Automatic physical inference with information maximizing neural networks

Tom Charnock[*]
Sorbonne Université, CNRS, UMR 7095, Institut d'Astrophysique de Paris, 98 bis boulevard Arago, 75014 Paris, France

Guilhem Lavaux[†]
Sorbonne Université, CNRS, UMR 7095, Institut d'Astrophysique de Paris, 98 bis boulevard Arago, 75014 Paris, France and Sorbonne Universités, Institut Lagrange de Paris, 98 bis boulevard Arago, 75014 Paris, France

Benjamin D. Wandelt[‡]
Center for Computational Astrophysics, Flatiron Institute, 162 5th Avenue, New York, New York 10010, USA; Sorbonne Université, CNRS, UMR 7095, Institut d'Astrophysique de Paris, 98 bis boulevard Arago, 75014 Paris, France; Sorbonne Universités, Institut Lagrange de Paris, 98 bis boulevard Arago, 75014 Paris, France; and Department of Astrophysical Sciences, 4 Ivy Lane, Princeton University, Princeton, New Jersey 08544, USA

(Received 10 February 2018; published 13 April 2018)

Compressing large data sets to a manageable number of summaries that are informative about the underlying parameters vastly simplifies both frequentist and Bayesian inference. When only simulations are available, these summaries are typically chosen heuristically, so they may inadvertently miss important information. We introduce a simulation-based machine learning technique that trains artificial neural networks to find nonlinear functionals of data that maximize Fisher information: information maximizing neural networks (IMNNs). In test cases where the posterior can be derived exactly, likelihood-free inference based on automatically derived IMNN summaries produces nearly exact posteriors, showing that these summaries are good approximations to sufficient statistics. In a series of numerical examples of increasing complexity and astrophysical relevance we show that IMNNs are robustly capable of automatically finding optimal, nonlinear summaries of the data even in cases where linear compression fails: inferring the variance of Gaussian signal in the presence of noise, inferring cosmological parameters from mock simulations of the Lyman-α forest in quasar spectra, and inferring frequency-domain parameters from LISA-like detections of gravitational waveforms. In this final case, the IMNN summary outperforms linear data compression by avoiding the introduction of spurious likelihood maxima. We anticipate that the automatic physical inference method described in this paper will be essential to obtain both accurate and precise cosmological parameter estimates from complex and large astronomical data sets, including those from LSST and Euclid.

DOI: 10.1103/PhysRevD.97.083004

I. INTRODUCTION

Current data analysis techniques in astronomy and cosmology often involve reducing large data sets into a collection of sufficient statistics [1–3]. There are several methods for condensing raw data to a set of summaries. Amongst others, these methods could be: principal component analysis (PCA) [4–8]; statistics including the mean, covariance, and higher point functions [9,10]; or calculating the autocorrelation or power spectrum [10,11]. Unfortunately, summaries calculated using the above methods can still be infeasibly large for data-space comparison. For example, analysis of weak lensing data from the Euclid and the Large Synoptic Survey Telescope (LSST) photometric surveys will have around 10^4 summary statistics [12]. Reducing the number of summaries further results in enormous losses in the information available in the raw data [12].

Another popular way of summarizing data is using the Massively Optimized Parameter Estimation and Data (MOPED) compression algorithm [2]. Summaries from MOPED are linear combinations of data that compress the number of data points down to the number of parameters of

[*]charnock@iap.fr
[†]lavaux@iap.fr
[‡]wandelt@iap.fr

a model describing the data. MOPED is completely lossless when noise in the data is independent of the parameters and when the likelihood is, at least to first order, Gaussian [2]. The MOPED algorithm has been used on many problems in astronomy and cosmology such as studying the star formation histories of galaxies [13–15], analyzing the cosmic microwave background [16,17], and identifying transients [18] to name but a few. Unfortunately, using linear combinations of the data for compression may not be optimal for maximizing the possible information available, even when the likelihood is known [19].

For many astronomical and cosmological problems, it can become impossibly difficult to write a likelihood function which describes, not only physics, but also includes any selection bias and instrumental effects. Recently, methods have become available to perform inference when a likelihood is not available via approximate Bayesian computation (ABC). ABC is a technique which allows samples to be drawn from an approximate posterior distribution. Forward simulations are first created using parameter values drawn from a prior and samples are accepted or rejected by comparing the *distance* of the simulation to the real data. To efficiently approach the true posterior distribution, it is convenient to couple ABC with a sampling procedure such as population Monte Carlo (PMC). ABC using PMC (PMC-ABC) is a method to obtain approximate parameter distributions by iterating through weighted samples from the prior [20,21] and can massively reduce the number of samples which need to be drawn during ABC.

