MNRAS **000**, 1–20 (2017) Preprint 8 December 2019 Compiled using MNRAS LATEX style file v3.0

Fast likelihood-free cosmology with neural density estimators and active learning

Justin Alsing,[1,2,3]⋆ Tom Charnock[4], Stephen Feeney[2] and Benjamin Wandelt[2,5]
[1]*Oskar Klein Centre for Cosmoparticle Physics, Stockholm University, Stockholm SE-106 91, Sweden*
[2]*Center for Computational Astrophysics, Flatiron Institute, 162 5th Ave, New York City, NY 10010, USA*
[3]*Imperial Centre for Inference and Cosmology, Department of Physics, Imperial College London, Blackett Laboratory, Prince Consort Road, London SW7 2AZ, UK*
[4]*Sorbonne Université, CNRS, UMR 7095, Institut d'Astrophysique de Paris, 98 bis bd Arago, 75014 Paris, France*
[5]*Sorbonne Université, Institut Lagrange de Paris (ILP), 98 bis boulevard Arago, F-75014 Paris, France*

Accepted XXX. Received YYY; in original form ZZZ

ABSTRACT
Likelihood-free inference provides a framework for performing rigorous Bayesian inference using only forward simulations, properly accounting for all physical and observational effects that can be successfully included in the simulations. The key challenge for likelihood-free applications in cosmology, where simulation is typically expensive, is developing methods that can achieve high-fidelity posterior inference with as few simulations as possible. Density-estimation likelihood-free inference (DELFI) methods turn inference into a density estimation task on a set of simulated data-parameter pairs, and give orders of magnitude improvements over traditional Approximate Bayesian Computation approaches to likelihood-free inference. In this paper we use neural density estimators (NDEs) to learn the likelihood function from a set of simulated datasets, with active learning to adaptively acquire simulations in the most relevant regions of parameter space on-the-fly. We demonstrate the approach on a number of cosmological case studies, showing that for typical problems high-fidelity posterior inference can be achieved with just $O(10^3)$ simulations or fewer. In addition to enabling efficient simulation-based inference, for simple problems where the form of the likelihood is known, DELFI offers a fast alternative to MCMC sampling, giving orders of magnitude speed-up in some cases. Finally, we introduce PYDELFI – a flexible public implementation of DELFI with NDEs and active learning – available at `https://github.com/justinalsing/pydelfi`.

Key words: data analysis: methods

1 INTRODUCTION

Likelihood-free inference (LFI) is emerging as a new paradigm for performing Bayesian inference under very complex generative models, using only forward simulations. This approach has great appeal for cosmological data analysis, since all effects that can be incorporated into forward simulations can be accounted for exactly in the inference pipeline, without having to resort to approximate calibrations and likelihood assumptions that may lead to biased inferences and/or mis-stated uncertainties.

The main challenge for likelihood-free applications in cosmology, where simulation is expensive, has been developing methods that can give high-fidelity posterior inference from a feasibly small number of forward simulations. Traditional approaches to likelihood-free inference have been based on Approximate Bayesian Computation (ABC), which involves (variants on) drawing parameters from some proposal, simulating mock data, and accepting/rejecting the parameters based on whether the simulated data fall within some ϵ-ball around the observed data (see Lintusaari et al. 2017 for a review). Whilst ABC has enabled a number of applications in astronomy and cosmology (Schafer & Freeman 2012; Cameron & Pettitt 2012; Weyant et al. 2013; Robin et al. 2014; Lin & Kilbinger 2015; Hahn et al. 2017; Kacprzak et al. 2017; Carassou et al. 2017; Davies et al. 2017; Ishida et al. 2015; Akeret et al. 2015; Jennings et al. 2016), ABC methods generally require a vast number of simulations, scaling exponentially with the number of model parameters, making them unfeasible when simulation is even modestly expensive.

Density-estimation likelihood-free inference (DELFI; Bonassi et al. 2011; Fan et al. 2013; Papamakarios & Murray 2016; Lueckmann et al. 2017; Papamakarios et al. 2018; Lueckmann et al. 2018; Alsing et al. 2018b) aims to train a flexible density estimator for the target posterior from a set of simulated data-parameter pairs, and can yield high-fidelity posterior inference from orders-of-magnitude

⋆ E-mail: justin.alsing@fysik.su.se

fewer simulations than traditional ABC-based methods. In this paper we introduce PYDELFI – a general purpose implementation of density-estimation likelihood-free inference using neural density estimators (NDEs) to learn the sampling distribution of the data as a function of the model parameters, employing active learning to adaptively run simulations in the most relevant regions of parameter space on-the-fly (based on Papamakarios et al. 2018; Lueckmann et al. 2018). We show that with NDEs and active learning, high-fidelity posteriors can be obtained for typical cosmological inference tasks from just a few thousand forward simulations. This opens up new possibilities for likelihood-free applications in cosmology.

The structure of this paper is as follows: In §2 we review density-estimation likelihood-free inference methods using neural density estimators and adaptive acquisition of simulations with active learning. In §3 we review data compression schemes for accelerating likelihood-free inference; approximate score-compression, deep network parameter estimators, and information maximizing neural networks (IMNN; Charnock et al. 2018). In §4 we introduce PYDELFI, briefly outlining the implementation details and features of the code. Tutorials and documentation for the code can be found at `https://github.com/justinalsing/pydelfi`. In §5–7 we validate and demonstrate the performance of the PYDELFI approach on some simple case studies from cosmology: analysis of the JLA supernova data (Betoule et al. 2014) (against a known likelihood for validation), tomographic cosmic shear pseudo-C_ℓ analysis, and inference of the HI ionization rate around $z \sim 6$ from high-redshift Lyman-α forests. We conclude with some discussion in §8.

2 DENSITY ESTIMATION LIKELIHOOD-FREE INFERENCE

In this section we provide a pedagogical review of density-estimation likelihood-free inference (§2.2) with neural density estimators (§2.3-2.4) and active learning to adaptively acquire simulations on-the-fly (§2.6). The methodology described in this section is based on Papamakarios & Murray (2016), Papamakarios et al. (2018), Lueckmann et al. (2018) and Alsing et al. (2018b).

2.1 Bayesian parameter inference with and without likelihoods

We begin with a brief review of Bayesian parameter inference to introduce the semantics and notation used throughout the rest of the paper.

The central object of any parameter inference task – whether Bayesian or frequentist – is the *sampling distribution* of hypothetical data $\mathbf{d}$ given the model $\mathcal{M}$ and parameters $\boldsymbol{\theta}$, $p(\mathbf{d}|\boldsymbol{\theta}, \mathcal{M})$. Here, the generative model $\mathcal{M}$ encodes some specification of how the data were generated in nature, covering the physics associated with both the underlying signal (eg., the cosmological and astrophysical processes governing the observable) and measurement process (eg., observational sampling, instrumental noise and artefacts, etc).

With the sampling distribution $p(\mathbf{d}|\boldsymbol{\theta}, \mathcal{M})$ specified, Bayesian parameter inference proceeds by fixing $\mathbf{d}$ to the values of the *observed data* $\mathbf{d}_o$, and targets the posterior density $p(\boldsymbol{\theta}|\mathbf{d}_o, \mathcal{M})$ (using Bayes' theorem):

$$p(\boldsymbol{\theta}|\mathbf{d}_o, \mathcal{M}) = \frac{p(\mathbf{d}_o|\boldsymbol{\theta}, \mathcal{M})\, p(\boldsymbol{\theta}|\mathcal{M})}{p(\mathbf{d}_o|\mathcal{M})}. \quad (1)$$

Here, $p(\mathbf{d}_o|\boldsymbol{\theta}, \mathcal{M})$ is the *likelihood function* (the sampling distribution evaluated at the observed data, as a function of the parameters), $p(\boldsymbol{\theta}|\mathcal{M})$ is the *prior distribution* encoding any information/beliefs about the parameter values prior to making observations $\mathbf{d}_o$, and the *evidence* $p(\mathbf{d}_o|\mathcal{M})$ is just a normalization constant for the purposes of parameter inference (but is useful for model selection tasks).

In the standard Bayesian inference paradigm, the posterior density is typically then explored using Markov Chain Monte Carlo (MCMC) sampling, variational inference, or other Bayesian computation methods (see eg., Gelman et al. 2013 for a review). These "traditional" methods ubiquitously rely on being able to compute the likelihood function $p(\mathbf{d}_o|\boldsymbol{\theta}, \mathcal{M})$, for a given $\mathbf{d}_o$ and $\boldsymbol{\theta}$.

However, for complicated generative models, the sampling distribution of the data – and hence the likelihood function – may be intractable in practice. In these situations, even though the likelihood is intractable, we are still able to forward simulate[1], ie., generate realizations of the observations $\mathbf{d}$ given model parameters $\boldsymbol{\theta}$. *Likelihood-free inference* methods rely only on our ability to perform forward simulations in order to explore the posterior, bypassing the need to ever compute the likelihood function directly. This class of methods has great appeal, since it allows us to perform inference under very complex forward models, unfettered by the need to be able to write down a tractable likelihood function.

As we will see in §3, likelihood-free methods typically benefit from compressing large data vectors down to a small number of informative *summary statistics* $\mathbf{t}$. Throughout the text we use $\mathbf{d}$ to denote an uncompressed data vector and $\mathbf{t}$ to denote a vector of compressed data summaries. We write as though data are always compressed to some summaries $\mathbf{t}$ for likelihood-free inference, although this need not always be the case if the data are already low-dimensional (relative to the number of simulations that can be performed – see §3 for discussion). We often use "data" and "data summaries" interchangeably in the text, being explicit where necessary to avoid confusion. Observed rather than hypothetical data or summaries are indicated by $\mathbf{d}_o$ and $\mathbf{t}_o$ respectively.

2.2 DELFI, three ways

Density-estimation likelihood free inference turns inference into a density estimation task on a set of simulated parameter-data (summary) pairs $\{\boldsymbol{\theta}, \mathbf{t}\}$. There are principally three ways to approach this density-estimation inference task (shown schematically in Figure 1):

(1) Fit a model to the joint density $p(\boldsymbol{\theta}, \mathbf{t})$, then obtain the posterior by evaluating the joint density at the observed data $\mathbf{t}_o$, $p(\boldsymbol{\theta}|\mathbf{t}) \propto p(\boldsymbol{\theta}, \mathbf{t} = \mathbf{t}_o)$ (Alsing et al. 2018b).

(2) Fit a model to the conditional density $p(\boldsymbol{\theta}|\mathbf{t})$, then obtain the posterior by evaluating at the observed data $\mathbf{t}_o$. (Papamakarios & Murray 2016; Lueckmann et al. 2017).

(3) Fit a model to the conditional density $p(\mathbf{t}|\boldsymbol{\theta})$, obtain the likelihood by evaluating at the observed data $\mathbf{t}_o$, and multiply by the prior to get the posterior $p(\boldsymbol{\theta}|\mathbf{t}_o) \propto p(\mathbf{t}_o|\boldsymbol{\theta}) \times p(\boldsymbol{\theta})$ (Papamakarios et al. 2018; Lueckmann et al. 2018).

Option 3 – learning the sampling distribution of the data as a function of the parameters – has some key advantages over the other two approaches. Firstly, by learning the sampling distribution of the data *conditional* on the parameters, it does not matter how

[1] Note that forward simulating a data realization $\mathbf{d}$ given parameters $\boldsymbol{\theta}$ under some model $\mathcal{M}$ is equivalent to drawing a sample from the implicit sampling distribution $p(\mathbf{d}|\boldsymbol{\theta}, \mathcal{M})$. This can be done even when the sampling density itself is intractable.

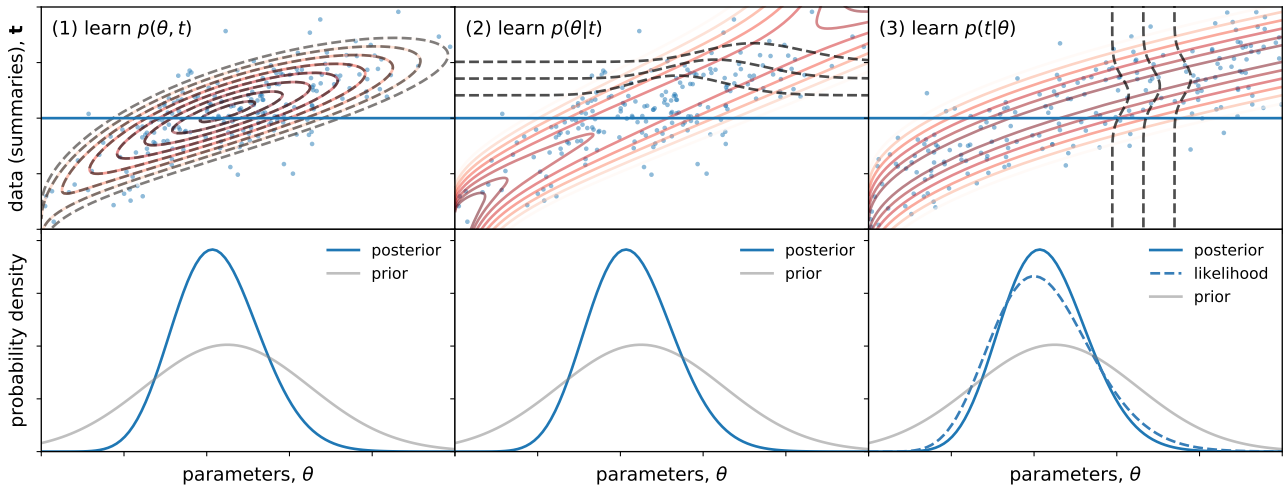

Figure 1. Schematic for the three ways of performing density-estimation likelihood-free inference from a set of simulated data (summary) parameter pairs $\{\mathbf{t}, \boldsymbol{\theta}\}$: (1) learn a flexible parametric model for the joint density $p(\boldsymbol{\theta}, \mathbf{t})$, (2) learn a flexible parametric model for the conditional density $p(\boldsymbol{\theta}|\mathbf{t})$ (as a function of $\mathbf{t}$), (3) learn a flexible parametric model for the conditional $p(\mathbf{t}|\boldsymbol{\theta})$ (as a function of $\boldsymbol{\theta}$). In each case, the goal is to learn the (conditional) density in the relevant region of parameter space, and take a slice at the observed data (summaries) to yield the target posterior or likelihood.

the parameters for running forward simulations were chosen. This gives complete freedom as to how simulations are acquired, so any schemes for adaptively acquiring simulations in the most relevant parts of parameter space can be employed without complication (see §2.6). In contrast, for options 1 and 2, parameters must either be drawn from the prior, or else drawn from some proposal density $q(\boldsymbol{\theta})$ and the resulting learned target density subsequently re-weighted by $p(\boldsymbol{\theta})/q(\boldsymbol{\theta})$. This re-weighting step can result in instabilities during training, or high variance importance weights (and low effective sample sizes) after sampling, or both (see Papamakarios et al. 2018 for discussion). By learning the likelihood function rather than the posterior, it is also more straightforward to explore different prior assumptions a posteriori without similar importance re-weighting issues.

Secondly, for applications where data are compressed to a small number of highly informative summaries, these will often tend to be asymptotically Gaussian, so for many problems the sampling distribution of the data summaries may be well-captured by a relatively simple density model (eg., a Gaussian mixture with a modest number of mixture components, or similar), even when the posterior (option 2) or joint distribution (option 1) is complicated.

