Unbiased Parameter Estimation and Inference using Deep Summaries

Application to a Cosmological Model

Abstract

The ability to obtain reliable point estimates of model parameters is of crucial importance in many fields of physics. This is often a difficult task given that the observed data can have a very high number of dimensions. In order to address this problem, we propose a novel approach to construct parameter estimators with a quantifiable bias using an order expansion of highly compressed deep summary statistics of the observed data. These summary statistics are learned automatically using an information maximising loss. Given an observation, we further show how one can use the constructed estimators to obtain approximate Bayes computation (ABC) likelihood estimates and their corresponding uncertainties that can be used for parameter inference using Gaussian process regression even if the likelihood is not tractable. We validate our method with an application to the problem of cosmological parameter inference of weak lensing mass maps. We show in that case that the constructed estimators are unbiased and have an almost optimal variance, while the posterior distribution obtained with the Gaussian process regression is close to the true likelihood and performs better or equally well than comparable methods.

1 INTRODUCTION

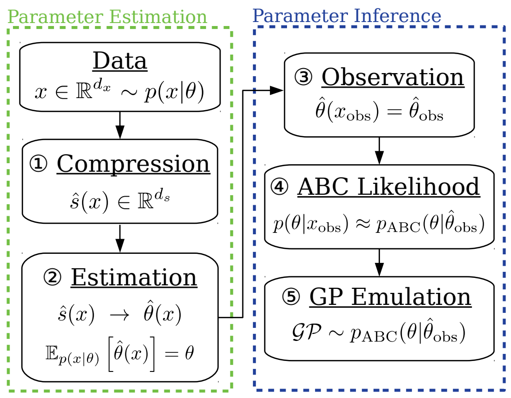
Parameter estimation and inference are common problems in many fields of physics. In parameter estimation, one tries to construct unbiased point estimators for model parameters and in parameter inference one is interested in the posterior distribution of the underlying model parameters, given a set of observations. These two problems are often difficult to solve in real-world scenarios for multiple reasons. First, there are usually only very few relevant parameters in physical models that are used to describe highly complex systems whose interactions are governed by physical laws. Due to the nature of these complex interactions it is rarely possible to directly observe the underlying model parameters, but rather one has to extract them from high dimensional data. A prime example for this is the field of cosmology or, more specifically, weak graviational lensing (see e.g. Schneider (2005) for a review). The currently most accepted ΛCDM (dark energy, cold dark matter) cosmological model contains only five relevant parameters that have to be extracted from the shape measurements of millions of galaxies. The relationship between the model parameters and the observed data is highly non-linear and not analytically tractable. A practical way to extract the desired parameters is via complex models and large cosmological simulations that can require millions of CPU and GPU hours to run (Potter et al., 2017). Given the large amount of data produced by these simulations, a standing problem is the compression of the observed data, such that one can optimally constrain the cosmological parameters. The recent advances in machine learning offer new possibilities in the field of cosmology. Especially deep neural networks, have gained a lot of attention (Charnock et al., 2018; **?**) and have been used in particular for parameter estimation (Herbel et al., 2018) and inference (Fluri et al., 2019; Jeffrey et al., 2020).

An important problem in the construction of parameter estimators is the quantification of their bias. For deep neural networks there has been extensive work concerning uncertainty estimation (Gal, 2016). Common approaches include drop out (Gal, Ghahramani, 2016) and Bayes by Backprop (Blundell et al., 2015). These methods help to quantify the intrinsic model uncertainties, however, they do not quantify potential biases of the models. In Chernozhukov et al. (2018) they are able to construct unbiased estimators, but the presented method is not applicable for general non-linear relationships between model parameters and observations.

In parameter inference one tries to calculate the posterior distribution and confidence intervals of the underlying model parameters given a set of observations. If the posterior distribution can be evaluated, one can resort to classic methods such as Markov chain Monte Carlo (MCMC) to obtain the desired confidence intervals. However, even if the posterior distribution is known, it might be unfeasible to perform a MCMC run if the likelihood requires the evaluation of expensive simulators. Approaches using Gaussian process regression have been developed (Teckentrup, 2019; Pellejero-Ibañez et al., 2020) in order to reduce the number of required likelihood evaluations and will be discussed in more detail below. In cases where the posterior distribution is unknown, one can resort to likelihood free inference (LFI) methods such as approximate Bayesian computation (ABC) (Fan, Sisson, 2018). These methods, while accurate and stable, again require a large number of simulations that might be unfeasible to run in a reasonable time. Combinations of Gaussian process regression and ABC have also been proposed, e.g. to emulate a discrepancy measure between simulations (Järvenpää et al., 2017). Further, Järvenpää et al. (2019) analysed Gaussian process regression of noisy log-likelihood estimates, primarily in a synthetic likelihood setting, where the unknown likelihood is approximated by a Gaussian distribution, focusing on uncertainty estimation and acquisition strategies for new simulations. Other LFI methods utilize machine learning, for example Alsing et al. (2019) proposed to use neural density estimators. Such methods require orders of magnitude fewer simulations, but require the network to learn high dimensional distributions.

In this work, we consider the common scenario where we only have access to a fixed grid of simulations generated prior to the analysis, additional simulations are therefore not obtainable, making it impossible to use acquisition strategies like the ones discussed in Järvenpää et al. (2019). The problem we address is to construct estimators of the underlying model parameters and to calculate confidence intervals of these parameters given a set of observations without having access to the likelihood function. An overview of the general approach is shown in Figure 1 where we distinguish between two parts of the pipeline: i) parameter estimation, where we construct an estimator of the model parameters and ii) parameter inference where we find confident regions for the model parameters using the estimator constructed in i). To achieve this goal we make three key contributions:

- In the first step where we construct a compressed representation of the data, we relax prior assumptions made in Charnock et al. (2018) so that the approach is applicable to the realistic setting used in our experiments.
- Next, we develop a method to construct estimators of the underlying model parameters with quantifiable biases using the compressed simulations.
- Finally, we demonstrate how one can utilize the constructed estimators to calculate ABC log-likelihood estimates and their corresponding uncertainties, which can in turn be used for parameter inference using Gaussian process regression.



Parameter Estimation
Parameter Inference
Data
$x \in \mathbb{R}^{d_x} \sim p(x|\theta)$
① Compression
$\hat{s}(x) \in \mathbb{R}^{d_s}$
② Estimation
$\hat{s}(x) \rightarrow \hat{\theta}(x)$
$\mathbb{E}_{p(x|\theta)}\left[\hat{\theta}(x)\right] = \theta$
③ Observation
$\hat{\theta}(x_{\text{obs}}) = \hat{\theta}_{\text{obs}}$
④ ABC Likelihood
$p(\theta|x_{\text{obs}}) \approx p_{\text{ABC}}(\theta|\hat{\theta}_{\text{obs}})$
⑤ GP Emulation
$\mathcal{GP} \sim p_{\text{ABC}}(\theta|\hat{\theta}_{\text{obs}})$

Figure 1: Overview of the presented approach that consists of two parts: first constructing parameter estimators $\hat{\theta}$ and then finding confidence regions of the model parameters.

Each of these aspects is important to run the pipeline as a whole in a realistic setting and we validate our findings using a cosmological model that is comparable to modern weak lensing surveys (e.g. Zürcher et al. (2020)). Finally, the code used in this work can be found in the supplementary material and will be available on `github`.

2 PROBLEM DESCRIPTION

We start with a detailed description of the two problems of parameter estimation and inference. In parameter estimation, the goal is to estimate the unobserved, usually low dimensional, model parameter $\theta \in \Theta \subseteq \mathbb{R}^{d_\theta}$ from observed, high dimensional data $x \in \mathcal{X} \subseteq \mathbb{R}^{d_x}$. The constructed estimator $\hat{\theta}(x)$ should be unbiased, i.e.

$$\mathbb{E}_{p(x|\theta)}[\hat{\theta}(x)] \equiv \mathbb{E}[\hat{\theta}(x)|\theta] \equiv \int_{\mathcal{X}} \hat{\theta}(x)p(x|\theta)\mathrm{d}x = \theta \quad (1)$$

and minimize the variance around the true value, i.e.

$$\mathrm{Var}[\hat{\theta}](\theta) = \mathbb{E}_{p(x|\theta)}[(\hat{\theta}(x) - \theta)^2]. \quad (2)$$

Such an estimator does not always exist. For example, if we have a family of distributions which fulfills $p(x|\theta) = p(x|\theta')$ for $\theta \neq \theta'$, it is impossible to construct an unbiased estimator $\hat{\theta}(x)$ for all $\theta \in \Theta$. We therefore only consider distributions $p(x|\theta)$ that allow the construction of such an estimator.

In parameter inference the goal is to calculate the posterior distribution $p(\theta|x)$ of the model parameter given an observation x and subsequently to find confidence intervals for the model parameter. The two conditional distributions $p(x|\theta)$ and $p(\theta|x)$, also called likelihood, are often unknown. However, we assume that there exists a computationally expensive simulator, which can produce samples of the observed space $x \sim p(x|\theta)$ for any possible model parameter θ. Throughout this work, we will consider an experiment design where the computationally expensive simulator was run on a set of n_θ different model parameters and n_x samples were drawn from the distribution $p(x|\theta_\alpha)$ for each parameter θ_α in the set. We will use Greek indices to indicate parameters contained in a set (e.g. $\theta_\alpha \in \mathbb{R}^{d_\theta}$) and Latin indices (e.g. $\theta_i \in \mathbb{R}$) to refer to individual elements of vectors and matrices.

