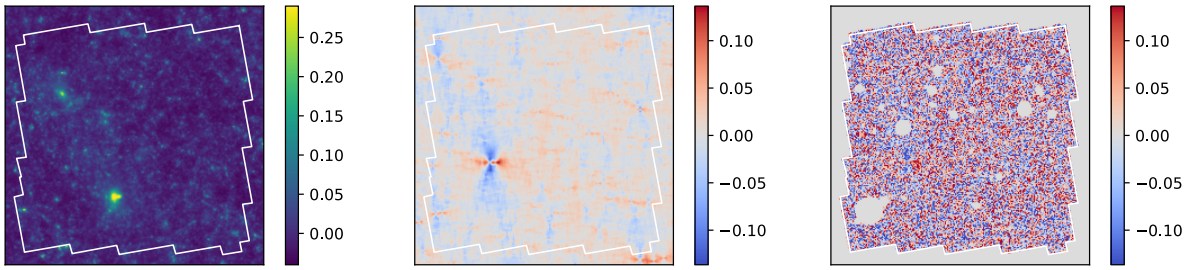


Fig. H.1: Example of mock convergence map obtained from κTNG simulations (left), and the real part of the corresponding clean (middle) and noisy (right) shear map. The COSMOS boundaries are delimited in white.

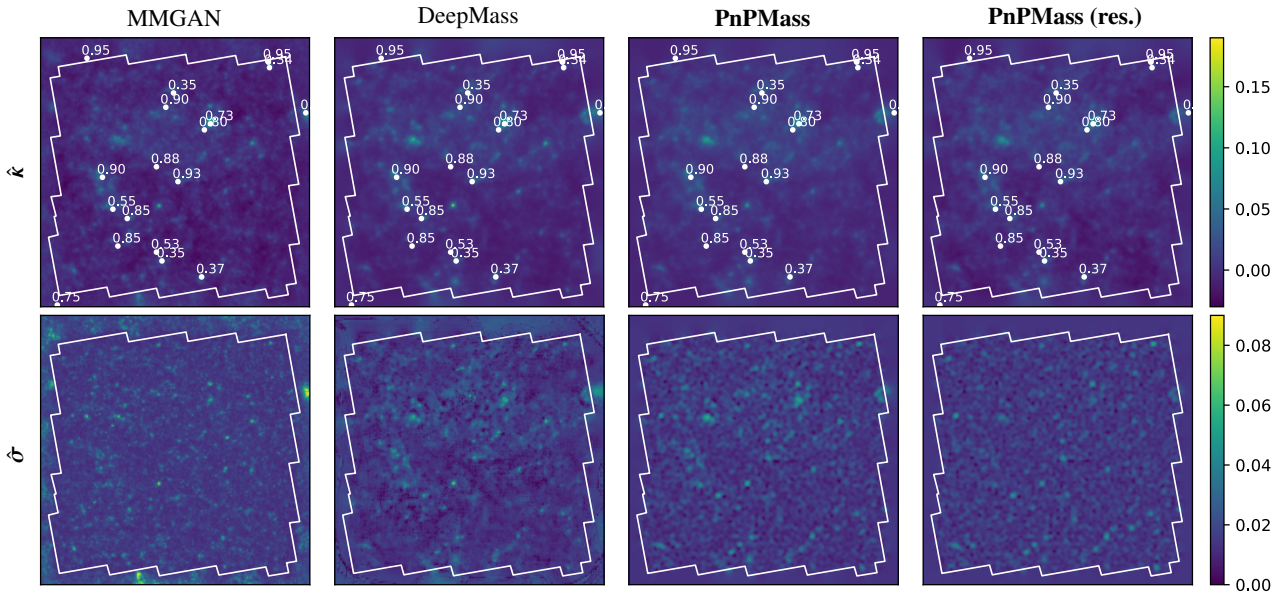


Fig. H.2: Reconstruction of the COSMOS field, using the same settings as in Fig. 4. Top row: point estimates; the white scatter plot indicates the location of known x-ray clusters (Finoguenov et al. 2007). Bottom row: uncertainty maps, obtained with order-2 moment networks for PnPMass and DeepMass, and by computing the per-pixel standard deviation over 32 samples for MMGAN.