Likelihood-free inference has been used for a variety of astronomical problems which include deducing quasar luminosity functions [22], understanding early time galaxy merger rate evolution [23], constraining cosmological parameters with supernova observations [24], interpreting galaxy formation [25], searching for the connection between galaxies and halos [26], measuring cosmological redshift distributions [27], inferring photometric evolution of galaxies [28], and calculating the ionizing background using the Lyman-α and Lyman-β forest transmission [29]. Each of the above examples are used in conjunction with publicly available (PMC-)ABC codes [30–32].

A two-step compression algorithm was defined in [33] that is capable of optimally summarizing data while preserving information when the likelihood is not known. The first step involves extracting informative statistics from raw data (or simulations of the data) heuristically, i.e. perhaps using the power spectrum or using PCA. The summaries of the simulations contain information about physics, selection bias, and the instrument. A second step then assumes an asymptotic likelihood to perform compression from the summaries gathered in the first step down to the number of parameters in the model as in MOPED or [19]. The choice of likelihood in the second step does not bias the inference of model parameters during ABC, although the compression will be closer to optimal by choosing a better likelihood function.

However, what if there is information in the data that we did not think to summarize in a first-step summary? In this paper we introduce the concept of *information maximizing neural networks* (IMNNs). Through the use of machine learning, we can circumvent the two-step compression used in [33] and find the most informative nonlinear data summaries by training a neural network using the Fisher information matrix as a reward function. In fact, if we already know some informative summaries, such as those calculated in the first step of [33], we can use the IMNN to calculate summaries of the data which optimally increase the information further and then including the IMNN summaries amongst the first-step summaries.

Once the network is trained, ABC proceeds as before. Model parameters can be drawn from a prior, used to generate simulations and once they are fed through the network, the IMNN summaries of the simulation can be compared to the summaries of the real data. Samples can then be accepted or rejected given the distance of the network summary of the simulation to the network summary of the real data to build the approximate posterior distribution of model parameters. The IMNN provides a framework to perform automatic physical inference simply by producing simulations.

In Sec. II we describe how to calculate the Fisher information matrix and how linear summaries of the data can conserve Fisher information using the MOPED algorithm. In Sec. III we lay out the procedure for creating nonlinear summaries of the data. An overview of how artificial neural networks work is presented in Sec. IV and we continue in Sec. V by showing how maximizing the determinant of the Fisher information matrix allows a network to be trained to provide the optimal nonlinear set of summaries. Next, in Sec. VI, we trace the steps to obtain parameter constraints from PMC-ABC using the network trained as prescribed in Sec. V. Finally, in Sec. VII, we give some test examples. The first test model provides an example where a single linear summary of the data would provide nearly no information about a parameter, but the nonlinear summary provided by a trained artificial neural network can extract the maximum information the data contains. The second example is more astronomically motivated, using the absorption of flux from quasars by neutral hydrogen to constrain the amplitude of scalar perturbations. Finally we use the network to summarize and constrain the central oscillation frequency of a gravitational wave burst from Laser Interferometer Space Antenna (LISA). This problem was used in [34] to show that MOPED compression introduces spurious maxima in the posterior distribution; we show that the nonlinear IMNN data compression introduced in this paper can avoid this peculiarity.

II. FISHER INFORMATION AND LINEAR COMPRESSION

A likelihood function $\mathcal{L}(\mathbf{d}|\boldsymbol{\vartheta})$ of some data, $\mathbf{d}$, with $n_\mathbf{d}$ data points, is informative about a model with a set of $n_{\boldsymbol{\vartheta}}$ parameters, $\boldsymbol{\vartheta}$. The more sharply peaked $\mathcal{L}(\mathbf{d}|\boldsymbol{\vartheta})$ is at a particular value of $\boldsymbol{\vartheta}$, the better $\boldsymbol{\vartheta}$ is known. The Fisher information describes how much information $\mathbf{d}$ contains about the linear parameters, $\boldsymbol{\vartheta}$, and can be calculated by finding the second moment of the score of the likelihood [35–37], i.e. the variance of the partial derivative of the natural logarithm of the likelihood with respect to the parameters at a fiducial parameter value, $\boldsymbol{\vartheta}^{\rm fid}$,