3 PARAMETER ESTIMATION

We start with a description of the first part of the pipeline shown in Figure 1, where our goal is to construct an estimator of the underlying model parameters. A simple way to construct such an estimator $\hat{\theta}(x)$ using the generated simulations is by doing regression. One can train a neural network $h(x,w)$, parameterized by weights $w \in \mathbb{R}^{d_w}$ to predict the underlying model parameter θ in a supervised fashion using, for example, the square loss over mini-batches of size N_b,

$$L(w) = \sum_{\alpha=1}^{N_b} \left\| h(x_\alpha, w) - \theta_\alpha^{\text{True}} \right\|^2,$$

where $\theta_\alpha^{\text{True}}$ is the true parameter of the randomly sampled simulation x_α. However, using this approach, there is no guarantee that the trained network $h(x,w)$ is an unbiased estimator of the model parameter. A potential source of this bias can be the choice of simulated parameters θ_α. By choosing a fixed grid of parameters θ_α one implicitly places a prior over the parameters the network tries to estimate. In **?** and Villaescusa-Navarro et al. (2020) it was shown that the network can learn this prior, potentially leading to large biases in the predictions of the network in areas where there is a lower parameter sampling density.

To alleviate this issue, we propose to split the problem into two parts. We start by training a network to produce informative, low-dimensional summary statistics $\hat{s}(x) = h(x,w) \in \mathbb{R}^{d_s}$, with $d_\theta \leq d_s \ll d_x$. In a second step, we use these summary statistics to construct an estimator $\hat{\theta}(x) \equiv \hat{\theta}(\hat{s}(x))$.

3.1 OBTAINING INFORMATIVE SUMMARIES

As mentioned earlier, we face the problem of producing a compact representation of the data, which we name *summary* (corresponding to step ① in Figure 1). To achieve this, Charnock et al. (2018) proposed to optimize the Cramér–Rao bound that expresses a lower bound on the variance of an estimator as a function of the Fisher information, starting with the assumption that the summary statistics generated by a neural network follow a Gaussian distribution. We will next show that this assumption is not necessary. Formally, the Fisher Information Matrix (FIM) is defined as

$$\mathbf{I}_{ij}(\theta) = \mathbb{E}_{p(x|\theta)} \left[\frac{\partial \log(p(x|\theta)}{\partial \theta_i} \frac{\partial \log(p(x|\theta)}{\partial \theta_j} \right], \quad (3)$$

where θ_i denotes the i-th entry of the underlying model parameter $\theta \in \mathbb{R}^{d_\theta}$. The Fisher information matrix is used in the Cramér-Rao bound, which bounds the covariance matrix of any summary $\hat{s}(x)$ as

$$\text{Cov}_\theta(\hat{s}) \succcurlyeq \frac{\partial \Psi_\theta(\hat{s})}{\partial \theta} \mathbf{I}(\theta)^{-1} \frac{\partial \Psi_\theta(\hat{s})}{\partial \theta}^T, \quad (4)$$

where the derivative with respect to θ is understood as a Jacobian matrix, $A \succcurlyeq B$ indicates that $A - B$ is positive definite and we defined

$$\Psi_\theta(\hat{s}) = \mathbb{E}_{p(x|\theta)}\left[\hat{s}(x)\right]. \quad (5)$$

Note, that if the summary $\hat{s}(x)$ is itself an unbiased estimator of θ, the Jacobian becomes the identity and the covariance matrix is bounded directly by the Fisher information matrix. A summary that achieves equality is called sufficient, which in turn implies Bayesian sufficiency, i.e.

$$p(\theta|x) = p(\theta|\hat{s}(x)), \quad (6)$$

which means that the summary retains all relevant information about the model parameters. Similarly to the work of Blum, François (2010) we will only consider summary statistics having the same dimensionality as the model parameter $d_s = d_\theta$. However, in the appendix A we explain how the same approach can be used for summaries with $d_s > d_\theta$. Assuming that the Jacobian is invertible, one can transform equation (4) into

$$\begin{aligned} -\log\det(\mathbf{I}(\theta)) \leq\ & \log\det(\text{Cov}_\theta(\hat{s})) \\ & - 2\log\det\left(\frac{\partial \Psi_\theta(\hat{s})}{\partial \theta}\right). \end{aligned} \quad (7)$$

The right side of this inequality can be used as a loss function to construct informative summaries. Note that this loss is equivalent to the log-determinant of the inverse Fisher matrix used in Charnock et al. (2018), without relying on the assumption that the summary follows a Gaussian likelihood, showing the generality of the approach.

Calculating the covariance matrix of the summaries in equation (7) is straightforward. The simulator, however, is generally not differentiable, preventing the exact calculation of the Jacobian. As in Charnock et al. (2018), we calculate the Jacobian via finite differences. One starts by generating a set of simulations using a fiducial parameter[1] θ_{fid} and small perturbations of it $\theta_{\text{fid}}^\pm = \theta_{\text{fid}} \pm \Delta\theta^\pm$, resulting in $2d+1$ parameters. Using this set of parameters one can approximate the Jacobian as

$$\frac{\partial \Psi_\theta(\hat{s})}{\partial \theta} \approx \frac{\Psi_{\theta_{\text{fid}}+\Delta\theta^+}(\hat{s}) - \Psi_{\theta_{\text{fid}}-\Delta\theta^-}(\hat{s})}{\Delta\theta^+ - \Delta\theta^-}, \quad (8)$$

[1]The term "fiducial parameter" refers to a reference parameter typically located in a region of interest and based on a previous analysis.

where the expectation $\Psi_\theta(\hat{s})$ is empirically calculated over multiple samples from $p(x|\theta)$. This reduces the variance at the price of an increased computational effort spent on a single step during the training. However, training the network only around a single (or a few) fiducial points leads to the advantage that the remaining datapoints can be used for testing and validation. It also reduces the overall training time compared to simple regression networks as one does not have to cycle through the entire data set of simulations.

Regularization The minimization of the righthand side of inequality (7) could be numerically unstable because the loss allows the entries of the Jacobian or the summary statistics itself to become arbitrary large or small. The reason for this is that there are infinitely many summary statistics that have the same Fisher information. In fact, any one-to-one transformation of summary statistics preserves the Fisher information, showing the degeneracy of the possible solutions. To cope with this issue we add a regularization term to the loss

$$L_{\text{regu}}(w) = \lambda \left\| \frac{\Psi_\theta(\hat{s})}{\partial \theta} - \mathbb{I}_{d_\theta} \right\|^2, \tag{9}$$

where λ is a hyper-parameter, $\mathbb{I}_d$ represents the d-dimensional identity matrix and we use the Frobenius norm as matrix norm. Additionally one can add a regularization term proportional to the length of the summary vectors $\|\hat{s}(x)\|$ to avoid arbitrarily large summaries. However, we never observed any sort of over- or underflow in the summary statistics during our experiments and did not use this type of regularization.

Adding Expert Knowledge We briefly mention that the approach described above allows for an easy incorporation of expert knowledge into the summaries. Consider the scenario where a set of summaries $(\hat{s}_1(x), \ldots, \hat{s}_n(x))$ generated using expert knowledge is available. One can simply incorporate this additional information into the model by concatenating the summaries to the fully connected layers. Another alternative is to perform another round of compression by training a different neural network to compress the list of summaries containing the previously trained summaries and the expert summaries $(\hat{s}(x), \hat{s}_1(x), \ldots, \hat{s}_n(x))$.

3.2 CONSTRUCTING ESTIMATORS

Next, we use the generated summaries to construct estimators of the underlying parameters (corresponding to step ② in Figure 1). To do so, we first perform the following series expansion:

$$\hat{\theta}_i(x) = a_i + b_{ij}\hat{s}^j(x) + c_{ijk}\hat{s}^j(x)\hat{s}^k(x) + \ldots, \tag{10}$$

where we used the Einstein summation convention, meaning that we sum over all indices that appear as lower and upper index in a term. One can choose the coefficients a_i, b_{ij}, etc. such that they minimize the bias of the resulting estimator around the chosen fiducial point θ_{fid}. In this work, we will only consider first-order expansions, however, the construction of higher-order terms follows the same principle. We start by expanding the expectation value of the summaries around the fiducial parameter

$$\Psi_\theta(\hat{s}) = \Psi_{\theta_{\text{fid}}}(\hat{s}) + \left.\frac{\partial \Psi_\theta(\hat{s})}{\partial \theta}\right|_{\theta=\theta_{\text{fid}}} (\theta - \theta_{\text{fid}}) + \mathcal{O}(\Delta\theta^2),$$

where we defined $\Delta\theta = \theta - \theta_{\text{fid}}$. Plugging this into the first-order expansion of equation (10), one can find the coefficients

$$a_i = \theta_{i,\text{fid}} - \left(\frac{\partial \Psi_{\theta_{\text{fid}}}(\hat{s})}{\partial \theta}\right)^{-1}_{ij} \Psi^j_{\theta_{\text{fid}}}(\hat{s}), \quad b_{ij} = \left(\frac{\partial \Psi_{\theta_{\text{fid}}}(\hat{s})}{\partial \theta}\right)^{-1}_{ij}$$

such that

$$\mathbb{E}_{p(x|\theta)}[\hat{\theta}(x)] = \theta + \mathcal{O}(\Delta\theta^2).$$

In practice one has to estimate the coefficient a_i and b_{ij} from the sampled simulations x of the fiducial parameter, leading to an additional small error from this approximation. We briefly note here, that the construction of higher-order estimators does not only require the estimation of the expected values $\Psi_{\theta_{\text{fid}}}(\hat{s})$ and its derivative, but also from higher-order moments. We show how to construct a second order estimator, having an error of order $\mathcal{O}(\Delta\theta^3)$, in appendix B.