In light of these considerations, we suggest implementing DELFI by learning the sampling distribution of the data (summaries) as a function of the model parameters as a sensible default approach[2]. With this choice made, DELFI can be broadly summarized as follows:

(i) Run simulations at different parameter values $\boldsymbol{\theta}$ to obtain simulated parameter-data pairs $\{\boldsymbol{\theta}, \mathbf{t}\}$,

(ii) Fit a parametric conditional density estimator $p(\mathbf{t}|\boldsymbol{\theta}; \mathbf{w})$ to the simulations $\{\boldsymbol{\theta}, \mathbf{t}\}$,

(iii) Evaluate the estimated conditional density at the observed data $\mathbf{t}_o$ to obtain the (learned) likelihood function $p(\mathbf{t}_o|\boldsymbol{\theta}; \mathbf{w})$.

An efficient algorithm for performing DELFI must then address three key questions:

(i) How do we parameterize the conditional density estimator $p(\mathbf{t}|\boldsymbol{\theta}; \mathbf{w})$ in a sensible way?

(ii) How do we run simulations in the most relevant parts of parameter space for the ultimate target, $p(\mathbf{t}_o|\boldsymbol{\theta}; \mathbf{w})$, to best use the available resources?

(iii) If the uncompressed data vector $\mathbf{d}$ is high-dimensional, how can we compress it effectively to some small set of informative summaries $\mathbf{d} \to \mathbf{t}$ to reduce the dimensionality of the density-estimation task, and hence reduce the number of simulations required?

In this paper we use neural density estimators (NDEs) as a flexible and efficient conditional density estimation framework for DELFI (based on Papamakarios & Murray 2016; Papamakarios et al. 2018; Lueckmann et al. 2018), employing ensembles of networks (with different initializations and architectures) to give robustness against small training sets and architecture choice. We give an overview of NDEs and network ensembles in §2.3 and 2.4.

For efficient acquisition of simulations, we use active learning, allowing the NDEs to call the simulator to run new simulations on-the-fly, based on the current likelihood-surface approximation. We discuss active learning strategies in §2.6.

We review key data compression schemes for accelerating DELFI in §3 (approximate-score compression, and deep network compression schemes).

2.3 Neural density estimators

Neural density estimators (NDEs) provide flexible parametric models for conditional probability densities $p(\mathbf{t}|\boldsymbol{\theta}; \mathbf{w})$, parameterized by neural networks with weights $\mathbf{w}$, which can be trained on a set of simulated data-parameter pairs $\{\mathbf{t}, \boldsymbol{\theta}\}$.

[2] However, we note that option 1 comes with its own unique advantage in that it provides an analytical estimate of the Bayesian evidence for free, provided an analytically integratable joint-density parameterization such as a Gaussian mixture is used (Alsing et al. 2018b). This may be preferred when the evidence is the primary target. Note the evidence estimated this way will be with respect to the compressed summaries, rather than the un-compressed data vector.

In this section we review two classes of NDEs that have proven useful in the context of likelihood-free inference: mixture density networks (MDNs; Bishop 1994) and masked autoregressive flows (MAFs; Papamakarios et al. 2017). Note this section assumes basic background knowledge of neural networks – see eg., Bishop (2006) for a comprehensive review.

2.3.1 Mixture Density Networks (MDN)

Mixture density networks constitute a class of models for the conditional density $p(\mathbf{t}|\boldsymbol{\theta};\mathbf{w})$ where the distribution for $\mathbf{t}$ at any given $\boldsymbol{\theta}$ is given by a mixture model, and the relative weights and properties of the mixture components are all free functions of $\boldsymbol{\theta}$, parameterized by a neural network with weights $\mathbf{w}$. For example, a Gaussian mixture density network[3] defines the following conditional density estimator,

$$p(\mathbf{t}|\boldsymbol{\theta};\mathbf{w}) = \sum_{k=1}^{n_c} r_k(\boldsymbol{\theta};\mathbf{w})\,\mathcal{N}\left[\mathbf{t}\,|\,\boldsymbol{\mu}_k(\boldsymbol{\theta};\mathbf{w}), \mathbf{C}_k \equiv \boldsymbol{\Sigma}_k(\boldsymbol{\theta};\mathbf{w})\boldsymbol{\Sigma}_k^T(\boldsymbol{\theta};\mathbf{w})\right], \tag{2}$$

ie., an n_c component Gaussian mixture model where the component weights $\{r_k(\boldsymbol{\theta};\mathbf{w})\}$, means $\{\boldsymbol{\mu}_k(\boldsymbol{\theta};\mathbf{w})\}$, and covariance factors[4] $\{\boldsymbol{\Sigma}_k(\boldsymbol{\theta};\mathbf{w})\}$ are all functions of $\boldsymbol{\theta}$ parameterized by a neural network with weights $\mathbf{w}$.

The MDN model is shown schematically in Figure 2; the network takes in parameters $\boldsymbol{\theta}$ and outputs the means, weights and covariances of the mixture model for $p(\mathbf{t}|\boldsymbol{\theta})$ corresponding to that input $\boldsymbol{\theta}$. The MDN network architecture typically has a number of intermediate dense hidden layers with some non-linear activation function (eg., tanh). In the output layer, the output nodes corresponding to the means have linear activations, as do the off-diagonal elements of the covariance matrices, whilst the diagonal covariance elements are passed through an exponential activation to ensure positive definiteness, and the mixture component weights are passed through a softmax activation[5] to ensure they are positive and sum to unity.

Note that a mixture density network parameterization of $p(\mathbf{t}|\boldsymbol{\theta})$ with a single Gaussian component defines a Gaussian likelihood where the mean and covariance are functions of the parameters – a common approximate likelihood used in many cosmological data analysis problems. Adding additional components immediately results in a more flexible density estimator and hence likelihood assumptions; Gaussian mixtures can represent any smooth probability density (given enough components).

2.3.2 Masked Autoencoders for Density Estimation (MADE)

Any probability density can be factorized as a product of one-dimension conditionals via applications of the chain rule:

$$p(\mathbf{t}|\boldsymbol{\theta}) = \prod_{i=1}^{\dim(\mathbf{t})} p(t_i|\mathbf{t}_{1:i-1},\boldsymbol{\theta}). \tag{3}$$

Neural autoregressive density estimators construct parametric densities for this set of one-dimensional conditionals, where the parameters of each of the conditionals are parameterized as a neural network (Uria et al. 2016). For example, one could model each conditional $p(t_i|\mathbf{t}_{1:i-1},\boldsymbol{\theta})$ as a Gaussian whose mean and variance are free functions of $(\mathbf{t}_{1:i-1},\boldsymbol{\theta})$, parameterized by a neural network. Gaussian Masked Autoencoders for Density Estimation (MADEs; Germain et al. 2015), depicted in Figure 3, do precisely this: the means and variances of each conditional density are parameterized by the neural network, where crucially the weights of the neural network layers are masked in such a way that the output nodes for $p(t_i|\mathbf{t}_{1:i-1},\boldsymbol{\theta})$ only depend on $(\mathbf{t}_{1:i-1},\boldsymbol{\theta})$ (ie., the autoregressive property is preserved). See Germain et al. (2015) for details of how to construct the binary network weight mask. As with MDNs, the hidden layers of the MADE have some non-linear activation functions (eg., tanh), whilst the output nodes associated with the conditional means have linear activation, and the output nodes associated with the variances have exponential activations (ensuring positivity).

By learning the means and variances of the autoregressive conditionals, a MADE can be thought of as learning the transform of the random variate $\mathbf{t}$ back to the unit normal:

$$\begin{aligned}&\mathbf{t}|\boldsymbol{\theta} \rightarrow \mathbf{u}(\mathbf{t},\boldsymbol{\theta};\mathbf{w}) \sim \mathcal{N}(\mathbf{0},\mathbf{I}),\\ &t_i|\boldsymbol{\theta} \rightarrow u_i = (t_i - \mu_i(\mathbf{t}_{1:i-1},\boldsymbol{\theta};\mathbf{w}))/\sigma_i(\mathbf{t}_{1:i-1},\boldsymbol{\theta};\mathbf{w}),\end{aligned} \tag{4}$$

where $\mathbf{w}$ are the (masked) weights of the neural network. The parametric density estimator for a MADE is hence given by,

$$\begin{aligned}p(\mathbf{t}|\boldsymbol{\theta};\mathbf{w}) &= \prod_i p(t_i|\mathbf{t}_{1:i-1},\boldsymbol{\theta};\mathbf{w})\\ &= \mathcal{N}\left[\mathbf{u}(\mathbf{t},\boldsymbol{\theta};\mathbf{w})|\mathbf{0},\mathbf{I}\right] \times \left|\frac{\partial \mathbf{u}(\mathbf{t},\boldsymbol{\theta};\mathbf{w})}{\partial \mathbf{t}}\right|\\ &= \mathcal{N}\left[\mathbf{u}(\mathbf{t},\boldsymbol{\theta};\mathbf{w})|\mathbf{0},\mathbf{I}\right] \times \prod_{i=1}^{\dim(\mathbf{t})} \sigma_i(\mathbf{t},\boldsymbol{\theta};\mathbf{w}).\end{aligned} \tag{5}$$

MADEs can be made more expressive by replacing the simple Gaussian autoregressive conditionals with something more flexible, such as a mixtures of normals (or other densities).

2.3.3 Masked Autoregressive flows (MAF)

Single MADE density estimators have two key limitations. Firstly, they are sensitive to the order of the factorization in Eq. (3); some densities may have simple (eg., unimodal) conditionals in one factorization-order, but not in another, and this is typically not known a priori (see Papamakarios et al. 2017 for an illustration). Secondly, the assumption of simple (eg., Gaussian) conditionals may be overly restrictive.

Masked Autoregressive Flows (MAF; Papamakarios et al. 2017) address both of these limitations by constructing a stack of MADEs, where the output $\mathbf{u}$ of each MADE is taken as input for the next, with random re-ordering of the chain-rule factorization between each MADE. With multiple stacked MADEs and re-ordering, MAFs constitute very flexible neural autoregressive density estimators suitable for likelihood-free inference (Papamakarios et al. 2018). MAFs then define the following conditional density estimator:

$$\begin{aligned}p(\mathbf{t}|\boldsymbol{\theta};\mathbf{w}) &= \prod_i p(t_i|t_{1:i-1},\boldsymbol{\theta};\mathbf{w})\\ &= \mathcal{N}\left[\mathbf{u}(\mathbf{t},\boldsymbol{\theta};\mathbf{w})|\mathbf{0},\mathbf{I}\right] \times \prod_{n=1}^{N_{\rm mades}} \prod_{i=1}^{\dim(\mathbf{t})} \sigma_i^n(\mathbf{t},\boldsymbol{\theta};\mathbf{w}),\end{aligned} \tag{6}$$

[3] We will henceforth take MDN to mean Gaussian MDN (although other mixture models may be useful in certain situations).

[4] To avoid redundancy from the positive-definiteness of the covariance matrices, it is practical if the neural network parameterizes only the (upper triangular) Cholesky factors of the component covariances.

[5] Softmax: $\mathbf{x} \rightarrow \exp(\mathbf{x})/\Sigma\exp(x_i)$.

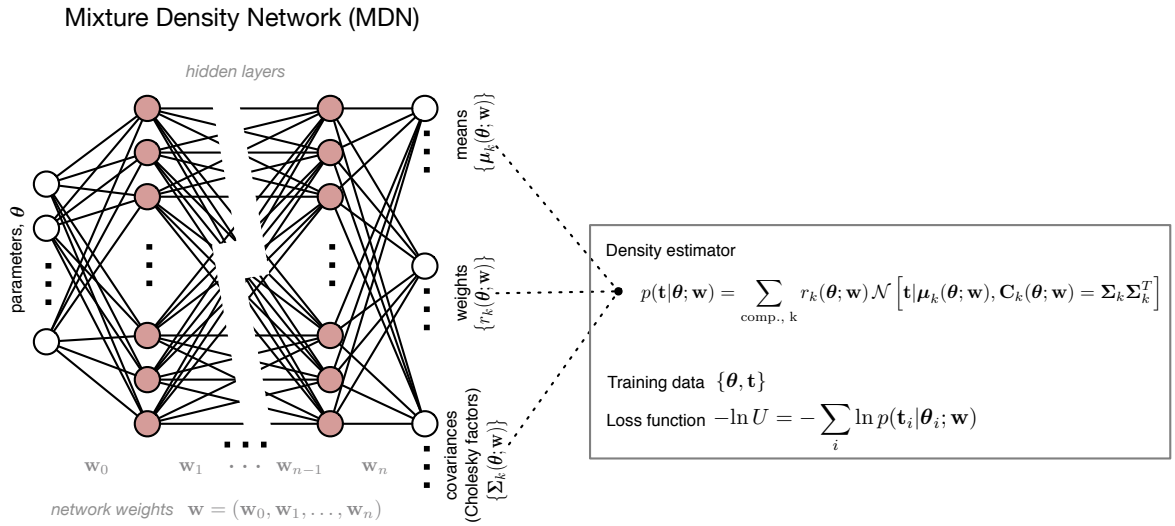

Figure 2. Schematic of the Mixture Density Network parameterization of the conditional density $p(\mathbf{t}|\boldsymbol{\theta})$. The means, weights and covariances of a Gaussian mixture model for $p(\mathbf{t}|\boldsymbol{\theta})$ are free functions of the parameters $\boldsymbol{\theta}$, parameterized by the weights, $\mathbf{w}$, of the neural network. The neural network takes $\boldsymbol{\theta}$ as input and outputs the parameters of the mixture model for those parameters.

where $\mathbf{u}$ is the output from the final MADE.

2.3.4 *Training neural density estimators*

To fit a neural density estimator to a set of simulated samples $\{\boldsymbol{\theta}, \mathbf{t}\}$, we want to find the weights of the neural network that minimize the Kullback-Leibler divergence between the parametric density estimator $p(\mathbf{t}|\boldsymbol{\theta}; \mathbf{w})$ and the target $p^*(\mathbf{t}|\boldsymbol{\theta})$:

$$D_{\mathrm{KL}}(p^* \,|\, p) = \int p^*(\mathbf{t}|\boldsymbol{\theta}) \ln \left(\frac{p(\mathbf{t}|\boldsymbol{\theta}; \mathbf{w})}{p^*(\mathbf{t}|\boldsymbol{\theta})} \right) d\mathbf{t} \tag{7}$$

Since we do not have access to the target density, only samples from it $\{\mathbf{t}, \boldsymbol{\theta}\}$, we take the (negative log) loss function to be a sum over the training samples:

$$-\ln U(\mathbf{w}|\{\boldsymbol{\theta}, \mathbf{t}\}) = -\sum_{i=1}^{N_{\mathrm{samples}}} \ln p(\mathbf{t}_i|\boldsymbol{\theta}_i; \mathbf{w}), \tag{8}$$

ie., a Monte Carlo estimate of the KL-divergence (up to an additive $\mathbf{w}$-independent constant), which is equivalent to the negative log-likelihood of the simulated data $\{\mathbf{t}, \boldsymbol{\theta}\}$ under the conditional density estimator $p(\mathbf{t}|\boldsymbol{\theta}; \mathbf{w})$. Note that the sum in Eq. (8) approximates Eq. (7), because the simulated data $\mathbf{t}$ (for their respective $\boldsymbol{\theta}$) in the training set $\{\mathbf{t}, \boldsymbol{\theta}\}$ are drawn from the target $p^*(\mathbf{t}|\boldsymbol{\theta})$ implicitly defined by the simulator.