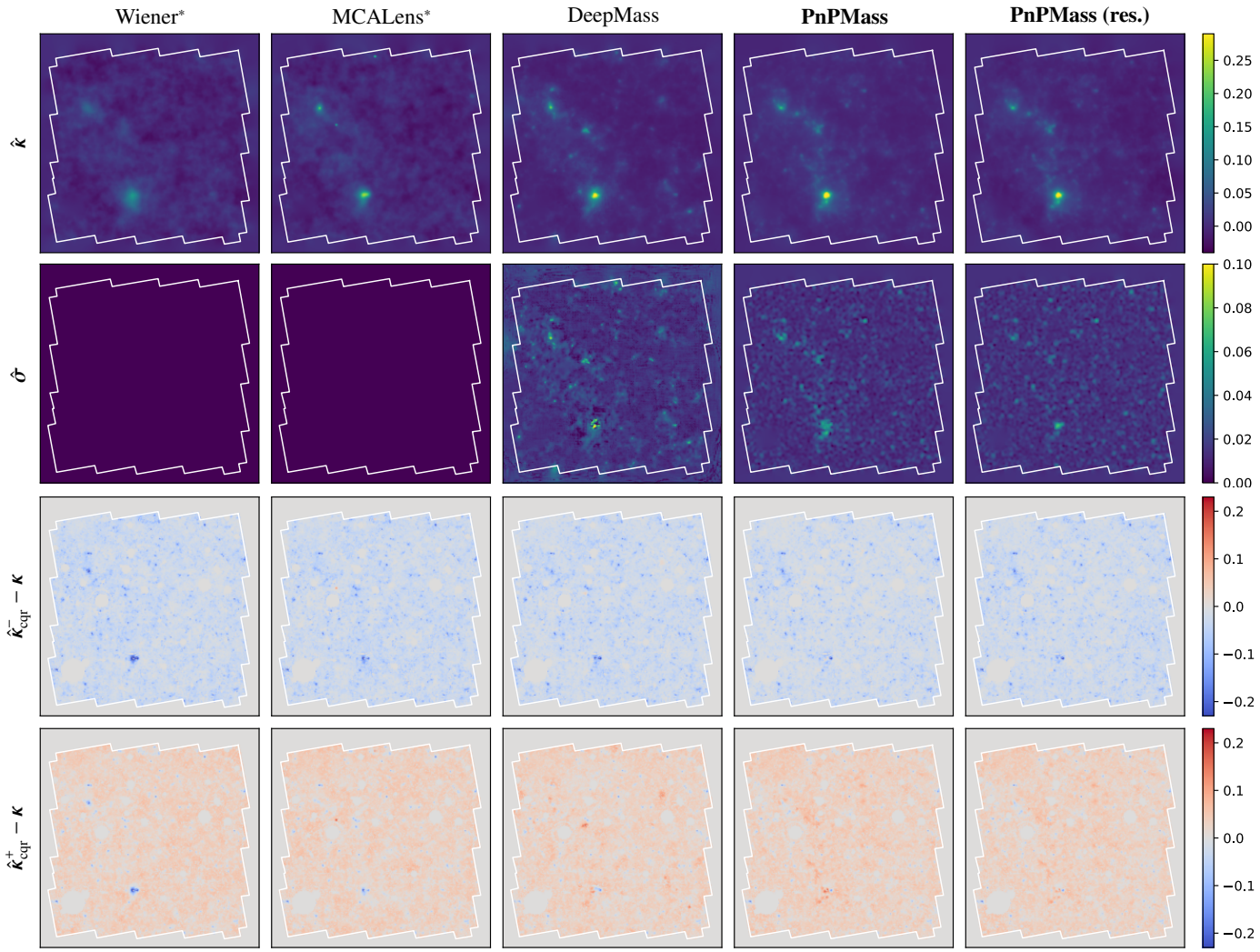


Fig. H.3: Example of mass-mapping reconstructions on the κTNG test dataset, for each mass-mapping method, run with the same parameters as in Table 2. The corresponding ground truth convergence map $\boldsymbol{\kappa}$ and noisy input $\boldsymbol{\gamma}$ are displayed in Fig. H.1. Top row: Point estimates, $\hat{\boldsymbol{\kappa}}$. Second row: Standard deviation estimates, $\hat{\boldsymbol{\sigma}}$, obtained with order-2 moment networks (used for pre-calibration error bars). Bottom rows: Lower and upper bounds, $\hat{\boldsymbol{\kappa}}^{-}_{\mathrm{cqr}}$ and $\hat{\boldsymbol{\kappa}}^{+}_{\mathrm{cqr}}$, after calibration with CQR. For visualization purpose, the bounds have been centered with respect to the ground truth $\boldsymbol{\kappa}$. *Zero-sized error bars before calibration.