Since the construction of the estimators requires only simulations of the fiducial parameter and its perturbations, the simulations from the other parameters can be used to measure the performance of the constructed estimators.

Combining Estimators Although the approach we described uses a single estimation point, it can simply be extended to combine several estimation points. For instance, Lavancier, Rochet (2014) proposed to use a linear combination.

4 PARAMETER INFERENCE

In this section, we show how one can use the previously described estimators to constrain the underlying model parameters using parameter inference (second part of the pipeline in Figure 1). The goal of parameter inference is to find confidence regions of the underlying model parameter θ, given an observation x_{obs} (corresponding to step ③ in Figure 1). This is no trivial task in practice, even if the conditional distribution $p(\theta|x_{\text{obs}})$ is known, because it requires the integration of the likelihood function to derive marginal distributions.

4.1 COMMON APPROACHES

Markov Chain Monte Carlo A common way of performing parameter inference with a known likelihood function is to use Markov chain Monte Carlo (MCMC). However, even if the conditional probability $p(\theta|x_{\text{obs}})$ is known, running a MCMC can be unfeasible if the evaluation of the likelihood is expensive. A possible solution to this issue was proposed by Pellejero-Ibañez et al. (2020). They used Gaussian Process regression to emulate the expensive likelihood $p(\theta|x_{\text{obs}})$ and performed a MCMC on the emulated likelihood to obtain the parameter constraints. The obtained one and two dimensional marginal distributions from the Gaussian Process emulation were in excellent agreement with the ground truth. However, this approach is not applicable if the analytic form of the likelihood, or good approximations of it, are not available.

Neural Density Estimators In the case of an unknown likelihood $p(\theta|x_{\text{obs}})$ one has to use likelihood free inference (LFI). A promising LFI technique is based on neural density estimators, as for instance proposed in Alsing et al. (2019), where a combination of mixture density models and auto-regressive flows is used to learn the distribution of the summaries $p(\hat{s}|\theta)$. The likelihood $p(\theta|s_{\text{obs}})$ can then be obtained by using Bayes theorem. We use the implementation of this method, called `PyDelfi`[2], as baseline for our experiments in section 5. The advantage of `PyDelfi` is that it can be applied to our settings as it does not require any on-the-fly simulations. Further, the constructed estimators, described in section 3, are usually close to the actual parameter making it much simpler to learn the distribution $p(\hat{\theta}(x)|\theta)$, as opposed to summaries whose distribution is completely unknown or have a higher dimensionality.

Approximate Bayesian Computation Another common way of performing LFI is called approximate Bayesian computation (ABC). In ABC one replaces the high dimensional data x with low a dimensional summary $\hat{s}(x)$. The ABC approximation of the likelihood (e.g. see Fan, Sisson (2018)) is then given by

$$p_{ABC}(\theta|s_{\text{obs}}) \propto \int K_h(\|s - s_{\text{obs}}\|)p(s|\theta)p(\theta)\mathrm{d}s, \qquad (11)$$

where $s_{\text{obs}} = \hat{s}(x_{\text{obs}})$ is the summary of the observation, $K_h(x) = K(x/h)/h$ is a kernel density function with scale parameter h, $\|\cdot\|$ defines a norm and $p(\theta)$ denotes the prior distribution of the model parameters. It is straightforward to see that the ABC likelihood converges to the conditional probability $p(\theta|s_{\text{obs}})$ for $h \to 0$. If the summary $\hat{s}(x)$ is sufficient, we even have

$$\lim_{h\to 0} p_{ABC}(\theta|s_{\text{obs}}) = p(\theta|s_{\text{obs}}) = p(\theta|x_{\text{obs}}). \qquad (12)$$

The simplest ABC algorithm is called rejection ABC which works similarly to MCMC. It first generates a random sample θ from $p(\theta)$ and then generates a sample x from $p(x|\theta)$. The new parameter θ is accepted if $\|\hat{s}(x) - s_{\text{obs}}\| < \varepsilon$, where the threshold ε is a hyper-parameter corresponding to the scale h. Using this technique, one can generate samples from the ABC posterior and use them to calculate the marginal distributions and eventually the confidence regions. The downside of this algorithm is that it requires a large number of simulations x, even with a carefully selected threshold. More advanced ABC algorithms such as population Monte Carlo ABC (PMC-ABC) (Ishida et al., 2015) require fewer on-the-fly simulations, but can not be applied to a fixed grid of simulations.

[2] `https://github.com/justinalsing/pydelfi`

4.2 GAUSSIAN PROCESS ABC

A combination of Gaussian process regression and ABC has been proposed by Järvenpää et al. (2017), using the Gaussian process to emulate the discrepancy measure of the ABC. Later, Järvenpää et al. (2019) used Gaussian process regression to emulate (noisy) likelihood estimates. However, they mainly consider a synthetic likelihood setting where the unknown likelihood is approximated by a Gaussian distribution, focusing on the error estimation of the posterior and acquisition strategies for new simulations. Although we do also use Gaussian process regression in our approach, we take a significant step further where we compute ABC likelihood estimates and their corresponding uncertainties using the estimator presented in Section 3. Afterwards, we use Gaussian process regression to emulate the likelihood similarly to Järvenpää et al. (2019). This does not require on-the-fly simulations as in standard ABC methods but still allows likelihood free inference. Further, it makes it possible to directly learn the likelihood $p(\theta|\hat{\theta}_{\text{obs}})$ conditioned on a set observation $\hat{\theta}_{\text{obs}}$ and does not require to learn the conditional distribution $p(\theta|\hat{\theta})$ jointly in θ and $\hat{\theta}$ like neural density estimators.

We use our available n_x draws from the distribution $p(x|\theta)$ and utilize the constructed estimator of the previous section. We can then obtain an unbiased, consistent and non-negative Monte Carlo estimate of the ABC posterior described by equation (11) (corresponding to step ④ in Figure 1) via

$$\hat{p}_{ABC}(\theta|\hat{\theta}_{\text{obs}}) \propto \frac{p(\theta)}{n_x}\sum_{\alpha=1}^{n_x} K_h(\|\hat{\theta}_\alpha - \hat{\theta}_{\text{obs}}\|), \qquad (13)$$

where the normalization is irrelevant for our purpose. Note that the scaling parameter h can be used to smooth the ABC likelihood if the fixed grid is too coarse. A good choice for the norm inside the kernel density function is the Fisher distance (Charnock et al., 2018) around the fiducial parameter

$$\|\hat{\theta}_\alpha - \hat{\theta}_{\text{obs}}\|^2 = (\hat{\theta}_\alpha - \hat{\theta}_{\text{obs}})^T \hat{\mathbf{I}}_{\text{fid}}(\hat{\theta}_\alpha - \hat{\theta}_{\text{obs}}),$$

where we defined the estimated Fisher matrix

$$\hat{\mathbf{I}}_{\text{fid}} = \left.\frac{\partial \Psi_\theta(\hat{s})}{\partial \theta}\right|_{\theta=\theta_{\text{fid}}}^T \mathrm{Cov}_{\theta_{\text{fid}}}(\hat{s})^{-1} \left.\frac{\partial \Psi_\theta(\hat{s})}{\partial \theta}\right|_{\theta=\theta_{\text{fid}}}.$$

For a first order estimator this is equivalent to the Mahalanobis distance of the summary and scales the entries of the summary according to their variances. As Pellejero-Ibañez et al. (2020) pointed out, performing Gaussian Process regression on the likelihood $p(\theta|\hat{\theta}_{\text{obs}})$ itself is difficult. One reason for this, is that the likelihood is usually close to zero in a large subset of the parameter space Θ. This can be alleviated by performing the regression on the log-likelihood $\log(p(\theta|\hat{\theta}_{\text{obs}}))$ instead. Using equation (13) we can construct an estimator of the ABC log-likelihood

$$\hat{\mathcal{L}}_{ABC}(\theta) = \log\left(\frac{p(\theta)}{n_x}\sum_{\alpha=1}^{n_x} K_h(\|\hat{\theta}_\alpha - \hat{\theta}_{\text{obs}}\|\right), \qquad (14)$$

where we left out a constant offset coming from the normalization constant of the ABC posterior estimate. One should note that this estimator is still consistent, but not unbiased. The bias follows directly from Jensen's inequality and becomes negligible for sufficiently large sample sizes n_x. Another issue with normal Gaussian process regression is that the variance of the estimators depends on the underlying parameter θ and is not constant over the parameter space Θ. Using `gpflow`[3] G. Matthews de et al. (2017) we adopt the following model for the Gaussian process regression (corresponding to step ⑤ in Figure 1)

$$\begin{aligned} \mathcal{L}_{ABC} &\sim \mathcal{GP}(0, k(\cdot,\cdot)) \\ \hat{\mathcal{L}}_{ABC}(\theta_\alpha)|\mathcal{L}_{ABC}, \theta_\alpha &\sim \mathcal{N}\left(\mathcal{L}_{ABC}(\theta_\alpha), \sigma_\alpha^2\right), \end{aligned}$$

where $k(\cdot,\cdot)$ is a kernel function and σ_α^2 is the estimated variance of the ABC log-likelihood estimate. To obtain the σ_α we use a Taylor series expansion of the moments. If $g : \mathcal{U} \subseteq \mathbb{R} \to \mathbb{R}$ is a smooth function and $X \in \mathcal{U}$ a continuous random variable with finite mean and variance $\mu_X, \sigma_X^2 < \infty$, one can approximate the variance of the transformed random variable as

$$\text{Var}[g(X)] \approx \left.\frac{\mathrm{d}g(x)}{\mathrm{d}x}^2\right|_{x=\mu_X} \sigma_X^2. \qquad (15)$$