For (Gaussian) MDN conditional density estimators, the loss is hence given by:

$$-\ln U(\mathbf{w}|\{\boldsymbol{\theta}, \mathbf{t}\}) = -\sum_i \ln \sum_{k=1}^{n_c} r_k(\boldsymbol{\theta}_i; \mathbf{w}) \, \mathcal{N}\left[\mathbf{t}_i \,|\, \boldsymbol{\mu}_k(\boldsymbol{\theta}_i; \mathbf{w}), \boldsymbol{\Sigma}_k(\boldsymbol{\theta}_i; \mathbf{w})\right]. \tag{9}$$

For MAF conditional density estimators, the loss is given by:

$$-\ln U(\mathbf{w}|\{\boldsymbol{\theta}, \mathbf{t}\}) = -\sum_i \ln \mathcal{N}\big(\mathbf{u}(\mathbf{t}_i, \boldsymbol{\theta}_i; \mathbf{w}) \,|\, \mathbf{0}, \mathbf{I}\big) + \sum_{n=1}^{N_{\mathrm{mades}}} \sum_{m=1}^{\dim(\mathbf{t})} \ln \sigma_m^n(\mathbf{t}_i, \boldsymbol{\theta}_i; \mathbf{w}). \tag{10}$$

The neural density estimators are then trained in the usual way by minimizing the negative log-loss with respect to the network weights, or inferring a posterior density over the weights given the training data (and some network weight prior).

Over-fitting can be mitigated by any of the standard regularization methods used for neural networks, such as early-stopping or dropout. Early-stopping splits the training data into a training and validation set, and terminates training when the loss ceases to improve for the validation set. Dropout masks some subset of the hidden units, chosen at random, at each training iteration (Srivastava et al. 2014). Further regularization can be achieved using ensembles of networks, or Bayesian networks, as described in the next section.

2.4 Bayesian networks, deep ensembles and stacked density estimators

Whilst training sufficiently complex NDEs on sufficiently large training sets provides a robust approach to likelihood-free inference in practice (Papamakarios & Murray 2016; Papamakarios et al. 2018), some simple sophistications can improve robustness to the choice of network architecture and small training sets (ie., when simulation is expensive).

Training individual NDEs (by minimizing the log-loss) on small training sets runs the risk of finding single local minima that may not best represent the training data. Furthermore, it may be difficult to choose an appropriate NDE network architecture for the problem at hand a priori; there is a trade-off to be made

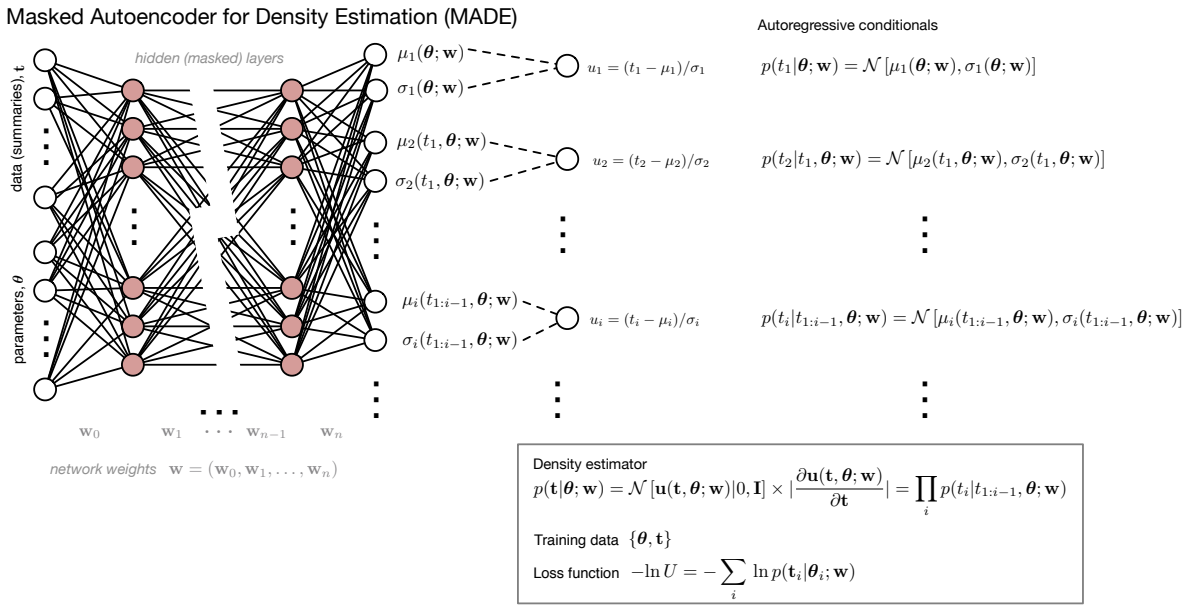

Figure 3. Schematic of the conditional Gaussian Masked Autoencoder for Density Estimation (MADE) parameterization of the conditional density $p(\mathbf{t}|\boldsymbol{\theta})$. The means and variances of the autoregressive conditionals are parameterized by the neural network, with the hidden layers carefully masked to ensure the autoregressive properties are satisfied. A masked autoregressive flow (MAF) is a stack of MADEs, where the output of each MADE is fed as input to the next, and the order of the autoregressive factorization is changed between MADEs.

between ensuring sufficient model complexity to fit the unknown data distribution, whilst avoiding over-fitting.

One simple resolution is to train an ensemble of NDEs (Lakshminarayanan et al. 2017; Lueckmann et al. 2018) $\{p_\alpha(\mathbf{t}|\boldsymbol{\theta};\mathbf{w})\}$, with a range of network architectures (and initializations). This ensemble can then be used to form a stacked density estimator for the learned likelihood-surface by stacking the individual trained NDEs,

$$p(\mathbf{t}|\boldsymbol{\theta};\mathbf{w}) = \sum_{\alpha=1}^{N_{\rm NDEs}} \beta_\alpha \, p_\alpha(\mathbf{t}|\boldsymbol{\theta};\mathbf{w}), \tag{11}$$

where the weights β_α are given by the relative likelihoods for each NDE, or their cross-validation scores (Smyth & Wolpert 1998, 1999). Stacked density estimators constructed this way are found to outperform a single best density estimator chosen from an ensemble of fits (Smyth & Wolpert 1998, 1999), and stacking ensembles of trained neural networks is common practice in machine learning for improving predictive accuracy.

As well as increasing robustness in the small training-set regime and against architecture choice, training ensembles of NDEs also allows for straightforward estimation of the uncertainty in the learned likelihood surface (ie., the weighted variance of the NDEs in the ensemble). This can be exploited in active learning schemes that use the uncertainty in the current likelihood-surface approximation to decide where to run new simulations, as described below.

A second approach to making neural networks robust in the small training-set regime is to train the networks in a Bayesian context, inferring a posterior distribution for the network weights given the training data, $p(\mathbf{w}|\{\mathbf{t},\boldsymbol{\theta}\})$ (see eg., Burden & Winkler 2008). This helps to regularize the networks in two ways; firstly, it allows us to put a prior over the network weights, eg., imposing some sparse regularization. Secondly, the inferred network-weight posterior can be used to define an expectation value of the network output, and also to assess uncertainty in the network output that can be exploited in active learning schemes, in a similar spirit to ensembles of networks. Bayesian networks are reported to be robust to over-fitting and remove the need for additional over-fitting mitigation strategies.

Bayesian networks have the appeal over network ensembles that they allow the user to control the regularization in an interpretable way through the prior, and provide principled uncertainties and expectation values for the network outputs. Ensembles on the other hand have the advantage that they are trivial to implement; optimizing an ensemble of networks is typically simpler and cheaper in practice than inferring the posterior over the weights for a single network (although fast approximate inference schemes such as variational inference help). Ensembles also make easy work of model averaging over different network architectures, which is more difficult (although still possible) in the Bayesian framework. Recent developments in Bayesian inference and subsequent marginalization over network architectures may prove useful in this context (Higson et al. 2018; Dikov et al. 2019).

2.5 Pre-training neural density estimators for likelihood-free inference

The weights of the neural density estimators are typically given random initializations by default. However, for some likelihood-free inference applications we may have an approximation for the sampling distribution $\tilde{p}(\mathbf{t}|\boldsymbol{\theta})$ available. In these cases, one can *pre-train* the NDEs by regressing them to the approximate $\tilde{p}(\mathbf{t}|\boldsymbol{\theta})$ before showing them any simulations, from which they can subsequently morph quickly towards the target when fed even a small number of simulations.

In practice this can be achieved as follows: generate a pre-training set $\{\boldsymbol{\theta},\mathbf{t},\tilde{p}\}$ by drawing parameters $\boldsymbol{\theta}$ from the prior $p(\boldsymbol{\theta})$, then drawing $\mathbf{t}$ from $\tilde{p}(\mathbf{t}|\boldsymbol{\theta})$ and evaluating $\tilde{p}(\boldsymbol{\theta},\mathbf{t})$. The NDEs can then be regressed to the approximate $\tilde{p}(\mathbf{t}|\boldsymbol{\theta})$ by minimizing the loss

function:

$$-\ln U(\mathbf{w}|\{\boldsymbol{\theta},\mathbf{t}\}) = -\sum_{i=1}^{N_{\rm samples}} \ln p(\mathbf{t}_i|\boldsymbol{\theta}_i;\mathbf{w}) - \ln \tilde{p}(\mathbf{t}_i|\boldsymbol{\theta}_i), \quad (12)$$

where the sum is over the pre-training set $\{\boldsymbol{\theta},\mathbf{t},\tilde{p}\}$. Similarly to Eq. (8), this loss represents a Monte Carlo estimate of the KL-divergence between the pre-training target $\tilde{p}(\mathbf{t}|\boldsymbol{\theta})$ and the NDE conditional density $p(\mathbf{t}|\boldsymbol{\theta};\mathbf{w})$. After this pre-training step, the pre-training data is discarded and the NDEs trained on simulations in the usual way.

In particular, when using approximate-score or IMNN compression, the compressed summaries can be cast into pseudo maximum-likelihood estimators through a simple shift and re-scaling (Alsing & Wandelt 2018b):

$$\mathbf{t} \to \boldsymbol{\theta}_* + \mathbf{F}_*^{-1}\mathbf{t}. \quad (13)$$

This hints at a natural first guess for their sampling distribution: Gaussian estimators for the parameters, with covariance $\mathbf{F}^{-1}$. In these cases, before running any simulations one can regress the NDEs to $\tilde{p}(\mathbf{t}|\boldsymbol{\theta}) \equiv \mathcal{N}(\mathbf{t}|\boldsymbol{\theta},\mathbf{F}^{-1})$. This initializes the NDEs to a rough (linear Gaussian) approximation of the target density. We call this initialization scheme *Fisher pre-training*.

In a similar spirit, for applications of DELFI where cheap but approximate simulations are available, the NDEs can be pre-trained on the approximate simulations first to give good starting points, before discarding the approximate sims and training on the full simulations in the usual way.

2.6 Adaptive acquisition of simulations with active learning

When performing likelihood-free inference in situations where forward simulation is expensive, the goal is to achieve the highest fidelity posterior inference with the fewest simulations possible. We therefore want to preferentially run simulations in the most interesting regions of the parameter space, which are not known a priori. Active learning allows the neural density estimators to call the simulator independently during training, automatically deciding on-the-fly where the best parameters to run new simulations are based on their current state of knowledge/ignorance of the target posterior. Here we present two key active learning approaches for adaptive acquisition of simulations for DELFI: sequential neural likelihood (based on Papamakarios et al. 2018), and Bayesian optimization style acquisition rules (based on Lueckmann et al. 2018).

2.6.1 *Active learning with Sequential Neural Likelihood (SNL)*

The Sequential Neural Likelihood (SNL; Papamakarios et al. 2018) approach runs simulations in a series of batches, where the parameters for each batch of new simulations are drawn from a proposal density based on the current posterior approximation, and the NDEs are re-trained after each new simulation batch. This way, the algorithm adaptively learns the most relevant parts of the parameter space to run new simulations and hence improve the ongoing posterior inference.

How to define an optimal proposal density based on the current posterior approximation is an open question. Papamakarios et al. (2018) used the current posterior approximation directly as their proposal for new simulations, which is a natural choice. An alternative is to use the geometric mean of the prior and the current posterior approximation as the proposal density, inspired by optimal proposal schemes for sequential Approximate Bayesian Computation (Alsing et al. 2018a); this has the nice property that it better samples the tails of the distribution, and may hence give more robust convergence there (we find this to be the case in experiments).

2.6.2 *Active learning with Bayesian optimization*

A second approach is to use Bayesian-optimization style acquisition rules for running new simulations (Lueckmann et al. 2018). In this scheme, the next simulation is run at the parameters that maximize some deterministic acquisition function $A(\boldsymbol{\theta})$, that encodes a trade-off between relevance (ie., being in a region of high posterior-density), and uncertainty in current learned posterior-density-surface. This active learning approach requires two ingredients: (1) some way of quantifying uncertainty in the current learned posterior surface, and (2) some carefully chosen acquisition rule.

Training an ensemble of NDEs (as described in §2.4) provides a straightforward estimate of the variance of the learned likelihood surface (the weighted variance of the NDEs in the ensemble), as do Bayesian networks (the variance of the network output under the network weight posterior). Defining an optimal acquisition rule poses a more challenging problem. A simple, pragmatic acquisition rule is just the current variance of the estimated posterior density (Lueckmann et al. 2018). This has the appeal that it is cheap and simple to compute, but has the disadvantage that it does not attempt to quantify the expected improvement in the density estimator after running a new simulation in any principled way. Järvenpää et al. (2018) try to address this by taking a more formal decision theoretic approach: minimizing the expected integrated variance of the approximate posterior, under a new simulation draw. Whilst this is clearly a better-motivated acquisition rule, it can be computationally cumbersome in practice since it involves optimizing over high-dimensional (parameter-space) integrals.

One might expect well-chosen deterministic acquisition rules (Bayesian optimization) to be more efficient than stochastically drawing new simulation parameters from an adaptive proposal (SNL). However, Durkan et al. (2018) reported that the two approaches gave broadly similar performance in a number of case studies in the context of DELFI. Furthermore, optimal simultaneous selection of multiple candidate points for Bayesian optimization style active learning (which is essential to leverage parallel simulation runs) provides additional challenges. These are active areas of research.

2.7 Global versus local emulators

For parameter inference tasks where the data have been observed in advance of the analysis, the target is the likelihood function $p(\mathbf{d}_{\rm obs}|\boldsymbol{\theta})$. In this situation, active learning helps us to selectively run simulations in the most relevant parts of parameter space to learn the target accurately. However, in some scenarios we may run many "experiments" that generate independent realizations of data $\mathbf{d}$ from the same data generating process, and we want to analyze those data as they are taken. In these situations, it is desirable to abandon active learning and build a global emulator for $p(\mathbf{d}|\boldsymbol{\theta})$ over the full prior volume, that can then be used to analyze any subsequent data $\mathbf{d}$ as they are observed. A typical example of this situation is event reconstruction for dark matter direct detection (eg., Simola et al. 2018): every time an event occurs it generates some data $\mathbf{d}$ (the response of the detectors to the event), and we want to infer the characteristics of the event (position, energy, etc) from those data. Having a pre-trained global emulator for $p(\mathbf{d}|\boldsymbol{\theta})$ would allow posterior inference from any event data $\mathbf{d}$ to be obtained rapidly, as

the events are observed. DELFI provides a natural framework for building global emulators for data sampling distributions for these scenarios.

3 DATA COMPRESSION

Whether data compression is required for performing DELFI or not depends critically on the the size of the data vector relative to the number of simulations that can be feasibly performed, given that one needs enough simulations to learn the sampling distribution of the data (summaries) as a function of the parameters, over the relevant parameter-space volume. For problems with modest dimensional data-vectors, or larger data vectors but where simulation is cheap, DELFI may be performed directly on the data without further compression. However, for large datasets with expensive simulations it is clearly advantageous to compress the data down to a small number of informative summary statistics, so that the density-estimation task need only be performed on the low-dimensional data-summaries.

In this section we review two key approaches to compressing N data down to p summaries – one per parameter – whilst aiming to retain as much information about the parameters as possible: approximate score-compression, and data compression with deep neural networks.

3.1 Approximate score-compression

When the likelihood function is known, the score function $\mathbf{t} = \nabla_{\boldsymbol{\theta}} \ln p(\mathbf{d}|\boldsymbol{\theta})$ yields compression of N data down to p summaries, one per parameter, such that the Fisher information of the data is preserved (provided the gradient is taken close to the true parameters, Alsing & Wandelt 2018b; Alsing et al. 2018b). For Gaussian data where the model depends on the parameters either through the mean or the covariance, score-compression is equivalent to MOPED (Heavens et al. 2000) or the optimal quadratic estimator (Tegmark et al. 1997), respectively.

For likelihood-free applications, the likelihood-function is obviously not known a priori, but the idea of score-compression can still provide a guiding hand for defining compressed data summaries. For many problems, whilst an exact likelihood is not known, an approximate (eg. Gaussian) likelihood may still be used for defining approximate score-compressed summaries, the only cost of the approximation being some loss of information. If no obvious likelihood approximation presents itself and the data space is not too large, one can learn the conditional density $p(\mathbf{d}|\boldsymbol{\theta})$ from simulations in the neighborhood of some fiducial parameters $\boldsymbol{\theta}_*$ (with an NDE), and use that to define an approximate score function. As a third approach, the score-function may be regressed directly from simulations in a likelihood-free manner using neural networks (Brehmer et al. 2018a,b,c).

3.1.1 Nuisance hardened approximate score-compression

For problems with p interesting parameters $\boldsymbol{\theta}$ and m additional nuisance parameters $\boldsymbol{\eta}$, Alsing & Wandelt (2018a) (building on Zablocki & Dodelson 2016) showed that it is possible to find n "nuisance hardened" score-compressed summaries (one per interesting parameter) that are insensitive-by-design to the nuisance parameters. This can be achieved through a simple projection involving the Fisher matrix (Zablocki & Dodelson 2016; Alsing & Wandelt 2018a),

$$\bar{\mathbf{t}}_{\boldsymbol{\theta}} = \mathbf{t}_{\boldsymbol{\theta}} - \mathbf{F}_{\boldsymbol{\theta}\boldsymbol{\eta}}\mathbf{F}^{-1}_{\boldsymbol{\eta}\boldsymbol{\eta}}\mathbf{t}_{\boldsymbol{\eta}}, \tag{14}$$

where $\mathbf{t}_{\boldsymbol{\theta}} = \nabla_{\boldsymbol{\theta}} \ln p(\mathbf{d}|\boldsymbol{\theta})$, $\mathbf{t}_{\boldsymbol{\eta}} = \nabla_{\boldsymbol{\eta}} \ln p(\mathbf{d}|\boldsymbol{\theta})$, the Fisher information matrix is given by $\mathbf{F} = -\langle \nabla_{(\boldsymbol{\theta},\boldsymbol{\eta})} \nabla^{T}_{(\boldsymbol{\theta},\boldsymbol{\eta})} \ln p(\mathbf{d}|\boldsymbol{\theta}) \rangle$, and $\bar{\mathbf{t}}_{\boldsymbol{\theta}} \in \mathbb{R}^p$ are the nuisance hardened summaries.

In the context of likelihood-free inference, because the score-compression and hence the nuisance parameter projection is approximate, the nuisance parameters should still be varied in the forward simulations to correctly capture any residual nuisance-sensitivity of the hardened summaries and give self-consistent nuisance marginalized posteriors (see Alsing & Wandelt 2018a for details).

The ability to project nuisance parameters in this way has profound implications for likelihood-free cosmology: the complexity of the inference task (and hence number of simulations required) now only depends on the number of interesting parameters[6], which for cosmological applications is typically relatively small ($\lesssim 10$).

3.2 Deep neural network data compression and information maximizing networks

An emerging trend in cosmology is to find cosmological parameter estimators from complex data sets by training deep neural networks to regress parameters from data simulations (Ravanbakhsh et al. 2016; Gupta et al. 2018; Ribli et al. 2018; Fluri et al. 2018a; Gillet et al. 2018). The resulting trained networks can be viewed as radical data compression schemes, summarizing large data sets down to a set of parameter estimators whose sampling distributions (and hence likelihood functions) are unknown. These neural network parameter estimators can be straightforwardly used as data summaries in a subsequent likelihood-free analysis. However, they typically require a large number of simulations spanning the full (relevant) parameter volume in order to train.

Combining the ideas of deep network and score compression, Information Maximizing Neural Networks (IMNN; Charnock et al. 2018) parameterize the data compression function $\mathbf{t}(\mathbf{d}) : \mathbb{R}^N \rightarrow \mathbb{R}^p$ as a neural network, training the network on a set of forward simulations such that the retained Fisher information content of the compressed summaries is maximized (see Charnock et al. 2018 for details). This tends towards optimal non-linear compression when provided with a sufficiently flexible architecture and representative simulations[7]. IMNNs have some advantages over other deep network parameter estimators. Firstly, they take fewer simulations to train, only requiring simulations around some fiducial parameters rather than spanning the full parameter volume. Secondly, by construction they implicitly Gaussianize the compressed summaries (and provide pseudo-maximum likelihood estimators from the transformed likelihood). This means that in a subsequent DELFI analysis, a relatively simple conditional density estimator may be used, requiring fewer simulations to converge. Thirdly, IMNNs also provide an estimated Fisher matrix that is useful for initializing density estimators for DELFI (see §4.2.3), and also for projecting out any nuisance parameters in the compression using Eq. (14).

Other novel deep network compression schemes train networks

[6] Since DELFI involves learning $p(\mathbf{t}|\boldsymbol{\theta})$, and when using nuisance hardened summary statistics, $\mathbf{t} \in \mathbb{R}^n$ and $\boldsymbol{\theta} \in \mathbb{R}^n$ irrespective of the presence or number of additional nuisance parameters in the problem.

[7] Asymptotic optimality is only expected for unimodal likelihoods and taking an expansion point close to the maximum-likelihood; this can be iterated if required.

to find data summaries based on their ability to distinguish (via classification) between different models (Merten et al. 2018). This is an active area of research.

3.3 Considerations for cosmological data analysis: two-step data compression

It is standard practice in cosmology to compress large data sets down to some set of informative summary statistics, motivated by knowledge of the underlying physics of the problem. For example, surveys are often compressed down to power spectra or higher order n-point statistics, supernovae lightcurves and spectra are compressed down to point estimates for their apparent magnitudes and redshifts, etc. Whilst massive data compression using the score or deep networks is, in principle, possible at the level of the raw data, we anticipate that applications of likelihood-free inference in cosmology will typically first construct a number of "first level summaries" (eg., the usual n-point statistics etc), and then perform a second massive compression step on those summaries (Alsing et al. 2018b).

Whilst this initial compression to some first-level summary statistics may seem unnecessary (and potentially lossy), it comes with some advantages over massively compressing maps (or raw data) directly. Even sophisticated simulations may represent incomplete descriptions of the true generative data model, limited by computational resources, incomplete knowledge of the instrument or hard-to-simulate non-linear physics, etc. The first level compression step allows us to use only aspects of the data that we expect to be well-modelled by the approximate simulations. For example, problems involving an N-body step might resort to approximations such as COLA (Tassev et al. 2013), which are only accurate for certain statistics and on certain scales. For cosmic microwave background analyses, noise simulations are typically expensive and approximations may be employed; cross-correlations between detector frequencies may be well-modelled, whereas the auto-power spectra may be less reliable.

On the other hand, applying flexible deep network compression schemes to the raw data offers the opportunity to learn highly informative data summaries that are not captured by standard cosmological estimators. However, this comes with the risk of learning features/approximations in the simulations that do not well-describe the data, and should therefore only be used (cautiously) for very high-fidelity simulators.

4 PYDELFI: A PUBLIC CODE FOR DENSITY ESTIMATION LIKELIHOOD-FREE INFERENCE

In this section we introduce PYDELFI – a flexible public code for performing density-estimation likelihood-free inference with NDEs and active learning. We briefly outline some of the implementation details and key features of the code here, referring the reader to `https://github.com/justinalsing/pydelfi` for tutorials and documentation.

4.1 Overview

Performing density-estimation likelihood-free inference with PYDELFI proceeds as follows:

(1) Specify the architectures – number of layers, hidden units, and activation functions – for an ensemble of neural density estimators (MDNs, MADEs, MAFs, or a combination of the three).

(2) Specify a `simulator()` function that takes in parameters and returns a simulated data vector.

(3) If data compression is required, specify a `compressor()` function that takes in a data vector and returns a vector of compressed summaries. This could be an implementation of approximate-score compression, a trained deep network parameter estimator, information-maximizing network, or otherwise.

(4) Provide the observed data vector and run PYDELFI using either the sequential neural likelihood or Bayesian optimization active learning methods, to learn the likelihood function. These are implemented as described in Algorithms 1 and 2 respectively. Simulation batches are run in parallel with MPI as standard, and the user has control over the number of simulations to run per round, the number of rounds to run, and the network training scheme (see below). The result is a callable likelihood function that improves during each round of new simulations and network training.

Alternatively, if a suite of simulations has been run beforehand (spanning the relevant parameter volume), these can be fed straight into PYDELFI and the ensemble of NDEs is trained on those (without exploiting the active learning strategies). For more optimal use of resources, however, it is advantageous to provide PYDELFI with a callable simulator so that it can exploit active learning to decide where to run simulations on-the-fly.

In the following sections we give brief details of the neural network implementation, initialization and training schemes (§4.2), active learning strategies (§4.3) and data compression options (§4.4).

Algorithm 1 Schematic outline of DELFI with the sequential neural likelihood method. In the description below, $\tilde{p}$ represents the current posterior approximation, π denotes the prior, and $\mathbf{t}_o$ denotes the observed data summaries.

```
// Create ensemble of NDEs:
NDEs = NDEs(chosen network architectures)

// (fisher pre-training happens here if desired)

// Choose initial proposal density q^(0)(θ):
q^(0)(θ) = chosen initial proposal

// SNL: run sims in batches with adaptive proposal
for n in 0 : n_rounds do
    for i in 0 : n_batch do
        θ_i ← q^(n)(θ)
        d_i ← simulator(d|θ_i)
        t_i = t(d_i)
        {t, θ}_training ← t_i, θ_i
    // train NDEs, update proposal after each round
    train(NDEs, {t, θ}_training)
    q^(n+1)(θ) = sqrt(p̃(θ|t_o)π(θ))
```

4.2 Neural network implementation and training

4.2.1 Training and mitigating over-fitting

All neural networks are implemented in TENSORFLOW (Abadi et al. 2015). As a default, we train the neural networks using the stochastic gradient optimizer ADAM (Kingma & Ba 2014), with a default batch-size of one tenth of the training set at each training cycle

Algorithm 2 Schematic outline of DELFI with Bayesian optimization. In the description below, $\tilde{p}$ represents the current posterior approximation, and A denotes the acquisition function, and $\mathbf{t}_o$ denotes the observed data summaries.

```
// Create ensemble of NDEs:
NDEs = NDEs(chosen network architectures)

// Choose acquisition rule A(θ|NDEs):
A(θ|NDEs) = chosen acquisition rule

// (fisher pre-training happens here if desired)

// run initial batch of sims with proposal q^(0)(θ)
for i in 1 : n_initial do
    θ_i ~ q^(0)(θ)
    d_i ~ simulator(d|θ_i)
    t_i = t(d_i)
    {t, θ}_training ← t_i, θ_i

// train the NDEs
train(NDEs, {t, θ}_training)

//  Bayesian optimization acquisition rounds: run sims in batches at the optimal acquisition point
for n in 1 : n_rounds do
    θ_n = argmax A(θ|NDEs)
    for i in 1 : n_batch do
        d_i ← simulator(d|θ_n)
        t_i = t(d_i)
        {t, θ}_training ← t_i, θ_n
    // train NDEs
    train(NDEs, {t, θ}_training)
```

and a learning rate of 0.001. Over-fitting is mitigated using early-stopping; during each training cycle, some fraction of the training set is set aside for validation (default 10%), and training is terminated when the validation-loss does not improve after some user-specified threshold number of epochs (default 20). Learning rates, cross-validation fractions and early-stopping thresholds can be easily controlled by the user. The use of ensembles of networks provides additional protection against over-fitting.

4.2.2 *Ensembles and stacking*

The learned likelihood function is constructed by stacking the NDEs in the ensemble, trained with early-stopping to avoid over-fitting, weighted by their relative cross-validation losses.

4.2.3 *Initialization: pre-training*

As described in §4.2.3, in cases where an approximate sampling distribution $\tilde{p}(\mathbf{t}|\boldsymbol{\theta})$ is available, this can be exploited to pre-train the NDEs to sensible starting points from which they can quickly morph to the target when subsequently fed a small number of simulations. This *pre-training* step can be straightforwardly performed in PYDELFI by feeding in a pre-training dataset of parameter-data summary-log density triplets $\{\boldsymbol{\theta}, \mathbf{t}, \ln\tilde{p}\}$ (cf §4.2.3), generated (by the user) from the prior and approximate sampling distribution. If the data summaries are pseudo maximum-likelihood estimators (from either score or IMNN compression) and a Fisher matrix is provided, The pre-training data are discarded after the initialization of the NDEs, before training on simulations in the usual way.

If no good approximate starting point is available, the network weights are initialized randomly by default.

4.3 Active learning

4.3.1 *Sequential neural likelihood*

We implement SNL as outlined in Algorithm 1. The user can specify an initial proposal for running the first batch of simulations. After each round of training, we draw new parameters for simulating from an updated proposal, $q(\boldsymbol{\theta}) = \sqrt{\tilde{p}(\boldsymbol{\theta}|\mathbf{t}_o)\pi(\boldsymbol{\theta})}$ – the geometric mean of the prior π and the current posterior approximation $\tilde{p}$ (inspired by Alsing et al. 2018a).

During each simulation acquisition batch, simulations are run in parallel with MPI. The number of simulations per batch is chosen by the user.