We find this to be a good approximation[4] for $g(x) = \log(x)$ if X is approximately normal distributed and $\mu_X/\sigma_X \gtrsim 2.5$. Therefore, we start by estimating the variance of our ABC posterior estimates using the central limit theorem

$$\text{Var}\left[\hat{p}_{ABC}(\theta_\alpha|\hat{\theta}_{\text{obs}})\right] \approx \frac{p(\theta_\alpha)^2}{n_x}\text{Var}\left[K_h(\|\hat{\theta}_\alpha - \hat{\theta}_{\text{obs}}\|\right],$$

and then propagate this variance to the log-likelihood via the Taylor series expansion

$$\sigma_\alpha^2 = \frac{p(\theta_i)^2}{n_x}\frac{\text{Var}\left[K_h(\|\hat{\theta}_\alpha - \hat{\theta}_{\text{obs}}\|\right]}{\hat{p}_{ABC}(\theta|\hat{\theta}_{\text{obs}})^2}. \qquad (16)$$

[3]`https://www.gpflow.org/`

[4]Using the mentioned bound, the next order term is approximately 10 times smaller for a Gaussian distributed X.

The evaluation of the Gaussian process with optimized kernel parameters is computationally cheap, making it straight forward to obtain all relevant properties of the posterior distribution $p_{ABC}(\theta|\hat{\theta}_{\text{obs}})$ with standard methods like MCMC.

5 EXPERIMENTS

To test our method, we apply it to the inference of cosmological parameters from weak gravitational lensing maps. For this purpose, we consider simplified Gaussian maps for which the sufficient summary statistics is known. In weak lensing, one measures the shapes of millions of galaxies across the sky that have been distorted by the density inhomogeneities in the universe on the line-of-sight, according to general relativity. The correlations between the measured shapes can then be used to construct convergence maps, which are maps of the mass overdensity projected on the sky. Convergence maps can be generated using large, computationally expensive, cosmological simulations. We refer the interested reader to Bartelmann, Schneider (2001) or Schneider (2005) for a detailed review.

5.1 SETUP

Model Parameter Our experiments are based on the flat wCDM cosmological model that extends the standard ΛCDM model with the dark energy equation of state parameter w_0. The remaining parameters of the ΛCDM are the total matter density fraction Ω_M, the baryonic matter density fraction Ω_b, the Hubble parameter $H_0 \equiv 100h$, the spectral index n_s and the fluctuation amplitude σ_8. Here, flat is referring to the total energy density parameter being equal to one $\Omega = \Omega_{\text{CDM}} + \Omega_b + \Omega_\Lambda \equiv 1$, where $\Omega_{\text{CDM}} = \Omega_M - \Omega_b$ is the dark matter density fraction and the dark energy density fraction Ω_Λ is determined by the constraint $\Omega = 1$, which corresponds to a geometrically flat universe. We consider two cases, i) a two-dimensional model (hereafter **2D**) where we only vary the parameters weak lensing is most sensitive to, namely Ω_M and σ_8; and ii) a six-dimensional model (hereafter **6D**) that allows all six parameters to vary.

Mock-Data Generation We consider a stage-III like survey, meaning that the convergence maps span $\sim 5000\,\text{deg}^2$

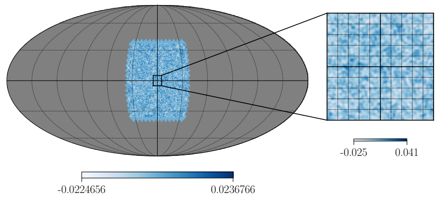


Figure 2: A GRF mimicking a convergence map of our mock survey generated for our fiducial parameter.

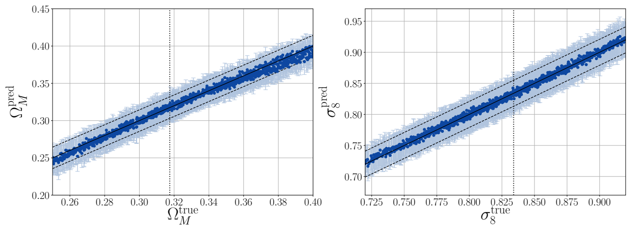


Figure 3: Mean predictions and their standard deviation of the test set of the **2D** model. The vertical dotted line indicates the fiducial paramter used for the training. The solid line indicates an unbiased prediction $y = x$, and the dashed lines above and below indicate the lower bound on the standard deviation for any estimator of the fiducial parameter.

on the sky (see Figure 2). Further, instead of using convergence maps generated from cosmological simulations, we use Gaussian random fields (GRF). The advantage of GRF is, that while being similar to realistic convergence maps, they are easy to model statistically. A detailed description of the data generation can be found in appendix C.2. A single sample x consists of four patches as shown in fiugre 2 corresponding to convergence maps at different distances from the observer. All patches were generated and pixelized using the Hierarchical Equal Area iso-Latitude Pixelization tool (`HEALPix`[5]) described in Gorski et al. (2005) and a single patch contains $\sim 400,000$ pixel. We chose the same fiducial parameters and perturbations as Villaescusa-Navarro et al. (2020) for the training, which are listed in table 2 of appendix C.1. Note that we place an additional prior on the degeneracy parameter $S_8 = \sigma_8\sqrt{\Omega/0.3}$, corresponding to the width of the degenerate contours (as shown in Figure 4). We generated $10,000$ simulations with the fiducial parameter and each of 12 individual perturbations. Additionally, we generated $n_x = 50$ simulations for $n_\theta = 1024$ different parameters sampled as a Sobol sequence inside the reported priors for the **2D** model and $n_\theta = 2048$ for the **6D** model.

GCNN Network The simulated survey area makes it impossible to neglect the curvature of the sky, i.e. to treat the data as a two-dimensional image. Therefore, it is not possible to apply standard convolutional neural networks directly on the data. We decided to use the graph-convolutional neural networks (GCNN) using the `HealPix` pixelization, called `DeepSphere`[6], described in Perraudin et al. (2018). `DeepSphere` treats the pixels of the generated data as graph, whose edges are given by the distances between the pixels on the sphere. More details of the networks and a quick overview of the functionality of `DeepSphere` can be found in appendix C.4.

[5]`http://healpix.sourceforge.net`

[6]`https://github.com/deepsphere/deepsphere-cosmo-tf2`

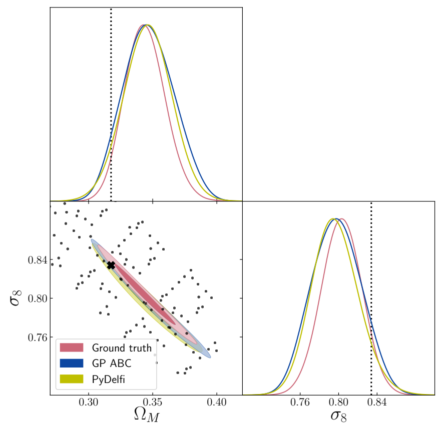


Figure 4: The posterior distributions of the **2D** model. The fiducial parameter values are indicated with a black cross in the two-dimensional marginal distribution and with vertical dotted lines in the one-dimensional marginal distributions. The black dots correspond to the parameters θ_α used for the GP ABC and `PyDelfi`.

Ground Truth The advantage of the model we use is that a sufficient summary statistics, the power spectrum (see appendix C.5), is known. All maps are generated from analytically predicted power spectra. This makes it possible to calculate the true likelihood $p(\theta|x_{\rm obs})$ for any observation $x_{\rm obs}$ and any valid parameter θ. Further, it allows for the explicit calculation of the Fisher information matrix for the fiducial parameters. These calculations are presented in appendix C.5 as well.

5.2 PARAMETER ESTIMATION

We trained a GCNN (see appendix C.4.2) for $100,000$ training steps using a batch size of 128. This batch size has to be multiplied with $2d_\theta + 1$, because of the necessary perturbations to estimate the derivatives with respect to the underlying parameters. Beside the loss defined in equation (7) we added the regularization term of equation (9) with parameter $\lambda = 10$. The observational noise (see appendix C.2.2) modeled as Gaussian random noise was sampled anew for each training step, but the same seed was used for each perturbation. Afterwards, we used the $10,000$ simulations of the fiducial parameter and its perturbations to construct a first order estimator. We compare the true parameters and the mean predictions of the constructed estimator of the **2D** model for the $n_x = 50$ maps generated for the $n_\theta = 1024$ test parameters in Figure 3. Even though the network was only trained on the fiducial parameter and its perturbations, the mean predictions are not significantly biased throughout the considered prior range. Further, the variance of the constructed estimator is close to the calculated lower bound of the known sufficent summary. A plot of the residues, i.e. the biases, can be found in appendix C.6.

Table 1: Overview of the parameter inference results. The number of evaluations corresponds to the number of different parameter θ used to obtain the posterior distribution. The constraints report the mean values and the 68% confidence regions. The JSD was calculated with respect to the MCMC run, which we consider as our ground-truth (lower values are better).