4.3.2 *Bayesian optimization*

We implement active learning with Bayesian optimization as described in Algorithm 2. The implemented acquisition function is the estimated posterior variance, calculated from the ensemble of NDEs; bespoke acquisition functions can be implemented by the user if needed.

Whilst simulations can be acquired one-by-one in this set-up, where parallel computing is available it is typically desirable to run many simulations concurrently. Therefore, PYDELFI runs batches of simulations in parallel (with MPI) at each derived acquisition point as default.

4.4 Data compression

PYDELFI comes with classes for approximate-score compression for common exponential family data-distributions. Where expectation values, covariances, derivatives, etc., need to be estimated from forward simulations, these are run in parallel with MPI as standard, otherwise pre-computed or analytical approximations can be fed in. For IMNN compression, a public implementation is available at `https://github.com/tomcharnock/IMNN`.

PYDELFI has the flexibility to take any (or no) compression scheme; bespoke compression schemes can be straightforwardly defined by the user and fed into PYDELFI.

5 CASE STUDY I (VALIDATION): JLA SUPERNOVAE ANALYSIS

Supernova data analysis is an interesting opportunity for likelihood-free methods, since the data are impacted by a number of systematic biases and selection effects that need to be carefully accounted for to obtain robust cosmological parameter constraints. Whilst progress has been made recently in developing Bayesian hierarchical models (BHMs) that attempt to carefully treat these effects (Mandel et al. 2009; March et al. 2011; Rubin et al. 2015; Shariff et al. 2016; Roberts et al. 2017; Hinton et al. 2018), likelihood-free methods have the advantage that the forward model complexity is unfettered by practical limitations of implementing and sampling high-dimensional BHMs.

As a validation test, we perform a simple analysis of the JLA

data (Betoule et al. 2014) under assumptions that allow us to compare against an exact known likelihood. The set-up is identical to Alsing et al. (2018b), which we review briefly below.

5.1 JLA data and model

The JLA sample is comprised of 740 type Ia supernovae with estimated apparent magnitudes $m_{\rm B}$, redshifts z, color at maximum-brightness C and stretch X_1 parameters. We take the data vector to be the vector of estimated apparent magnitudes $\mathbf{d} = (\hat{m}_{\rm B}^1, \hat{m}_{\rm B}^2, \ldots, \hat{m}_{\rm B}^M)$, where uncertainties in z, C and X_1 are implicitly accounted for in the covariance matrix (see Betoule et al. 2014, also Figure 4).

We take the expected apparent magnitudes of type Ia supernovae to be given by (Tripp 1998),

$$m_{\rm B} = 5\log_{10}\left[\frac{D_{\rm L}^*(z;\boldsymbol{\theta})}{10{\rm pc}}\right] - \alpha X_1 + \beta C + M_{\rm B} + \delta M\,\Theta(M_{\rm stellar} - 10^{10}M_\odot) \quad (15)$$

where $D_{\rm L}^*$ is the luminosity distance (at reference $h = 1$), $\boldsymbol{\theta}$ are the cosmological parameters (see below), α and β are calibration parameters for the stretch and color, and $M_{\rm B}$ and δM characterize the host stellar-mass dependent reference absolute magnitude. Θ is the Heaviside function.

We assume a flat wCDM cosmology parameterized by matter density $\Omega_{\rm m}$ and dark energy equation-of-state $p/\rho = w_0$.

5.2 Simulations

For this validation case, simulations are just draws from the (exact) Gaussian sampling distribution of the data, ie., drawing Gaussian data from Eq. (16).

5.3 Data compression

For this validation case we assume the data are Gaussian,

$$\ln p(\mathbf{d}|\phi) = -\frac{1}{2}(\mathbf{d} - \boldsymbol{\mu}(\phi))^T\mathbf{C}^{-1}(\mathbf{d} - \boldsymbol{\mu}(\phi)) - \frac{1}{2}\ln|\mathbf{C}|, \quad (16)$$

with mean given by Eq. (15), and we assume a fixed covariance matrix from Betoule et al. (2014) (see also Alsing et al. 2018b for details of the covariance matrix).

For data compression, we use the score of the Gaussian likelihood:

$$\mathbf{t} \equiv \nabla_{\boldsymbol{\theta}}\mathcal{L}_* = \nabla_{\boldsymbol{\theta}}^T\boldsymbol{\mu}_*\mathbf{C}^{-1}(\mathbf{d} - \boldsymbol{\mu}_*), \quad (17)$$

where '$*$' indicates evaluation at fiducial parameters $\boldsymbol{\theta}_* = (0.202, -0.748, -19.04, 0.126, 2.644, -0.0525)$.[8]

[8] Found in a few iterations of the pseudo maximum-likelihood estimator, Eq. (13).

5.4 Priors

We assume broad Gaussian priors on the parameters $\boldsymbol{\theta} = (\Omega_{\rm m}, w_0, \alpha, \beta, M_{\rm B}, \delta M)$ with mean and covariance:

$$\boldsymbol{\mu}_{\rm P} = (0.3,\ -0.75,\ -19.05,\ 0.125,\ 2.6,\ -0.05),$$

$$\mathbf{C}_{\rm P} = \begin{pmatrix} 0.4^2 & -0.24 & 0 & 0 & 0 & 0 \\ -0.24 & 0.75^2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.1^2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.025^2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.25^2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.05^2 \end{pmatrix}, \quad (18)$$

with additional hard prior boundaries on $\Omega_{\rm m} \in [0, 0.6]$ and $w_0 \in [-1.5, 0]$.

5.5 DELFI set-up

We ran DELFI using the SNL active learning scheme as described in §4. An ensemble of six NDEs was used: five MDNs with 1–5 Gaussian components respectively, each with two hidden layers of 50 hidden units, and a MAF containing five MADEs, each with two hidden layers of 50 units. We use tanh activation functions throughout.

Simulations were run in batches of 250 after an initial Fisher pre-training step to initialize the network ensemble.

5.6 Results

Fig. 5 (inset) shows the convergence of the DELFI NDE-ensemble as a function of the number of simulations. Convergence is achieved after $O(10^3)$ simulations. This is a substantial improvement on the 20,000 simulation requirement reported for the same problem in Alsing et al. (2018b). This also represents a substantial improvement over the Bayesian optimization likelihood-free inference (BOLFI) approach presented in Leclercq (2018). That work reported 6000 simulations were required for the same toy JLA analysis problem (but only inferring two parameters, with the other four implicitly marginalized over). The BOLFI approach implemented in that work also requires much stronger assumptions about the sampling distribution of the data, implicitly assuming that the data are Gaussian with a known covariance matrix but unknown mean (although less restrictive implementations of BOLFI are possible; Gutmann & Corander 2016).

Fig. 5 shows the recovered DELFI posterior after 1000 simulations (red), against a long-run MCMC chain for validation (black). The DELFI and MCMC posteriors are in excellent agreement.

For this case study, pre-training the networks took ∼minutes per NDE, after which subsequent SNL training rounds took tens of seconds per NDE (on a 3GHz Intel Core i7 processor). Since the simulations are inexpensive in this simple case, the cost of performing DELFI was dominated by training the networks and sampling intermediate proposal densities.

5.7 Discussion

This simple validation case gives insight into the relative performance of DELFI and MCMC sampling for simple problems where the sampling distribution of the data is known, and the likelihood can be evaluated exactly for given model parameters. In the JLA example above, DELFI was well converged after $O(10^3)$ forward simulations (which are of the same cost as likelihood evaluation

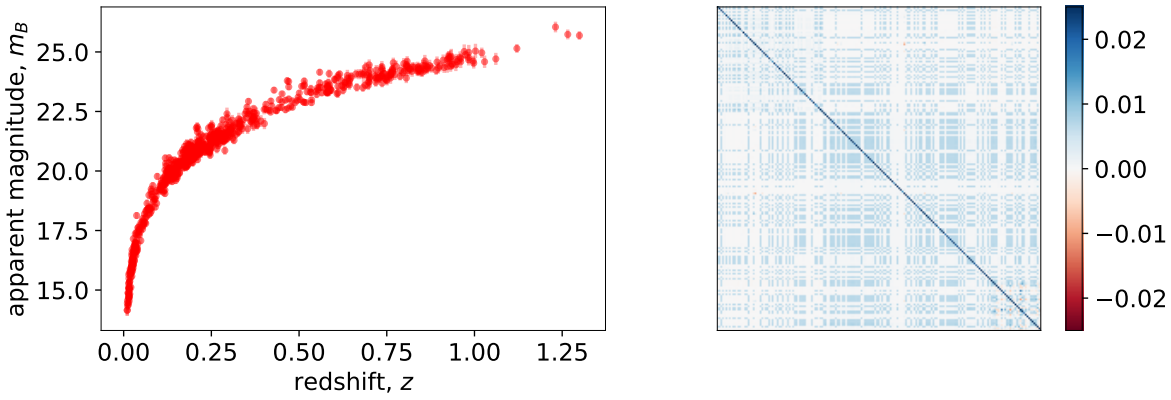

Figure 4. Left: Measured apparent magnitudes (with uncertainties) against their measured redshifts for the JLA supernova sample. Right: Covariance matrix corresponding to the observed apparent magnitudes.

for this case), while MCMC sampling typically requires one or two orders-of-magnitude more likelihood calls to give well-converged chains. Since training neural density estimators on small training sets is computationally inexpensive, in the many cases where the cost of DELFI is dominated by making draws from $p(\mathbf{d}|\boldsymbol{\theta})$ DELFI offers a fast and accurate alternative to MCMC sampling, giving orders of magnitude speed-up for typical ~ 6 parameter problems with expensive likeihoods (ie., when the cost of likelihood evaluations/simulations dominate over the cost of training the networks). The number of simulations required for DELFI to converge for these known-likelihood problems can be further reduced by training an NDE that corresponds to the known data sampling distribution, eg., a Gaussian with parameter-dependent mean and fixed covariance matrix for this particular validation case.

The performance gain of DELFI over MCMC for these known-likelihood inference problems is due in part to the data compression step, turning the inference task into a low-dimensional conditional density estimation task. For unimodal likelihoods, the score provides asymptotically optimal compressed summaries and is readily available, and application of DELFI as a replacement for MCMC is straightforward. However, for multimodal likelihoods, more care must be taken in defining approximately sufficient statistics for the problem at hand.

6 CASE STUDY II: TOMOGRAPHIC COSMIC SHEAR PSEUDO-C_ℓ ANALYSIS

Cosmic shear data are well suited to likelihood-free analyses, containing a large number of effects that may be simulated (to varying degrees) but are challenging to build into an accurate likelihood function. Non-linear physics and baryonic feedback (Rudd et al. 2008; Harnois-Déraps et al. 2015), intrinsic alignments (Joachimi et al. 2015), shape and photo-z measurement systematics (Massey et al. 2012; Mandelbaum 2018; Salvato et al. 2018), image blending (Mandelbaum 2018), reduced shear corrections (Krause & Hirata 2010), non-trivial non-Gaussian sampling distributions for common summary statistics (Sellentin et al. 2018), the redshift-dependent source galaxy population model (Kannawadi et al. 2018), etc., all have the potential to bias parameter inferences if not carefully accounted for. As well as promising more principled inference, LFI may also open up the possibility to extract extra information from non-standard lensing observables (eg., magnification Hildebrandt et al. 2009; van Waerbeke 2010; Hildebrandt et al. 2013; Duncan et al. 2013; Heavens et al. 2013; Alsing et al. 2015a) and non-linear scales via, eg., peak counts (Kratochvil et al. 2010; Fluri et al. 2018b), bispectrum (Cooray & Hu 2001) etc.

For this simple demonstration, we perform cosmological parameter inference from tomographic shear pseudo-C_ℓs for a Euclid-like survey. We focus on the large-scales where the pseudo-C_ℓ likelihood is intractable and standard Gaussian likelihood approximations are expected to break down.

6.1 Tomographic shear data and model

As light from distant galaxies propagates through the Universe on its way to us, it gets gravitationally lensed by the intervening large-scale structure, imprinting a coherent distortion on the galaxy images observed on the sky. This coherent lensing distortion field provides a unique probe of both the evolution of the 3D matter distribution, and geometry of the Universe via the distance-redshift relation. In particular, the observed shapes of galaxies are modified by the lensing "cosmic shear" fields, with their ellipticities ϵ picking up an additive distortion (in the weak lensing limit):

$$\epsilon = \epsilon_{\rm int} + \gamma, \tag{19}$$

where $\epsilon_{\rm int}$ is the unobserved intrinsic (unlensed) ellipticity, and γ is the additional shear due to gravitational lensing. The statistical properties of the shear field γ provide a sensitive probe of cosmology (see eg., Kilbinger 2015 for a review).

A weak lensing survey involves measuring the shapes, redshifts and angular positions on the sky of a large number of galaxies, which are then used to constrain cosmological parameters by eliciting the statistics of the cosmic shear signal in the data. We will take our data vector to be a set of (pixelized) shear maps estimated from a lensing survey $\mathbf{d} = (\boldsymbol{\gamma}^{(1)}, \boldsymbol{\gamma}^{(2)}, \ldots, \boldsymbol{\gamma}^{(n_z)})$, with galaxies grouped into n_z tomographic redshift bins (based on their estimated redshifts). The estimated shear in a given tomographic bin α and pixel p is taken to be:

$$\gamma_p^{(\alpha)} = \sum_{i \in (p,\alpha)} \hat{\epsilon}_i / N_p^{(\alpha)}, \tag{20}$$

ie., the average estimated ellipticity of the $N_p^{(\alpha)}$ galaxies in that pixel. The unknown intrinsic ellipticities are assumed to be zero mean

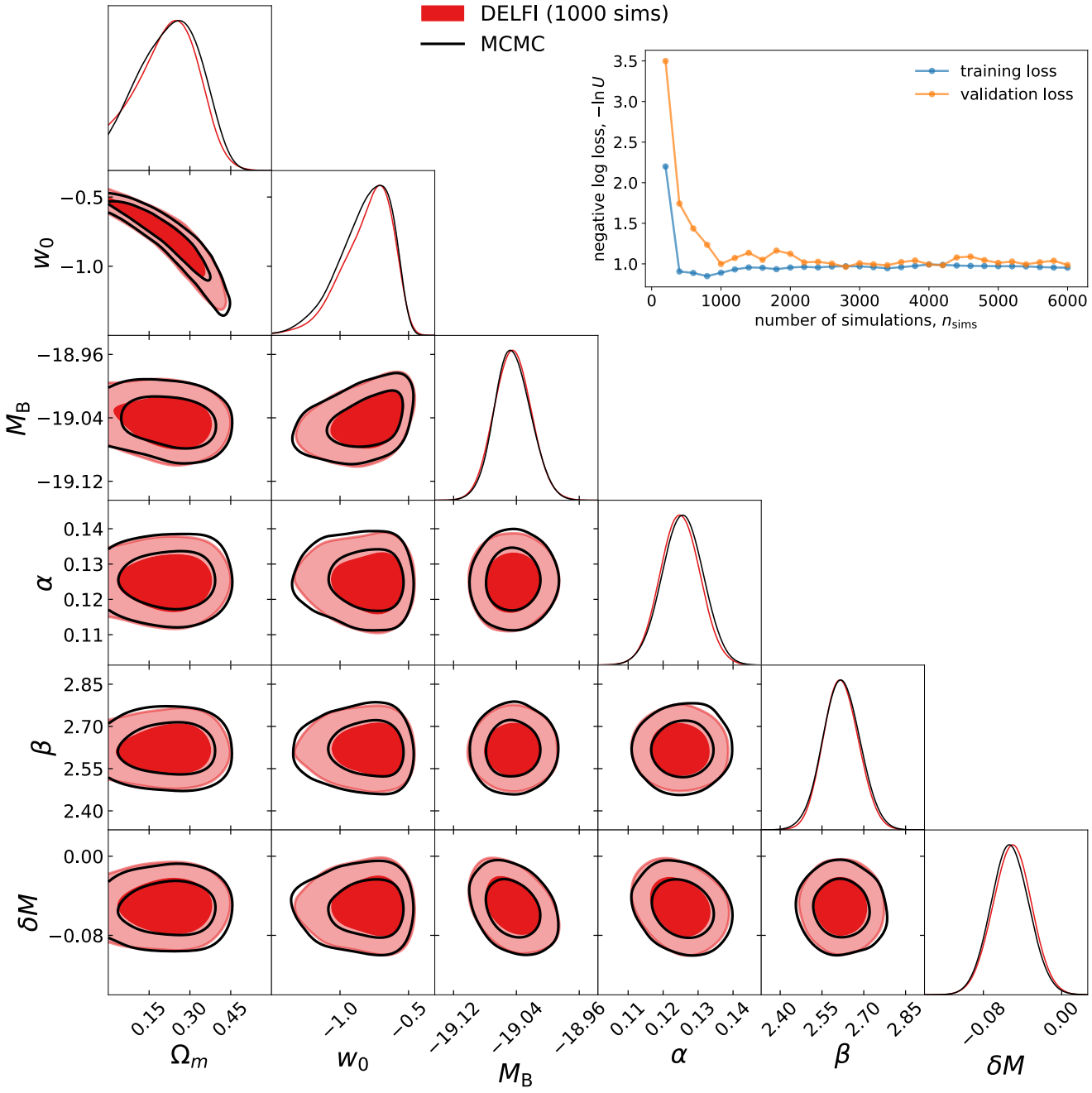

Figure 5. 68 and 95% credible regions of the 2D projections of the inferred posteriors for the JLA supernovae analysis, from a long-run MCMC chain (black) and DELFI from 1000 forward simulations (red). The posteriors from DELFI (after just 1000 simulations) and the long-run MCMC chain are in excellent agreement.

random variates with standard deviation σ_e, giving Gaussian "shape noise" $\sigma_p^{(\alpha)} = \sigma_e/\sqrt{N^{(\alpha)}}$ on each pixel (in the limit of many galaxies per pixel). The shape noise will invariably be anisotropic due to varying number of sources per pixel, and maps will be substantially masked due to incomplete sky coverage, masking around bright sources in the survey etc.

Mock data for this case study are generated for a survey set-up similar to the upcoming ESA *Euclid* survey (Laureijs et al. 2011) (as described in §6.2), and are shown in Figure 6.

6.1.1 *Tomographic shear power spectra*

In this case study we will focus on extracting information from the tomographic power spectra of the cosmic shear fields.

For a given (flat) cosmological model and parameters, the predicted angular power spectra between tomographic redshift bins α and β are given by[9] (Kaiser 1992, 1998; Hu 1999, 2002; Takada

[9] In the Limber approximation, (Limber 1954).

& Jain 2004; Kitching et al. 2017),

$$C^{\gamma\gamma}_{\ell,\alpha\beta} = \int \frac{d\chi}{\chi^2}\, w_\alpha(\chi) w_\beta(\chi)\,[1+z(\chi)]^2\, P_\delta\left(\frac{\ell}{\chi}; z(\chi)\right), \tag{21}$$

with comoving distance-redshift relation $\chi(z)$, matter power spectrum $P_\delta(k\ \chi)$, and lensing weight functions given by

$$w_\alpha(\chi) = \frac{3\Omega_{\rm m} H_0^2}{2}\chi \int_\chi^{\chi_{\rm H}} d\chi'\ n_\alpha(\chi')\frac{\chi'-\chi}{\chi'}, \tag{22}$$

where $n_\alpha(\chi)d\chi = p_\alpha(z)dz$ is the redshift distribution for galaxies in redshift bin α. Cosmological parameters enter in both the matter power spectrum and distance-redshift relation. We will assume a flat ΛCDM cosmology with parameters $\boldsymbol{\theta} = (\sigma_8, \Omega_{\rm m}, \Omega_{\rm b}, h, n_s)$.

6.2 Simulations

The simulations for this demonstration proceed as follows:

(i) Simulate a Gaussian random shear field on tomographic slices (in healpix pixelization (Gorski et al. 2005) with $n_{\rm side} = 128$, $\ell_{\rm max} = 3n_{\rm side}-1 = 383$), with auto- and cross- power spectra corresponding to the input cosmology $\boldsymbol{\theta}$ (cf., Eq. 21).
(ii) Add (anisotropic) shape noise to the healpix maps.
(iii) Apply Euclid-like mask (footprint and star-mask; Figure 6).
(iv) Compute tomographic pseudo-C_ℓ auto- and cross- band powers from the noisy tomographic maps.

We assume a survey set-up similar to the upcoming ESA *Euclid* survey (Laureijs et al. 2011): 15,000 square degrees with a mean galaxy number density of $\bar{n} = 30\,\mathrm{arcmin}^{-2}$, an overall galaxy redshift distribution $n(z) \propto z^2 \exp\left[-(1.41z/z_m)^{1.5}\right]$ with a median $z_m = 0.9$, Gaussian photo-z errors with standard deviation $\sigma_z = 0.05*(1+z)$, and five tomographic bins with equal mean galaxy number density per bin. Modes are binned into ten log-spaced bands between $\ell = 10$ and $\ell = 383$.

For the shape noise, we add zero-mean Gaussian noise to each pixel with variance $\sigma_e^2/N_p^{(i)}$, where $\sigma_e = 0.3$ and N_p is the number of galaxies in pixel p, tomographic bin i. Galaxies are Poisson distributed among pixels according to the mean number density per tomographic slice to give realistic, anisotropic shape noise.

The mock data for this demonstration are simulated following the procedure above, assuming a Planck 2018 cosmology (Aghanim et al. 2018): $\sigma_8 = 0.811$, $\Omega_{\rm m} = 0.315$, $\Omega_{\rm b} = 0.049$, $h = 0.674$ and $n_s = 0.965$, $w_0 = -1.03$.

6.3 Data compression

We compress the noisy, masked tomographic shear maps down in two steps. First, we compute from the maps a set of tomographic auto- and cross- angular pseudo-C_ℓ band powers, for K ℓ-bands and n_z tomographic bins:

$$\tilde{\mathbf{d}} = (\hat{C}_{\mathcal{B}_1,11}, \hat{C}_{\mathcal{B}_1,12}, \ldots, \hat{C}_{\mathcal{B}_1,n_z n_z}, \hat{C}_{\mathcal{B}_2,11}, \ldots, \hat{C}_{\mathcal{B}_K,n_z,n_z}) \tag{23}$$

where

$$\hat{C}_{\mathcal{B}_k,ij} = \sum_{\ell\in\mathcal{B}_k}\sum_{m=-\ell}^{\ell} \hat{a}^{(i)}_{\ell m}\hat{a}^{(j)*}_{\ell m}, \tag{24}$$

and $\{\hat{a}^{(i)}_{\ell m}\}$ are the E-mode spherical harmonic coefficients of the noisy, masked, tomographic shear maps. Note that in the likelihood-free framework, there is no need to deconvolve the mask or subtract the noise bias from the estimated band powers; these are all taken care of (exactly) in the forward simulations.

Secondly, we compress the tomographic band powers $\tilde{\mathbf{d}}$ assuming they are approximately Gaussian distributed (ie., MOPED compression Heavens et al. 2000), giving compressed summaries:

$$\mathbf{t} \equiv \nabla_{\boldsymbol{\theta}}\mathcal{L}_* = \nabla^T_{\boldsymbol{\theta}}\boldsymbol{\mu}_*\mathbf{C}^{-1}(\tilde{\mathbf{d}} - \boldsymbol{\mu}_*), \tag{25}$$

taking fiducial parameters $(\sigma_8, \Omega_{\rm m}, \Omega_{\rm b}, h, n_s) = (0.8, 0.3, 0.05, 0.7, 0.96)$ for performing the compression. We estimate the covariance and mean by running 10^3 forward simulations. The derivatives are estimated as a forward difference using 100 pairs of simulations per parameter, with matched random seeds to suppress sample variance, and step sizes of 5% for each parameter respectively. The extra simulation burden here could easily be eliminated by using analytical models for the mean (masked) band powers and covariance matrix in place of Monte Carlo estimates (we use Monte Carlo estimates here for convenience reasons only).

Note that the requirements on the accuracy of the mean, covariance and derivatives used for the data compression are much less onerous than for an approximate Gaussian likelihood-based analysis: any errors in these estimated quantities can *only* lead to sub-optimality in the compression, in contrast to a likelihood-based analysis where errors/incorrect-assumptions can bias parameter inferences.

Since the pseudo-C_ℓs are not expected to be exactly Gaussian distributed (particularly at low ℓ), the second compression step may lose a small amount of information.

6.4 Priors

We assume broad independent Gaussian priors over $\boldsymbol{\theta} = (\sigma_8, \Omega_{\rm m}, \Omega_{\rm b}, h, n_s)$ with means $\boldsymbol{\mu}_{\boldsymbol{\theta}} = (0.8, 0.3, 0.05, 0.7, 0.96)$, standard deviations $\boldsymbol{\sigma}_{\boldsymbol{\theta}} = (0.3, 0.3, 0.1, 0.3, 0.3)$, and hard parameter limits $\sigma_8 \in [0.4, 1.2]$, $\Omega_{\rm m} \in [0, 1]$, $\Omega_{\rm b} \in [0, 0.3]$, $h \in [0.4, 1]$, and $n_{\rm s} \in [0.7, 1.3]$.

6.5 DELFI set-up

We ran DELFI using the SNL active leaning scheme described in §4. An ensemble of six NDEs was used: five MDNs with 1–5 Gaussian components respectively, each with two hidden layers of 50 hidden units, and a MAF containing five MADEs, each with two hidden layers of 50 units. We use tanh activation functions throughout.

Simulations were run in batches of 200 after an initial Fisher pre-training step to initialize the network ensemble. We note that for this case study, the computational cost of training the NDEs is small compared to the cost of running simulations (which in this case is dominated by the spherical-harmonic transforms).

6.6 Results

Figure 7 (inset) shows the convergence of the DELFI NDE-ensemble as a function of the number of forward simulations; convergence is achieved after $O(10^3)$ forward simulations. Figure 7 shows that the input cosmological parameters are well recovered (within uncertainties).

For this case study, pre-training the networks took ∼minutes per NDE, after which subsequent SNL training rounds took tens of seconds per NDE (on a 3GHz Intel Core i7). Simulations took a few seconds each, so already for these only modestly expensive

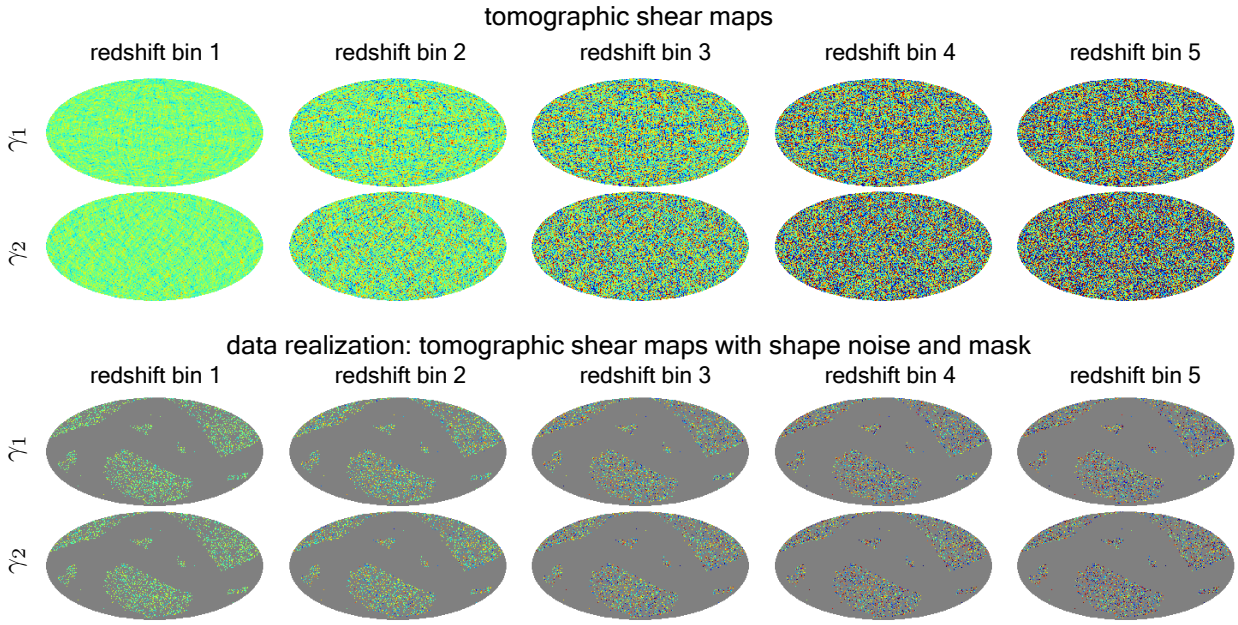

Figure 6. Schematic of the mock tomographic cosmic shear data. Top: realization of Gaussian tomographic cosmic shear fields (generated for the fiducial cosmology). The two components of the complex, spin-2 shear field are shown: $\gamma \equiv \gamma_1 + i\gamma_2$. Bottom: the realized maps but with shape noise and mask added. The shape noise levels and mask are taken for a Euclid-like survey. The maps are subsequently compressed down to a small set of summary statistics in two steps: firstly, maps are compressed to auto- and cross- angular (E-mode) power spectra, and these power spectra are then further compressed using approximate-score compression (§6.3).

simulations the total cost of performing DELFI was dominated by running the simulations rather than training the neural networks.

6.7 Discussion

The forward modelling assumptions described above are the same as those used in hierarchical modelling approaches to cosmic shear parameter inference (Alsing et al. 2015b, 2016); even in this simple demonstration we could make certain more principled assumptions about the statistical model for the data than standard Gaussian-likelihood cosmic shear analyses with relative ease. While the Bayesian hierarchical approach samples the likelihood of the noisy map data and infers the tomographic shear fields explicitly as a by-product, the likelihood-free approach analyzes the compressed data and implicitly marginalizes over the latent shear maps, targeting the posterior distribution of the cosmological parameters only. In this context, the likelihood-free analysis can be viewed as a fast alternative to sampling a full hierarchical model, with the caveat that some information may be lost in the data compression step(s). However, the likelihood-free framework will allow us to extend the forward model to describe the data at the catalog or image level – complexity that would quickly become intractable for hierarchical modelling approaches.

In this simple demonstration we made the simplifying assumptions of Gaussian shear fields and considered power spectra only in the first-level compression step. This can naturally be extended to non-Gaussian lensing simulations, and higher-order statistics added to the list of first-level summaries.