		Constraints							
	Method	Ω_M	Ω_b	h	n_s	σ_8	w_0	**JSD**	**Evaluations**
2D	MCMC	$0.34^{+0.01}_{-0.02}$	-	-	-	$0.80^{+0.02}_{-0.02}$	-	0.0	500,000
	GP ABC	$0.35^{+0.02}_{-0.02}$	-	-	-	$0.80^{+0.03}_{-0.03}$	-	**0.055**	128
	PyDelfi	$0.35^{+0.02}_{-0.02}$	-	-	-	$0.80^{+0.02}_{-0.03}$	-	0.166	128
6D	MCMC	$0.34^{+0.02}_{-0.02}$	$0.05^{+0.01}_{-0.01}$	$0.73^{+0.10}_{-0.10}$	$0.94^{+0.06}_{-0.06}$	$0.81^{+0.03}_{-0.03}$	$-1.01^{+0.17}_{-0.17}$	0.0	10,000,000
	GP ABC	$0.34^{+0.03}_{-0.03}$	$0.05^{+0.02}_{-0.01}$	$0.72^{+0.13}_{-0.10}$	$0.93^{+0.06}_{-0.08}$	$0.81^{+0.03}_{-0.04}$	$-1.05^{+0.09}_{-0.25}$	**0.026**†	2048
	PyDelfi	$0.34^{+0.02}_{-0.02}$	$0.05^{+0.02}_{-0.02}$	$0.70^{+0.11}_{-0.12}$	$0.94^{+0.07}_{-0.07}$	$0.80^{+0.03}_{-0.04}$	$-1.06^{+0.08}_{-0.24}$	0.028†	2048

† Mean Jensen-Shannon divergence of the 15 two-dimensional marginal distributions.

5.3 PARAMETER INFERENCE

Using the trained network we perform a GP ABC. We evaluated the ABC likelihood estimates as described in section 4 using a sigmoid kernel

$$K_h(x) = \frac{2}{\pi h} \frac{1}{\exp\left(\frac{x}{h}\right) + \exp\left(-\frac{x}{h}\right)},$$

with scale parameter $h = 0.35$ for the **2D** model and $h = 0.45$ for the **6D** model. We investigate the impact of the scale paramter on the resulting posterior in appendix C.8. For the **2D** model we use the first 128 parameters of the generated Sobol sequence and fit a Gaussian process using a standard Matèrn 5/2 covariance function with automatic relevance determination (ARD). This was done with the `gpflow` package. We also used the parameter normalization suggested in Pellejero-Ibañez et al. (2020), rotating and scaling the input parameters θ_α such that they are uncorrelated with zero mean and unit variance. The posterior distribution obtained by running a MCMC with the fitted GP is shown in Figure 4.

For the **6D** model, we use a Sobol sequence of $n_\theta = 2048$ parameters. In this model, most of the variance is caused by the observational noise. This can be used to effectively increase the sample size $n_x = 50$. For each simulation we create 50 version of observational noise and use the 50×50 mock surveys to estimate the ABC likelihoods. Afterwards, the same Gaussian process regression as for the **2D** model is performed. We provide a plot of the two-dimensional marginal posterior distributions in appendix C.7.

We use exactly the same data to train a neural density estimator. We use the same model as presented in section 6 of Alsing et al. (2019) only adapting the dimensions. The results can be seen in Figure 4. We also trained a neural density estimator of the summary statistics $\hat{s}$ as opposed to the estimates $\hat{\theta}$ for the **2D** model. However, the network did not converge to an acceptable result, showing that the distribution of the constructed estimators is easier to learn.

We assess the performance of our method in two ways: by comparing the mean of the posterior and by calculating the Jensen–Shannon (JS) divergence with respect to the MCMC run. A high precision is extremely important for comsological parameter inference. All results are listed in table 1. Both methods, GP ABC and `PyDelfi` perform very well on both models. The GP ABC, however, yields mean parameter values that are on average closer to the ground truth, as well as a lower JS divergence. The difference of the JS divergence is larger for the **2D** model because `PyDelfi` produces a posterior distribution that is longer in the $S_8 = \sigma_8\sqrt{\Omega_M/0.3}$ direction (i.e. thickness of the degenerate contours).

6 CONCLUSION

In this work, we presented a novel approach to construct parameter estimators from highly informative summaries. The constructed estimators have a quantifiable bias and are suitable for parameter inference using the presented GP ABC approach or the neural density estimators presented in Alsing et al. (2019). We validated our approach with a realistic cosmological model, where the constructed estimators are almost unbiased over the whole parameter range while having a close to optimal variance. Given a set of observations, the posterior distribution of the underlying model parameters using GP ABC is close to the true posterior and outperforms the neural density estimators for the **2D** model and performs equally well for the **6D** model. Finally, an implementation of the described methods in python is provided in the supplementary material.

References

Alsing, J., Charnock, T., Feeney, S., and Wandelt, B. Fast likelihood-free cosmology with neural density estimators and active learning // Monthly Notices of the Royal Astronomical Society. 07 2019. 488, 3. 4440–4458.

Amara, A. and Réfrégier, A. Optimal surveys for weak-lensing tomography // Monthly Notices of the Royal Astronomical Society. 10 2007. 381, 3. 1018–1026.

Bartelmann, M. and Schneider, P. Weak gravitational lensing // Physics Reports. 2001. 340, 4. 291–472.

Blum, M. and François, O. Non-linear regression models for Approximate Bayesian Computation // Statistics and Computing. 2010. 20. 63–73.

Blundell, C., Cornebise, J., Kavukcuoglu, K., and Wierstra, D. Weight Uncertainty in Neural Networks // Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37. 2015. 1613–1622. (ICML'15).

Charnock, T., Lavaux, G., and Wandelt, B. D. Automatic physical inference with information maximizing neural networks // Phys. Rev. D. Apr 2018. 97. 083004.

Chernozhukov, V., Chetverikov, D., Demirer, M., et al. Double/debiased machine learning for treatment and structural parameters // The Econometrics Journal. 01 2018. 21, 1. C1–C68.

Chisari, N. E., Alonso, D., Krause, E., et al. Core Cosmology Library: Precision Cosmological Predictions for LSST // The Astrophysical Journal Supplement Series. may 2019. 242, 1. 2.

Fan, Y. and Sisson, S. A. ABC Samplers // arXiv e-prints. II 2018. arXiv:1802.09650.

Fluri, J., Kacprzak, T., Lucchi, A., et al. Cosmological constraints with deep learning from KiDS-450 weak lensing maps // Phys. Rev. D. Sep 2019. 100. 063514.

de G. Matthews, A. G., van der Wilk, M., Nickson, T., et al. GPflow: A Gaussian Process Library using TensorFlow // Journal of Machine Learning Research. 2017. 18, 40. 1–6.

Gal, Y. Uncertainty in Deep Learning. 2016.

Gal, Y. and Ghahramani, Z. Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning // Proceedings of The 33rd International Conference on Machine Learning. 48. New York, New York, USA: PMLR, 20–22 Jun 2016. 1050–1059. (Proceedings of Machine Learning Research).

Gorski, K. M., Hivon, E., Banday, A. J., et al. HEALPix: A Framework for High-Resolution Discretization and Fast Analysis of Data Distributed on the Sphere // The Astrophysical Journal. apr 2005. 622, 2. 759–771.

Herbel, J., Kacprzak, T., Amara, A., et al. Fast point spread function modeling with deep learning // Journal of Cosmology and Astroparticle Physics. jul 2018. 2018, 07. 054–054.

Ishida, E., Vitenti, S., Penna-Lima, M., et al. cosmoabc: Likelihood-free inference via Population Monte Carlo Approximate Bayesian Computation // Astronomy and Computing. 2015. 13. 1 – 11.

Järvenpää, M., Gutmann, M., Vehtari, A., and Marttinen, P. Parallel Gaussian process surrogate Bayesian inference with noisy likelihood evaluations // arXiv e-prints. V 2019. arXiv:1905.01252.

Järvenpää, M., Gutmann, M. U., Pleska, A., et al. Efficient Acquisition Rules for Model-Based Approximate Bayesian Computation // Bayesian Anal. 06 2017. 14, 2. 595–622.

Jeffrey, N., Alsing, J., and Lanusse, F. Likelihood-free inference with neural compression of DES SV weak lensing map statistics // Monthly Notices of the Royal Astronomical Society. 11 2020. 501, 1. 954–969.

Kingma, D. P. and Ba, J. Adam: A Method for Stochastic Optimization // arXiv e-prints. XII 2014. arXiv:1412.6980.

Lange, K. L., Little, R. J. A., and Taylor, J. M. G. Robust Statistical Modeling Using the t Distribution // Journal of the American Statistical Association. 1989. 84, 408. 881–896.

Lavancier, F. and Rochet, P. A general procedure to combine estimators // arXiv e-prints. I 2014. arXiv:1401.6371.

Pellejero-Ibañez, M., Angulo, R. E., Aricó, G., et al. Cosmological parameter estimation via iterative emulation of likelihoods // Monthly Notices of the Royal Astronomical Society. 10 2020. 499, 4. 5257–5268.

Perraudin, N., Defferrard, M., Kacprzak, T., and Sgier, R. DeepSphere: Efficient spherical Convolutional Neural Network with HEALPix sampling for cosmological applications // Astronomy and Computing. 2018.

Pires, S., Starck, J. L., Amara, A., et al. Cosmological model discrimination from weak lensing data // Invisible Universe. 1241. VI 2010. 1118–1127. (American Institute of Physics Conference Series).