7 CASE STUDY III: IONIZING BACKGROUND FROM HIGH-Z LYMAN-α FORESTS

The Lyman-α (Lyα) forest at $z \sim 6$ measured from high redshift quasar spectra probes the ionizing background and thermal state of the intergalactic medium (IGM) around the end of the epoch of reionization (see McQuinn 2016 for a review). At these redshifts, the Universe is largely opaque to Lyα; the forest is characterized by narrow Lyα transmission spikes corresponding to small, low density regions (Oh & Furlanetto 2005), separated by extended Gunn-Peterson troughs (Gunn & Peterson 1965) where Lyα is completely absorbed (see Figure 8). The transmitted fraction of the quasar flux is given by $F \equiv e^{-\tau_{\mathrm{Ly}\alpha}}$, where the optical depth $\tau_{\mathrm{Ly}\alpha} \propto T^{-0.7}\Delta_b^2/\Gamma_{\mathrm{HI}}$ depends on the temperature T, gas density Δ_b and HI ionization rate Γ_{HI}. The statistics of the Lyα transmission spikes can hence be used to constrain the ionization rate (and thermal state), but the likelihood function for the observed flux transmission is intractable; likelihood-free inference is required to draw principled inferences from these data. For a recent application of ABC in this context, see Davies et al. (2017).

In this demonstration we will show how DELFI can be used to infer the ionization rate Γ_{HI} from observed segments of Lyα forest at $z \sim 6$, using hydrodynamical simulations to forward model the Lyα transmission. In this toy demonstration we will recover the HI ionization rate assuming a uniform ionizing background, fixed thermal state and consider Lyα only. However, we highlight that likelihood-free methods offer exciting new prospects for constraining poorly understood inhomogeneous reionization processes and thermal histories from high-z Lyα and Lyβ forest observations (Davies & Furlanetto 2016; D'Aloisio et al. 2017; Davies et al. 2017).

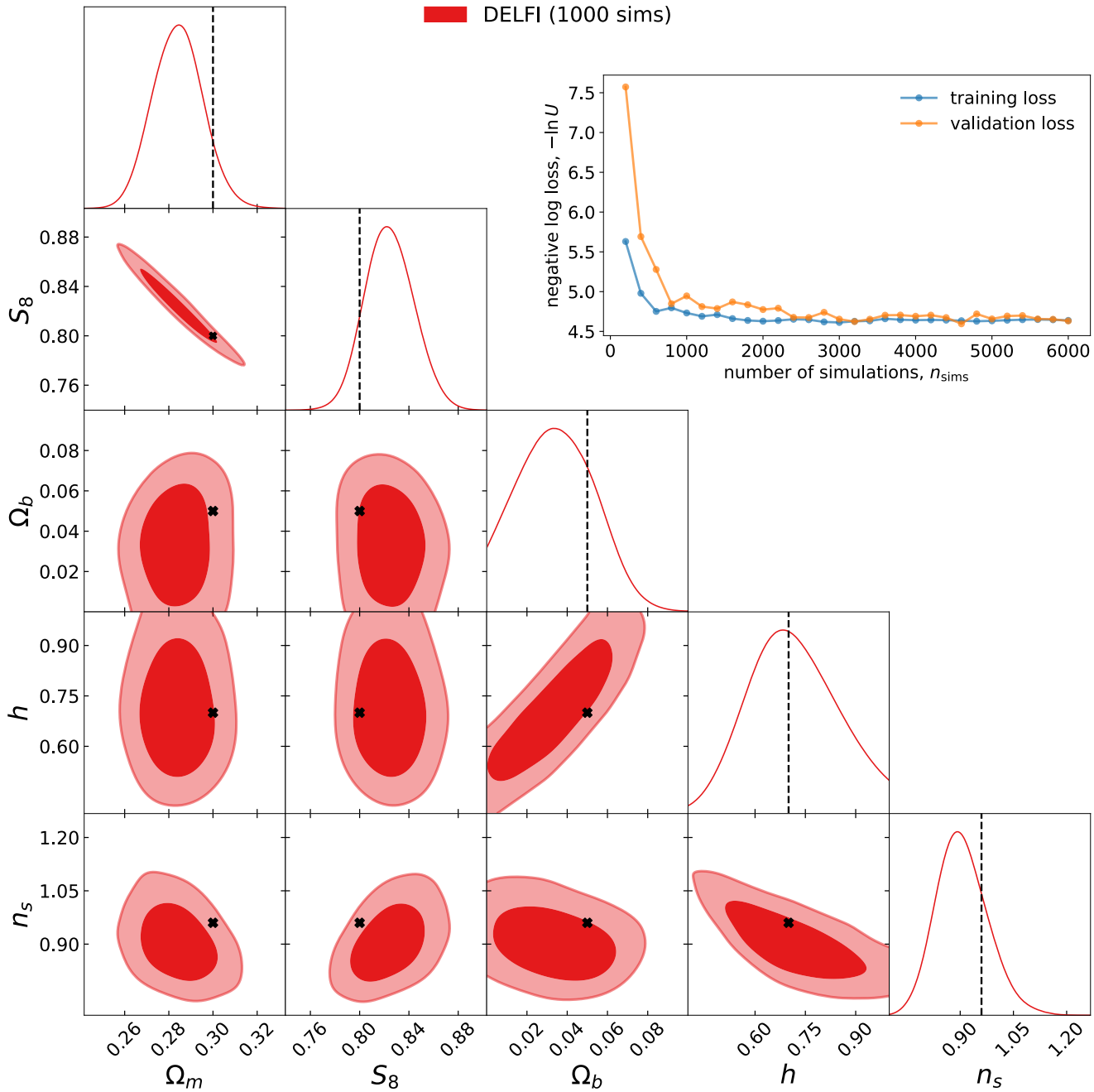

Figure 7. 68 and 95% credible regions of the 2D projections of the inferred DELFI posterior after 1000 simulations, for the cosmic shear tomographic pseudo-C_ℓ case study. Input parameters (black crosses) are well recovered, within uncertainties.

7.1 Data and simulations

We simulate mock Lyα forest segments for a given ionization rate $\Gamma_{\rm HI}$ using the Sherwood hydrodynamical simulation suite (Bolton et al. 2016), as follows:

(i) Generate a random skewer through a $z = 6$ snapshot of a 40Mpc$/h$ hydro-simulation box (from the Sherwood suite), ran with a fiducial ionization rate $\Gamma^*_{\rm HI} = 2.56 \cdot 10^{-13}{\rm s}^{-1}$ at $z = 6$ (see Bolton et al. 2016 for details of the Sherwood simulation set-up). The HI fraction is computed in (2048) cells along the line-of-sight, assuming ionization equilibrium, and the resulting Lyα transmission fraction F calculated (including the effects of peculiar motions and thermal broadening).

(ii) The transmission flux F is then re-scaled by $e^{-\Gamma^*_{\rm HI}/\Gamma_{\rm HI}}$ to impose the ionization rate we want to simulate. Note that in this simple demonstration we are fixing the instantaneous temperature and thermal history to their default values from the Sherwood suite, and also neglect large-scale fluctuations in $\Gamma_{\rm HI}$ that are expected to arise from inhomogeneous reionization.

(iii) Add zero mean Gaussian noise to the flux values, with standard deviation $\sigma = 0.01$.

Mock data are generated from the above simulation pipeline with

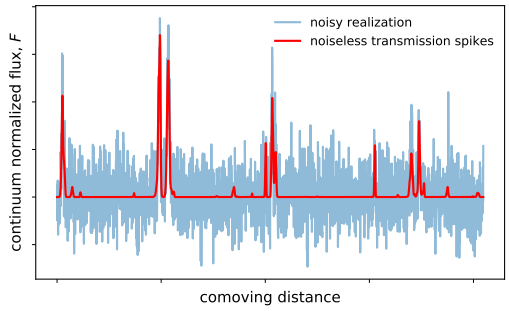

Figure 8. Typical example of Lyα forest transmission spikes at $z \sim 6$ (red) spanning comoving distance $40Mpc/h$, and the same forest segment with observational noise added.

a fiducial ionization rate $\Gamma^*_{\rm HI} = 2.56 \cdot 10^{-13}{\rm s}^{-1}$, show in Figure 8. The same pipeline is then used to generate forward simulations for inferring $\Gamma_{\rm HI}$ from those data using DELFI.

7.2 Data compression

In this simple demonstration we compress the flux data-vector in two stages: First, we compute fifty percentiles of the 2048 flux values, from 2 to 100 in steps of 2%. This is motivated by the notion that the most of the information about the ionization rate should be contained in the PDF of the flux values, which can be conveniently summarized by a set of percentiles.

We then compress the vector of percentiles down to a single summary statistic for $\Gamma_{\rm HI}$ using an IMNN. We use a fully-connected network with three dense layers with 128, 64 and 32 hidden units respectively, and leaky-ReLu activation functions with activation parameter $\alpha_{\rm ReLu} = 0.01$. For the training set we use 5000 simulations at the fiducial $\Gamma^*_{\rm HI}$, and an additional 5000 random-seed matched simulation pairs with $\Gamma_{\rm HI} = \Gamma^*_{\rm HI} \pm 1 \cdot 10^{-13}$ for the derivatives[10].

7.3 Priors

We take a uniform prior $\Gamma_{\rm HI} \in \left[0, 6 \cdot 10^{-13}\right]$ s^{-1}.

7.4 DELFI set-up

We ran DELFI using the SNL active learning scheme. We use an ensemble of five neural density estimators: five MDNs with 1–5 Gaussian components respectively, each with two hidden layers of 30 hidden units, and again we use tanh activations throughout. Simulations were run in batches of 100 for the SNL scheme, after an initial Fisher pre-training step to initialize the networks (using on the estimated Fisher matrix from the IMNN compression).

7.5 Results

In Figure 9 (left) shows the recovered posterior on the ionization rate $\Gamma_{\rm HI}$; the input value (red dashed vertical line) is well recovered. We find the DELFI ensemble of neural density estimators converges extremely fast in this case, after only $O(10^2)$ simulations (Figure 9; right).

8 CONCLUSIONS AND DISCUSSION

Density-estimation likelihood-free inference (DELFI) implemented using NDEs to learn the sampling distribution of the data (summaries) as a function of the model parameters, and adaptively acquiring simulations with active learning, provides an efficient framework for likelihood-free inference in cosmology. When combined with massive data compression, high-fidelity posteriors may be achieved from just $O(10^3)$ forward simulations for typical ~ 6 parameter inference tasks. Advances in nuisance-parameter hardened data compression mean that this expected performance may be preserved irrespective of the presence or number of additional nuisance parameters that need to be marginalized over (Alsing & Wandelt 2018a). Even without data compression, DELFI with NDEs and active learning provides a state-of-the-art framework for simulation-based inference (although more simulations will be required for larger, uncompressed data vectors).

We have introduced PYDELFI – a general purpose implementation of DELFI with NDEs and active learning (and data compression) – available with tutorials and documentation at `https://github.com/justinalsing/pydelfi`. PYDELFI opens up new possibilities for likelihood-free analyses of complex cosmological data sets, using rich generative models containing physical and observational effects that would otherwise be challenging or impossible to include accurately into a traditional likelihood-based analysis.

For standard inference tasks where the form of the likelihood-function can be assumed known, we note that PYDELFI can actually be faster (and more accurate for given resources) than MCMC sampling. By turning the inference problem into a low-dimensional density-estimation task, DELFI effectively builds a fast neural network emulator for the likelihood-function, in a similar spirit to variational inference. We have shown that this can converge quickly, after just $O(10^3)$ simulations for typical problems, which are typically similar in cost to likelihood evaluations (for simple likelihoods). Meanwhile, MCMC methods would typically require many more likelihood calls to yield well sampled posteriors, for the same number of model parameters. The number of simulations to attain convergence for DELFI in these simple cases may be minimized by using neural density estimators that correspond exactly to the form of the known likelihood, eg., a Gaussian with parameter dependent mean and fixed covariance matrix.

An emerging trend in cosmology is to build emulators for summary statistics for which no robust analytical model exists, such as the non-linear matter power spectrum on small scales (Heitmann et al. 2013), 21cm power spectrum (Schmit & Pritchard 2017; Kern

[10] Note that given a hydrosimulation box, generating realizations of Lyα segments by taking skewers through the box is relatively inexpensive. We therefore made no attempt to reduce/optimize the number of simulations needed to train the IMNN in this case study. We leave detailed exploration of optimal compression of Lyα forsets (eg., without pre-compression to percentiles, optimal IMNN architectures, etc) to future work.

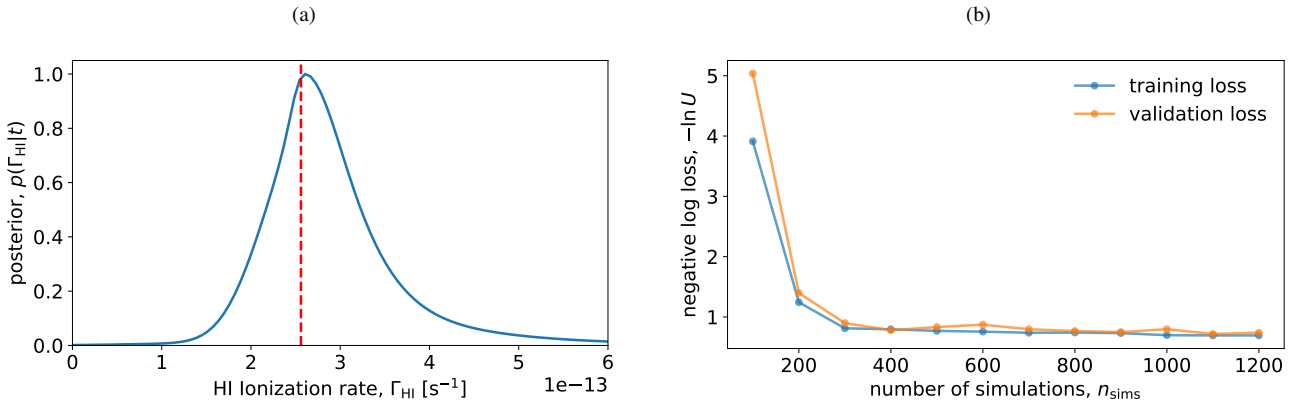

Figure 9. Left: Recovered posterior for the ($z = 6$) HI ionization rate from the high-z Lyα forest, from DELFI after 300 simulations (mock data shown in Figure 8). Right: Convergence of the DELFI NDE ensemble for the ionizing background inference task. The DELFI ensemble of NDEs converges extremely quickly in this low-dimensional case, after only $O(10^2)$ simulations.

et al. 2017), Lyman-α power spectrum (Rogers et al. 2018; Bird et al. 2018), weak lensing Minkowski functionals (Marques et al. 2018), and many others. DELFI has a deep connection to emulation methods. Emulators in cosmology have been mostly concerned with learning the expectation value of some summary statistics as a function of the model parameters, which would then typically be plugged into a standard Gaussian likelihood analysis with an estimated covariance matrix. DELFI goes further and builds an emulator for the sampling distribution of the summary statistics, as a function of the model parameters, thereby addressing the expectation emulation and inference tasks in one go and without resorting to restrictive or ad hoc likelihood assumptions in the inference step.