Potter, D., Stadel, J., and Teyssier, R. PKDGRAV3: beyond trillion particle cosmological simulations for the next era of galaxy surveys // Computational Astrophysics and Cosmology. V 2017. 4, 1. 2.

Schneider, P. Weak Gravitational Lensing // arXiv e-prints. IX 2005. astro–ph/0509252.

Sellentin, E. and Heavens, A. F. Parameter inference with estimated covariance matrices // Monthly Notices of the Royal Astronomical Society: Letters. 12 2015. 456, 1. L132–L136.

Sgier, R., Fluri, J., Herbel, J., et al. Fast Lightcones for Combined Cosmological Probes // arXiv e-prints. VII 2020. arXiv:2007.05735.

Teckentrup, A. L. Convergence of Gaussian Process Regression with Estimated Hyper-parameters and Applications in Bayesian Inverse Problems // arXiv e-prints. VIII 2019. arXiv:1909.00232.

Villaescusa-Navarro, F., Hahn, C., Massara, E., et al. The Quijote Simulations // The Astrophysical Journal Supplement Series. aug 2020. 250, 1. 2.

Villaescusa-Navarro, F., Wandelt, B. D., Anglés-Alcázar, D., et al. Neural networks as optimal estimators to marginalize over baryonic effects // arXiv e-prints. XI 2020. arXiv:2011.05992.

Zürcher, D., Fluri, J., Sgier, R., et al. Cosmological Forecast for non-Gaussian Statistics in large-scale weak Lensing Surveys // arXiv e-prints. VI 2020. arXiv:2006.12506.

A HIGH DIMENSIONAL SUMMARY STATISTICS

Optimizing the Cramér–Rao bound for summary statistics that have a higher dimensionality as the underlying model parameter $d_s > d_\theta$ is not possible with the technique presented in the main paper because the Jacobian is not square and therefore not invertible. However, one can nevertheless define an information maximizing loss, assuming that the Jacobian has a full rank d_θ, we can multiply the information inequality again with the Jacobian

$$\mathbf{J}^T \mathrm{Cov}_\theta(\hat{s})\mathbf{J} \geq \mathbf{J}^T\mathbf{J}\mathbf{I}(\theta)^{-1}\mathbf{J}^T\mathbf{J},$$

where we defined

$$\mathbf{J} = \frac{\partial \Psi_\theta(\hat{s})}{\partial \theta}.$$

Taking again the log-determinant function we arrive at the following inequality

$$\begin{aligned}-\log\det(\mathbf{I}(\theta)) \leq\ & \log\det(\mathbf{J}^T\mathrm{Cov}_\theta(\hat{s})\mathbf{J}) \\ & -2\log\det(\mathbf{J}^T\mathbf{J}).\end{aligned}$$

The right side of this inequality can be used as information maximizing loss for higher dimensional summary statistics and reduces to equation (7) for $d_s = d_\theta$.

B SECOND ORDER ESTIMATOR

A second order estimator also includes quadratic terms of the summary statistics

$$\hat{\theta}_i(x) = a_i + b_{ij}\hat{s}^j(x) + c_{ijk}\hat{s}^j(x)\hat{s}^k(x). \tag{17}$$

We start by taking the expected value of the individual terms of order expansion

$$\begin{aligned}\mathbb{E}_{p(x|\theta)}\left[a_i\right] &= a_i \\ \mathbb{E}_{p(x|\theta)}\left[b_{ij}\hat{s}^j(x)\right] &= b_{ij}\Psi^j_\theta(\hat{s}) \\ \mathbb{E}_{p(x|\theta)}\left[c_{ijk}\hat{s}^j(x)\hat{s}^k(x)\right] &= c_{ijk}\Psi^j_\theta(\hat{s})\Psi^k_\theta(\hat{s}) \\ &\quad + \mathrm{Tr}_{jl}\left[c_{ijk}\mathrm{Cov}(\theta)^k{}_l\right],\end{aligned}$$

where Tr_{jl} is the summation of all terms with $j = l$. Next, we expand all θ dependent terms around the fiducial parameter up to second order. We start with the expected value

$$\begin{aligned}\Psi_\theta(\hat{s})_i = \Psi_{\theta_\text{fid}}(\hat{s})_i & \\ &+ \frac{\partial \Psi_{\theta_\text{fid}}(\hat{s})_i}{\partial\theta_j}(\theta - \theta_\text{fid})^j \\ &+ \frac{\partial\partial \Psi_{\theta_\text{fid}}(\hat{s})_i}{\partial\theta_j\partial\theta_k}(\theta - \theta_\text{fid})^j(\theta - \theta_\text{fid})^k \\ &+ \mathcal{O}(\Delta\theta^3).\end{aligned}$$

We expand the covariance matrix in the same way

$$\begin{aligned}\mathrm{Cov}(\theta)_{ij} = \mathrm{Cov}(\theta_\text{fid})_{ij} & \\ &+ \frac{\partial \mathrm{Cov}(\theta_\text{fid})_{ij}}{\partial\theta_k}(\theta - \theta_\text{fid})^k \\ &+ \frac{\partial\partial \mathrm{Cov}(\theta_\text{fid})_{ij}}{\partial\theta_k\partial\theta_l}(\theta - \theta_\text{fid})^k(\theta - \theta_\text{fid})^l \\ &+ \mathcal{O}(\Delta\theta^3).\end{aligned}$$

Plugging these expressions into the expected value of equation (17) one obtains an expansion of the desired estimator in terms of the statistical moments of the summary statistics and their derivatives. Assuming that $d_s = d_\theta$ one can obtain $d_\theta^3 + d_\theta^2 + d_\theta$ linear equations, defining a_i, b_{ij} and c_{ijk}, by demanding that

$$\mathbb{E}_{p(x|\theta)}\left[\hat{\theta}_i(x)\right] = \theta + \mathcal{O}(\Delta\theta^3).$$

Solving the set of linear equations is straightforward, however, accurately estimating the higher order moments and their derivative requires a large amount of simulations. Also, it should be noted, that one has to generate additional simulations to obtain perturbations of the fiducial parameter such that one can estimate the higher order derivatives.

C COSMOLOGICAL MODEL

This appendix includes further details about the cosmological model.

C.1 FIDUCIAL PARAMETER AND PRIORS

For the training and setup of our model we chose the same fiducial parameters and perturbations as Villaescusa-Navarro et al. (2020), which are listed in table 2. The degeneracy parameter $S_8 = \sigma_8\sqrt{\Omega/0.3}$, corresponding to the width of the so called "cosmic banana" (e.g. shown in Figure 4) is not independent and captures the degeneracy of the two parameters Ω_M and σ_8.

Table 2: Overview of the chosen fiducial parameters, their perturbations and priors. The degeneracy parameter $S_8 = \sigma_8\sqrt{\Omega/0.3}$ captures the correlations between Ω_M and σ_8.

θ_i	$\theta_{\text{fid},i}$	$\Delta\theta_i$	Prior
Ω_M	0.3175	0.01	$[0.25, 0.4]$
Ω_b	0.049	0.002	$[0.03, 0.07]$
h	0.6711	0.02	$[0.5, 0.9]$
n_s	0.9624	0.02	$[0.8, 1.2]$
σ_8	0.834	0.015	$[0.72, 0.92]$
w_0	-1.0	0.05	$[-1.3, -0.7]$
S_8	0.858	N/A	$[0.79, 0.94]$

C.2 MOCK-SURVEY SETTINGS

We use spherical Gaussian random fields (GRF) in our cosmological model to mimic real convergence maps, as mentioned in the main text. A spherical GRF is completely determined by its spherical harmonic power spectrum (see appendix C.3 for more details), which means that the power spectrum is a sufficient statistic and that one can use power spectra to generate GRF. The power spectrum of convergence maps is itself a summary statistic often used for cosmological parameter inference and advanced codes exist that allow the fast generation of spectra (~ 1 second) for any set of cosmological parameters. In our model we use the publicly available code `CCL`[7] (The Core Cosmology Library) described in Chisari et al. (2019) to predict the spectra. The generation of convergence power spectra does not only require the cosmological parameters, but also observational parameters. The most important observational parameter is the galaxy distribution along the line-of-sight of the observer, i.e. how likely it is to observe a galaxy at a certain distance (redshift) from the observer, called the redshift distribution (see appendix C.2.1 for more details). We use the same total redshift distribution as Zürcher et al. (2020). Further, we consider a survey setting where the observed galaxies are assigned into four bins according to their redshift. This allows the creation of four convergence maps, all having a different redshift distribution. However, their features are correlated in a certain way, since they stem from the same universe. These correlation are defined by their cross-spectra, which can also be predicted using `CCL`.

C.2.1 Redshift Distribution

The redshift distribution of a weak lensing survey is the probability of observing a galaxy at a given distance from the observer (redshift). For our model we use the redshift distribution described in Zürcher et al. (2020) which has an analytic formula

$$n(z) \propto z^{1.5} \exp\left(-\left(\frac{z}{0.31}\right)^{1.1}\right),$$

and is a realistic redshift distribution for a stage-III cosmological survey. The normalization of the distribution depends on the redshift boundaries of the survey (maximum observed distance). For our model we chose $z_{\text{min}} = 0.0$ (position of the observer) and $z_{\text{max}} = 3.5$. Redshift distributions are normalized like any probability distribution by the condition

$$\int_{z_{\text{min}}}^{z_{\text{max}}} n(z) \mathrm{d}z = 1.$$

As mentioned previously, we consider a survey where the galaxies are assigned to one of four bins according to their observed redshift. Each bin should have approximately the same number of galaxies. Optimally, one could measure the redshift of a galaxy exactly, such that redshift distributions of the individual bins follow the same redshift distribution as the survey truncated with different z_{min} and z_{max}. However, in reality redshift measurements are prone to errors such that the individual redshift distributions broaden. We model this measurement error following Amara, Réfrégier (2007) assuming that the standard error of a redshift measurement is given by $\sigma_z = \delta(1+z)$, with $\delta = 0.01$. The resulting redshift distributions of the individual bins are shown in Figure 5 and do not have a closed form expression.

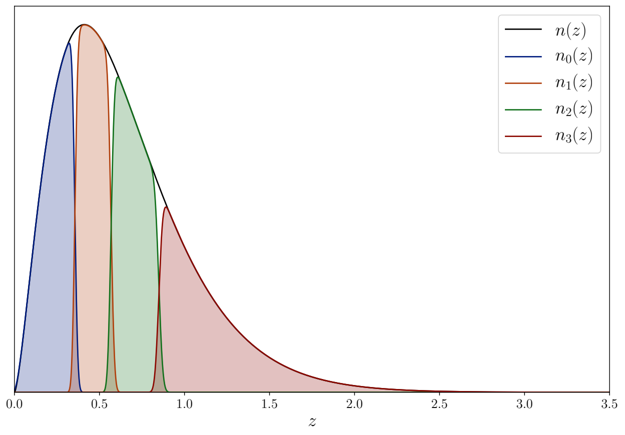


Figure 5: Redshift distributions of the individual bins and the total redshift distribution of the considered mock survey.

C.2.2 Observational Noise

The measured data in realistic cosmological surveys contains a substantial amount of noise. The observational noise can be modeled as Gaussian random noise (Pires et al., 2010) that mainly depends on the number of observed galaxies. The distribution is centered at zero and its standard deviation is given by

$$\sigma_{\text{noise}} = \frac{\sigma_e}{\sqrt{An}},$$

where $\sigma_e = 0.3$ is a typically measured ellipticity dispersion, A is the area of a given pixel and n is the galaxy number density. We add observational noise to all our simulations with a chosen a galaxy density of $n = 12$ galaxies/arcmin^2 which translates to 3 galaxies/arcmin^2 per redshift bin. Further, it can be shown that the power spectrum of a pure (pixelized) noise map is constant

$$\mathcal{C}_{\text{noise}} = \frac{f_{\text{sky}}}{n_{\text{pix}}} 4\pi\sigma_{\text{noise}}, \tag{18}$$

where f_{sky} corresponds to the fraction of the sky that is covered by the survey and n_{pix} to the number of pixels required to cover the whole sphere.

[7]https://github.com/LSSTDESC/CCL

C.3 GAUSSIAN RANDOM FIELDS ON THE SPHERE

Any real function on the sphere $f : S^2 \to \mathbb{R}$ can be represented as a series

$$f(\theta, \phi) = \sum_{\ell=0}^{\infty} \sum_{m=-\ell}^{\ell} \tilde{a}_{\ell m} \tilde{Y}_{\ell m}(\theta, \phi),$$

where the $\tilde{a}_{\ell m} \in \mathbb{R}$ are the real spherical harmonic coefficients and the $\tilde{Y}_{\ell m}(\theta, \phi)$ are the real spherical harmonics which can be expressed in terms of the complex spherical harmonics

$$\tilde{Y}_{\ell m}(\theta, \phi) = \begin{cases} \sqrt{2}\mathcal{R}(Y_{\ell m}(\theta, \phi)) & m > 0 \\ Y_{\ell 0} & m = 0 \\ \sqrt{2}\mathcal{I}(Y_{\ell |m|}(\theta, \phi)) & m < 0 \end{cases}$$

A homogeneous and isotropic Gaussian random field (GRF) on the sphere can be defined as a function where the spherical harmonic coefficients are uncorrelated Gaussian random variables with zero mean

$$\tilde{a}_{\ell m} \sim \mathcal{N}(0, \mathcal{C}_\ell),$$

where the variance is given by the (angular) power spectrum $\mathcal{C}_\ell$. The GRF is therefore fully defined by its power spectrum and given a realization of a GRF we can construct an unbiased estimate of the underlying power spectrum via

$$\hat{\mathcal{C}}_\ell = \frac{1}{2\ell+1} \sum_{m=-\ell}^{\ell} \tilde{a}_{\ell m}^2. \tag{19}$$

It is also useful to define the cross-spectra between two GRF T_1 and T_2 that might stem from different power spectra, which is usually done in terms if the complex spherical harmonic coefficients

$$\mathcal{C}_\ell^{12} = E\left[\mathcal{R}(a_{\ell m}^1)\mathcal{R}(a_{\ell m}^2) + \mathcal{I}(a_{\ell m}^1)\mathcal{I}(a_{\ell m}^2)\right].$$

The cross spectrum can be estimated in the same way as the normal power spectrum and it easy to see that if $\mathcal{C}_\ell^1 = \mathcal{C}_\ell^2$, then, the cross spectrum reduces to the normal power spectrum $\mathcal{C}_\ell^{12} = \mathcal{C}_\ell^1 = \mathcal{C}_\ell^2$.

C.3.1 Survey Geometry

The $\mathcal{C}_\ell$ estimation described by equation (19) is only unbiased if the spherical harmonic coefficients were calculated from a full sky map. Realistic surveys, however, do not observe the entire night sky and often have complicated survey masks (see Figure 2). Such a survey mask leads to a bias in the estimation of the power spectrum called mode-coupling. It is possible to remove this bias with the Wigner matrix formalism. It turns out that the expected value of the estimated masked sky spectrum $\hat{\mathcal{C}}_{\ell,\mathrm{masked}}$ is connected to the underlying power spectrum via a linear transformation

$$E\left[\hat{\mathcal{C}}_{\ell,\mathrm{masked}}\right] = \sum_{\ell'} W_{\ell\ell'} \mathcal{C}_{\ell'}, \tag{20}$$

where $W_{\ell\ell'}$ is the Wigner matrix. The Wigner matrix is completely defined by the survey mask and its calculation is described in appendix D of Sgier et al. (2020). For continuous GRF it is sufficient to have a non-vanishing area to guarantee that the Wigner matrix is invertible. This holds true most of the time even for pixelized GRF. However, due to numerical stability it is common to apply the Wigner matrix to the underlying power spectrum as oposed to reconstruct an unbiased estimator from the masked power spectrum (see appendix C.5).

C.3.2 Gaussian Random Field Generation

Generating a GRF with a single power spectrum is fairly straight forward. However, generating different GRF with given cross spectra requires a more advanced generation procedure. The necessary algorithm is described in appendix C of Sgier et al. (2020).

Finally, to actually generate the spherical GRF we use the Hierarchical Equal Area iso-Latitude Pixelization tool (`HEALPix`) described in Gorski et al. (2005), using `CCL` to calculate the input power- and cross-spectra of convergence maps given a set of cosmological parameters and the necessary redshift distributions. Our cosmological model used `CCL` to predict spectra up to $\ell_{\max} = 1000$. In `Healpix` the resolution of a map is given by the `nside` parameter, which we fixed for all experiments to `nside` $= 512$, leading to $\sim 400,000$ relevant pixel (see Figure 2) for each of the four redshift bins.

C.4 GRAPH CONVOLUTIONAL NEURAL NETWORK

This section contains details about the graph convolutional neural network (GCNN) used for the experiments. We provide more information about the architecture and training parameter.

C.4.1 Architecture

The chosen survey configuration of the model makes it impossible to treat the data as flat images. There exist multiple approaches to deal with spherical data. We chose to use `DeepSphere` (Perraudin et al., 2018) that treats the pixels of the the generated GRF as nodes in a graph. The distances between the nodes are given by the spherical geometry of the maps. The graph convolutional filters in `DeepSphere` are expressed in terms of Chebyshev polynomials and applied to the data using the graph Laplacian. Additionally it uses the hierarchical pixelization approach of `HealPix` to downsample the data while maintaing is spherical geometry. This downsampling procedure is refered to as pseudo convolutions and displayed in Figure 6. To use this feature, however, one has to add a padding to the input of the GCNN such that a valid downsampling is possible (see edges of the patch shown in Figure 2). The architecture of the used network is shown in table 3. All presented layer use the ReLU activation function. The architecture of the residual layers is shown as a flowchart in Figure 7. All graph convolutions use Chebyshev polynomials with degree $K = 5$. Further details and a concrete implementation can be found in the supplementary material will be published on `github`.

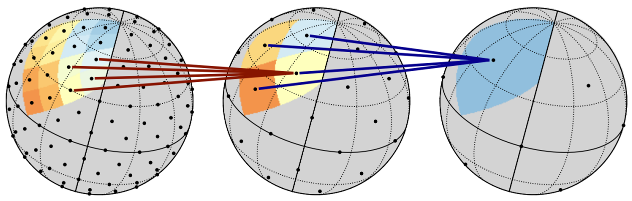


Figure 6: Downsampling procedure using the hierarchical pixelization approach of `HealPix`. Four pixels are combined into one. The weights of the combination can be learned during the training of the network. Figure taken from Perraudin et al. (2018).