Likelihood-free inference also has a deep connection to Bayesian hierarchical modelling (BHM) approaches to cosmological data analysis. BHMs specify a generative model for the data, which in turn defines a joint likelihood for the hyper-parameters (eg., cosmological and global nuisance parameters) and some latent variables (for example, initial potential fluctuations, true properties and redshifts of individual objects in a survey, etc). These typically high-dimensional likelihoods are then sampled using MCMC (or otherwise), and inference of both the hyper-parameters and latent-variables reported. BHMs and likelihood-free methods are of the same spirit in that they both aim to do inference under as complete a generative model description for the data as possible. However, sampling high-dimensional BHMs for complex forward models is hard and computationally intensive work, and there are often limitations on how rich the implemented models can be in practice. Here likelihood-free methods have a clear advantage over BHMs; simulating forwards is much easier than solving the inverse problem with MCMC sampling or otherwise, and adding extra complexity to the forward model has virtually no impact on the difficulty of the inference task for likelihood-free methods (other than any added cost of running simulations). On the other hand, while likelihood-free methods may rely on data compression to be tractable for high-dimensional data vectors and expensive simulators, sampling methods can target the posterior for the uncompressed data directly and yield inferences of the latent variables as a (potentially useful) by-product.

By relying entirely on forward simulations, the likelihood-free approach marks a shift in the way observational cosmology is done in practice. The scientific effort is reduced to: (1) taking data, (2) building as faithful a forward model and simulation pipeline as possible for those data, and (3) if necessary, devising some data compression scheme to reduce the number of simulations required to achieve accurate posteriors with LFI. Activities that typically make up a large part of traditional cosmological data analysis efforts—constructing and calibrating intermediate estimators, building and validating approximate likelihoods, computing accurate covariance matrices, *etc.*—no longer enter into the critical path[11] of scientific reasoning (although they may still be relevant for optimal data compression, but without the same onerous requirements on accuracy as for likelihood-based methods). All critical assumptions underpinning the analysis are then concisely and completely summarized by the forward model specification; this makes for robust science, and clear and simple scientific reporting.

ACKNOWLEDGEMENTS

This work is supported by the Simons Foundation. Justin Alsing was partially supported by the research project grant “Fundamental Physics from Cosmological Surveys" funded by the Swedish Research Council (VR) under Dnr 2017-04212. Benjamin Wandelt acknowledges support by the Labex Institut Lagrange de Paris (ILP) (reference ANR-10-LABX-63) part of the Idex SUPER, and received financial state aid managed by the Agence Nationale de la Recherche, as part of the programme Investissements d’avenir under the reference ANR-11-IDEX-0004-02. Tom Charnock is supported by the ANR BIG4 grant ANR-16-CE23-0002 of the French Agence Nationale de la Recherche and would like to thank NVIDIA for the donation of the Quadro P6000 used in building and testing PYDELFI.

REFERENCES

Abadi M., et al., 2015, TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems, `http://tensorflow.org/`

Aghanim N., et al., 2018, arXiv preprint arXiv:1807.06209

Akeret J., Refregier A., Amara A., Seehars S., Hasner C., 2015, Journal of Cosmology and Astroparticle Physics, 2015, 043

[11] "Critical" in the sense that if they are not done accurately enough, the resulting scientific inferences may be biased.

Alsing J., Wandelt B. D., 2018a, in prep.
Alsing J., Wandelt B., 2018b, Monthly Notices of the Royal Astronomical Society: Letters, 476, L60
Alsing J., Kirk D., Heavens A., Jaffe A. H., 2015a, Monthly Notices of the Royal Astronomical Society, 452, 1202
Alsing J., Heavens A., Jaffe A. H., Kiessling A., Wandelt B., Hoffmann T., 2015b, Monthly Notices of the Royal Astronomical Society, 455, 4452
Alsing J., Heavens A., Jaffe A. H., 2016, Monthly Notices of the Royal Astronomical Society, 466, 3272
Alsing J., Wandelt B. D., Feeney S. M., 2018a, arXiv preprint arXiv:1808.06040
Alsing J., Wandelt B., Feeney S., 2018b, Monthly Notices of the Royal Astronomical Society, 477, 2874
Betoule M. e. a., et al., 2014, Astronomy & Astrophysics, 568, A22
Bird S., Rogers K. K., Peiris H. V., Verde L., Font-Ribera A., Pontzen A., 2018, arXiv preprint arXiv:1812.04654
Bishop C. M., 1994, Technical report, Mixture density networks. Citeseer
Bishop C. M., 2006, Pattern recognition and machine learning. springer
Bolton J. S., Puchwein E., Sijacki D., Haehnelt M. G., Kim T.-S., Meiksin A., Regan J. A., Viel M., 2016, Monthly Notices of the Royal Astronomical Society, 464, 897
Bonassi F. V., You L., West M., 2011, Statistical applications in genetics and molecular biology, 10
Brehmer J., Cranmer K., Louppe G., Pavez J., 2018b, arXiv preprint arXiv:1805.00013
Brehmer J., Cranmer K., Louppe G., Pavez J., 2018c, arXiv preprint arXiv:1805.00020
Brehmer J., Louppe G., Pavez J., Cranmer K., 2018a, arXiv preprint arXiv:1805.12244
Burden F., Winkler D., 2008, in , Artificial neural networks. Springer, pp 23–42
Cameron E., Pettitt A., 2012, Monthly Notices of the Royal Astronomical Society, 425, 44
Carassou S., de Lapparent V., Bertin E., Borgne D. L., 2017, arXiv preprint arXiv:1704.05559
Charnock T., Lavaux G., Wandelt B. D., 2018, Physical Review D, 97, 083004
Cooray A., Hu W., 2001, The Astrophysical Journal, 548, 7
D'Aloisio A., McQuinn M., Davies F. B., Furlanetto S. R., 2017, Monthly Notices of the Royal Astronomical Society, 473, 560
Davies F. B., Furlanetto S. R., 2016, Monthly Notices of the Royal Astronomical Society, 460, 1328
Davies F. B., Hennawi J. F., Eilers A.-C., Lukić Z., 2017, arXiv preprint arXiv:1703.10174
Dikov G., van der Smagt P., Bayer J., 2019, arXiv preprint arXiv:1901.04436
Duncan C. A. J., Joachimi B., Heavens A. F., Heymans C., Hildebrandt H., 2013, Monthly Notices of the Royal Astronomical Society, 437, 2471
Durkan C., Papamakarios G., Murray I., 2018, arXiv preprint arXiv:1811.08723
Fan Y., Nott D. J., Sisson S. A., 2013, Stat, 2, 34
Fluri J., Kacprzak T., Lucchi A., Refregier A., Amara A., Hofmann T., 2018a, arXiv preprint arXiv:1807.08732
Fluri J., Kacprzak T., Sgier R., Réfrégier A., Amara A., 2018b, arXiv preprint arXiv:1803.08461
Gelman A., Stern H. S., Carlin J. B., Dunson D. B., Vehtari A., Rubin D. B., 2013, Bayesian data analysis. Chapman and Hall/CRC
Germain M., Gregor K., Murray I., Larochelle H., 2015, in International Conference on Machine Learning. pp 881–889
Gillet N., Mesinger A., Greig B., Liu A., Ucci G., 2018, arXiv preprint arXiv:1805.02699
Gorski K. M., Hivon E., Banday A., Wandelt B. D., Hansen F. K., Reinecke M., Bartelmann M., 2005, The Astrophysical Journal, 622, 759
Gunn J. E., Peterson B. A., 1965, The Astrophysical Journal, 142, 1633
Gupta A., Matilla J. M. Z., Hsu D., Haiman Z., 2018, Physical Review D, 97, 103515
Gutmann M. U., Corander J., 2016, The Journal of Machine Learning Research, 17, 4256
Hahn C., Vakili M., Walsh K., Hearin A. P., Hogg D. W., Campbell D., 2017, Monthly Notices of the Royal Astronomical Society, 469, 2791
Harnois-Déraps J., van Waerbeke L., Viola M., Heymans C., 2015, Monthly Notices of the Royal Astronomical Society, 450, 1212
Heavens A. F., Jimenez R., Lahav O., 2000, Monthly Notices of the Royal Astronomical Society, 317, 965
Heavens A., Alsing J., Jaffe A., 2013, arXiv.org
Heitmann K., Lawrence E., Kwan J., Habib S., Higdon D., 2013, The Astrophysical Journal, 780, 111
Higson E., Handley W., Hobson M., Lasenby A., 2018, Monthly Notices of the Royal Astronomical Society, 483, 4828
Hildebrandt H., van Waerbeke L., Erben T., 2009, preprint, 507, 683
Hildebrandt H., et al., 2013, Monthly Notices of the Royal Astronomical Society, astro-ph.CO, 488
Hinton S., et al., 2018, arXiv preprint arXiv:1811.02381
Hu W., 1999, ApJ, 522, L21
Hu W., 2002, Phys. Rev. D, 65, 023003
Ishida E., et al., 2015, Astronomy and Computing, 13, 1
Järvenpää M., Gutmann M. U., Pleska A., Vehtari A., Marttinen P., et al., 2018, Bayesian Analysis
Jennings E., Wolf R., Sako M., 2016, arXiv preprint arXiv:1611.03087
Joachimi B., et al., 2015, Space Science Reviews, 193, 1
Kacprzak T., Herbel J., Amara A., Réfrégier A., 2017, arXiv preprint arXiv:1707.07498
Kaiser N., 1992, The Astrophysical Journal, 388, 272
Kaiser N., 1998, The Astrophysical Journal, 498, 26
Kannawadi A., et al., 2018, arXiv e-prints
Kern N. S., Liu A., Parsons A. R., Mesinger A., Greig B., 2017, The Astrophysical Journal, 848, 23
Kilbinger M., 2015, Reports on Progress in Physics, 78, 086901
Kingma D. P., Ba J., 2014, arXiv preprint arXiv:1412.6980
Kitching T. D., Alsing J., Heavens A. F., Jimenez R., McEwen J. D., Verde L., 2017, Monthly Notices of the Royal Astronomical Society, 469, 2737
Kratochvil J. M., Haiman Z., May M., 2010, Physical Review D, 81, 043519
Krause E., Hirata C. M., 2010, Astronomy & Astrophysics, 523, A28
Lakshminarayanan B., Pritzel A., Blundell C., 2017, in Advances in Neural Information Processing Systems. pp 6402–6413
Laureijs R., et al., 2011, preprint, astro-ph.CO
Leclercq F., 2018, arXiv preprint arXiv:1805.07152
Limber D. N., 1954, The Astrophysical Journal, 119, 655
Lin C.-A., Kilbinger M., 2015, Astronomy & Astrophysics, 583, A70
Lintusaari J., Gutmann M. U., Dutta R., Kaski S., Corander J., 2017, Systematic biology, 66, e66
Lueckmann J.-M., Goncalves P. J., Bassetto G., Öcal K., Nonnenmacher M., Macke J. H., 2017, in Advances in Neural Information Processing Systems. pp 1289–1299
Lueckmann J.-M., Bassetto G., Karaletsos T., Macke J. H., 2018, arXiv preprint arXiv:1805.09294
Mandel K. S., Wood-Vasey W. M., Friedman A. S., Kirshner R. P., 2009, The Astrophysical Journal, 704, 629
Mandelbaum R., 2018, Annual Review of Astronomy and Astrophysics, 56, 393
March M., Trotta R., Berkes P., Starkman G., Vaudrevange P., 2011, Monthly Notices of the Royal Astronomical Society, 418, 2308
Marques G. A., Liu J., Matilla J. M. Z., Haiman Z., Bernui A., Novaes C. P., 2018, arXiv preprint arXiv:1812.08206
Massey R., et al., 2012, Monthly Notices of the Royal Astronomical Society, 429, 661
McQuinn M., 2016, Annual Review of Astronomy and Astrophysics, 54, 313
Merten J., Giocoli C., Baldi M., Meneghetti M., Peel A., Lalande F., Starck J.-L., Pettorino V., 2018, arXiv preprint arXiv:1810.11027
Oh S. P., Furlanetto S. R., 2005, The Astrophysical Journal Letters, 620, L9
Papamakarios G., Murray I., 2016, in Advances in Neural Information Processing Systems. pp 1028–1036
Papamakarios G., Murray I., Pavlakou T., 2017, in Advances in Neural Information Processing Systems. pp 2338–2347
Papamakarios G., Sterratt D. C., Murray I., 2018, arXiv preprint arXiv:1805.07226

Ravanbakhsh S., Oliva J. B., Fromenteau S., Price L., Ho S., Schneider J. G., Póczos B., 2016, in ICML. pp 2407–2416
Ribli D., Pataki B. Á., Csabai I., 2018, arXiv preprint arXiv:1806.05995
Roberts E., Lochner M., Fonseca J., Bassett B. A., Lablanche P.-Y., Agarwal S., 2017, Journal of Cosmology and Astroparticle Physics, 2017, 036
Robin A., Reylé C., Fliri J., Czekaj M., Robert C., Martins A., 2014, Astronomy & Astrophysics, 569, A13
Rogers K. K., Peiris H. V., Pontzen A., Bird S., Verde L., Font-Ribera A., 2018, arXiv preprint arXiv:1812.04631
Rubin D., et al., 2015, The Astrophysical Journal, 813, 137
Rudd D. H., Zentner A. R., Kravtsov A. V., 2008, The Astrophysical Journal, 672, 19
Salvato M., Ilbert O., Hoyle B., 2018, arXiv preprint arXiv:1805.12574
Schafer C. M., Freeman P. E., 2012, in , Statistical Challenges in Modern Astronomy V. Springer, pp 3–19
Schmit C. J., Pritchard J. R., 2017, Monthly Notices of the Royal Astronomical Society, 475, 1213
Sellentin E., Heymans C., Harnois-Déraps J., 2018, Monthly Notices of the Royal Astronomical Society, 477, 4879
Shariff H., Jiao X., Trotta R., van Dyk D. A., 2016, The Astrophysical Journal, 827, 1
Simola U., Pelssers B., Barge D., Conrad J., Corander J., 2018, arXiv preprint arXiv:1810.09930
Smyth P., Wolpert D., 1998, in Advances in neural information processing systems. pp 668–674
Smyth P., Wolpert D., 1999, Machine Learning, 36, 59
Srivastava N., Hinton G., Krizhevsky A., Sutskever I., Salakhutdinov R., 2014, The Journal of Machine Learning Research, 15, 1929
Takada M., Jain B., 2004, MNRAS, 348, 897
Tassev S., Zaldarriaga M., Eisenstein D. J., 2013, Journal of Cosmology and Astroparticle Physics, 2013, 036
Tegmark M., Taylor A. N., Heavens A. F., 1997, The Astrophysical Journal, 480, 22
Tripp R., 1998, Astronomy and Astrophysics, 331, 815
Uria B., Côté M.-A., Gregor K., Murray I., Larochelle H., 2016, The Journal of Machine Learning Research, 17, 7184
Weyant A., Schafer C., Wood-Vasey W. M., 2013, The Astrophysical Journal, 764, 116
Zablocki A., Dodelson S., 2016, Physical Review D, 93, 083525
van Waerbeke L., 2010, MNRAS, 401, 2093

This paper has been typeset from a TEX/LATEX file prepared by the author.