Table 3: Architecture of the used GCNN. We report layer type, output shape (N_b being the batch size) and number of trainable parameters. The residual layer (see Figure 7) is repeated ten times. The output shape depends on the model. For the **2D** model we have $N_{\text{out}} = 2$ and $N_{\text{out}} = 6$ for the **6D** model.

Layer Type	**Output Shape**	**# Parameter**
Input	(N_b, 428032, 4)	0
Pseudo Convolution	(N_b, 107008, 16)	272
Pseudo Convolution	(N_b, 26752, 32)	2080
Pseudo Convolution	(N_b, 6688, 64)	8256
Pseudo Convolution	(N_b, 1672, 128)	32896
Layer-Normalization	(N_b, 1672, 128)	256
Residual Layer	(N_b, 1672, 128)	170528
	⋮	
Residual Layer	(N_b, 1672, 128)	170528
Pseudo Convolution	(N_b, 418, 128)	32896
Flatten	(N_b, 53504)	0
Layer-Normalization	(N_b, 53504)	107008
Fully Connected	(N_b, N_{out})	53504N_{out}

C.4.2 Training

All graph convolutional neural networks presented in this work were trained for 100,000 steps with batch size of $128(d_\theta + 1)$ distributed among 32 GPUs on the cluster computer Piz Daint[8]. The weights were optimized using the Adam optimizer (Kingma, Ba, 2014) with initial learning rate of 0.0001 and moments $\beta_1 = 0.9$ and $\beta_2 = 0.999$. Additionally, to avoid large weight updates, we applied global gradient clipping, reducing the global norm of the gradients in a training step to 5.0 if necessary.

[8] `https://www.cscs.ch/computers/piz-daint/`

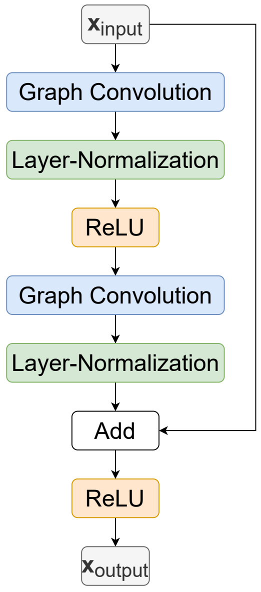


Figure 7: Architecture of the used residual layer.

C.5 GROUND TRUTH

The advantage of this model is that a sufficient summary statistics is known from the beginning. All maps are generated from the power spectra calculated with `CCL`. Let $\mathcal{C}(\theta)$ be such a spectrum, already corrected for the survey geometry (see appendix C.3.1). Given a spectrum $\mathcal{C}_{\text{obs}}$ measured from an observation x_{obs}, the posterior distribution is given by (Sellentin, Heavens, 2015)

$$p(\theta|\mathcal{C}(\theta), \mathcal{C}_{\text{obs}}, S) \propto \frac{\det(S)^{-\frac{1}{2}}}{\left[1 + \frac{Q}{N-1}\right]^{\frac{N}{2}}}, \tag{21}$$

where Q is defined as

$$Q = (\mathcal{C}(\theta) + \mathcal{C}_{\text{noise}} - \mathcal{C}_{\text{obs}})^T S^{-1} (\mathcal{C}(\theta) + \mathcal{C}_{\text{noise}} - \mathcal{C}_{\text{obs}}),$$

the covariance matrix S is estimated from the $N = 10,000$ simulations of the fiducial parameter and $\mathcal{C}_{\text{noise}}$ is the noise contribution, which does not depend on the cosmological parameters. We make the common assumption that the covariance matrix S does not depend on the cosmological parameters θ. For parameter inference, one can directly integrate the posterior distribution with a MCMC. For parameter estimation one can use the likelihood to calculate the Fisher information matrix, which in turn gives the smallest possible variance of any estimator. It can be shown (Lange et al.,

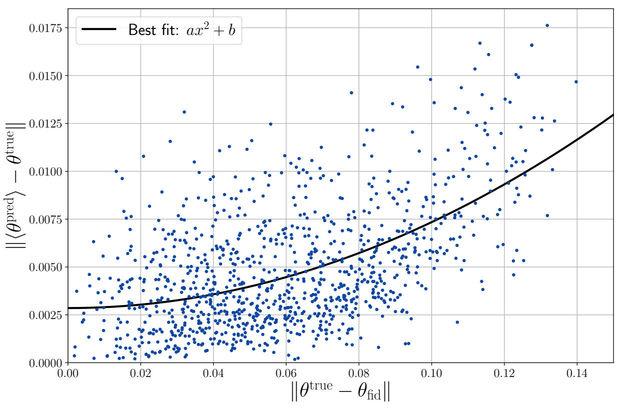


Figure 8: Bias of the mean predictions of Figure 3 plotted against the distance of the underlying parameter to the fiducial parameter.

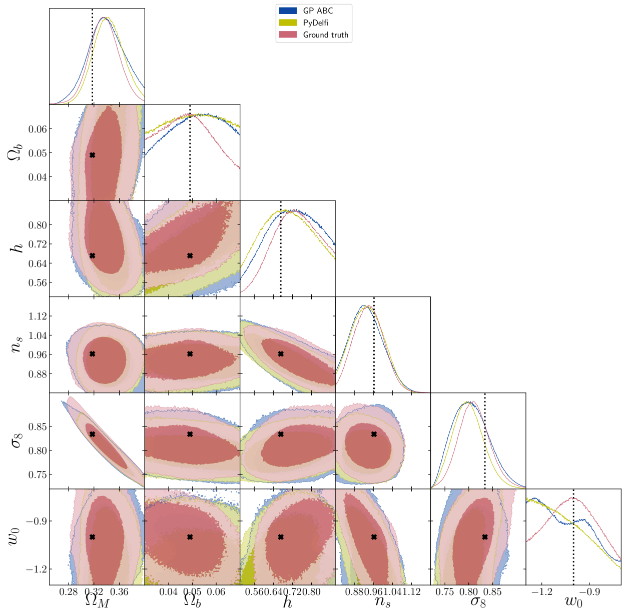


Figure 9: The posterior distributions of our **6D** model. The fiducial parameter is indicated with a black cross in the two-dimensional marginal distributions and with vertical dotted lines in the one-dimensional marginal distributions.

1989), that the Fisher information matrix of the likelihood defined in equation (21) is given by

$$\mathbf{I}_{ij}(\theta)=\frac{N}{N+2}\frac{\partial\mathcal{C}(\theta)}{\partial\theta_i}^T S^{-1}\frac{\partial\mathcal{C}(\theta)}{\partial\theta_j}. \tag{22}$$

Leading to a lower bound on the variance of any unbiased estimator Var $\left[\hat{\theta}_i\right]\geq(\mathbf{I}^{-1})_{ii}$. We calculate the lower bound for the fiducial parameters using the perturbed spectra to numerical approximate the derivatives.

C.6 BIAS OF THE FIRST ORDER ESTIMATOR

The bias of the mean predictions presented in Figure 3 is shown in Figure 8. The bias is approximately quadratic in the distance to the fiducial parameter, as expected from a first order estimator. The scatter is caused by the finite sample number used to calculate the means and the first order estimator.

C.7 MARGINAL DISTRIBUTIONS OF THE 6D MODEL

We show a plot of the marginal distribution for our **6D** model obtained with GP ABC and neural density estimators in Figure 9. The marignal distribution from both methods are in excellent agreement with the true posterior.

C.8 IMPACT OF THE SCALE PARAMETER

The estimation of the ABC likelihood requires the setting of a scale parameter h that is used inside the kernel function (e.g. see equation (13)). This parameter essentially impacts the ABC posterior as a smoothing factor. A parameter that is chosen too large will lead to a posterior distribution that will be too broad, i.e. smoothed out like a kernel density estimation with inappropriately large bandwidth. If the scale is chosen to small, the posterior estimates will be very noisy because the kernel will map the distance of most simulations to the observation to zero or almost zero. Nevertheless should the posterior distribution of the Gaussian process not be too sensitve with respect to the scale parameter. We show the posterior of the **2D** model for differently chosen scale parameters in Figure 10. As expected, the contours become broader as the scale parameter increases. All of the generated posterior distributions are consistent with the ground truth.

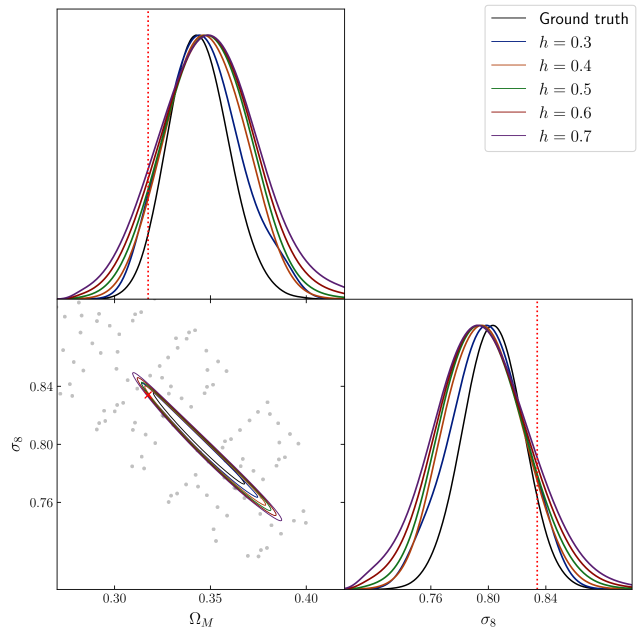


Figure 10: The 68% confidence contours of the **2D** model for different scale parameter h compared to the ground truth.