$$\mathbf{F}_{\alpha\beta}(\boldsymbol{\vartheta}) = \int d\mathbf{d}\mathcal{L}(\mathbf{d}|\boldsymbol{\vartheta})\frac{\partial \ln\mathcal{L}(\mathbf{d}|\boldsymbol{\vartheta})}{\partial\vartheta_\alpha}\frac{\partial \ln\mathcal{L}(\mathbf{d}|\boldsymbol{\vartheta})}{\partial\vartheta_\beta}\bigg|_{\boldsymbol{\vartheta}=\boldsymbol{\vartheta}^{\rm fid}} = \left\langle\frac{\partial \ln\mathcal{L}(\mathbf{d}|\boldsymbol{\vartheta})}{\partial\vartheta_\alpha}\frac{\partial \ln\mathcal{L}(\mathbf{d}|\boldsymbol{\vartheta})}{\partial\vartheta_\beta}\right\rangle\bigg|_{\boldsymbol{\vartheta}=\boldsymbol{\vartheta}^{\rm fid}}. \tag{2.1}$$

Equation (2.1) can be rewritten as

$$\mathbf{F}_{\alpha\beta}(\boldsymbol{\vartheta}) = -\left\langle\frac{\partial^2 \ln\mathcal{L}(\mathbf{d}|\boldsymbol{\vartheta})}{\partial\vartheta_\alpha\partial\vartheta_\beta}\right\rangle\bigg|_{\boldsymbol{\vartheta}=\boldsymbol{\vartheta}^{\rm fid}} \tag{2.2}$$

when the likelihood is twice continuously differentiable [36–38]. A large Fisher information for a given set of data indicates that the data is informative about the parameters and therefore the parameters can be measured more effectively [38]. In particular, the minimum variance of an estimator of a parameter, $\boldsymbol{\vartheta}$, is given by the Cramér-Rao bound [39,40], which states that

$$\langle(\vartheta_\alpha - \langle\vartheta_\alpha\rangle)(\vartheta_\beta - \langle\vartheta_\beta\rangle)\rangle \geq (\mathbf{F}^{-1})_{\alpha\beta}, \tag{2.3}$$

such that finding the maximum Fisher information provides the minimum variance for estimators of $\boldsymbol{\vartheta}$. Note that the Cramér-Rao inequality only holds under certain conditions, i.e. that the score function is defined for all $\mathbf{d}$ in the support of the likelihood and that differentiation and taking the expectation commute. The Cramér-Rao bound limits the second moment of any estimator, but does not limit the shape of the confidence regions [41]. In the case that the likelihood of the data in a particular model is Gaussian, the logarithm of the likelihood can be written as

$$-2\ln\mathcal{L}(\mathbf{d}|\boldsymbol{\vartheta}) = (\mathbf{d}-\mu(\boldsymbol{\vartheta}))^T\mathbf{C}^{-1}(\mathbf{d}-\mu(\boldsymbol{\vartheta})) + \ln|2\pi\mathbf{C}|, \tag{2.4}$$

where $\mathbf{d}$ is the data and $\mu(\boldsymbol{\vartheta})$ is the mean of the model given parameters $\boldsymbol{\vartheta}$, which we will denote μ for convenience. $\mathbf{C}$ is the covariance of the data and is assumed to be independent of the parameters. Using the MOPED algorithm [2], $\mathbf{d}$ can be compressed from the number of points in the data, $n_\mathbf{d}$, to the number of parameters of the model, $n_{\boldsymbol{\vartheta}}$, simply by seeking the linear combination of data which optimizes the linearized parameters. The MOPED compression is lossless in the sense that the Fisher information is conserved under the transformation

$$x_\alpha = \mathbf{r}_\alpha^T\mathbf{d} \tag{2.5}$$

where α labels the parameter and $\mathbf{r}_\alpha$ is calculated by maximizing the Fisher information ensuring that $\mathbf{r}_\alpha$ is orthogonal to $\mathbf{r}_\beta$ (where $\alpha \neq \beta$). The form of $\mathbf{r}_\alpha$ is

$$\mathbf{r}_1 = \frac{\mathbf{C}^{-1}\boldsymbol{\mu}_{,1}}{\sqrt{\boldsymbol{\mu}_{,1}^T\mathbf{C}^{-1}\boldsymbol{\mu}_{,1}}}, \tag{2.6}$$

for the first parameter, ϑ_1, and where $\partial/\partial\vartheta_\alpha \equiv {}_{,\alpha}$. For each parameter afterwards,

$$\mathbf{r}_\alpha = \frac{\mathbf{C}^{-1}\boldsymbol{\mu}_{,\alpha} - \sum_{i=1}^{\alpha-1}(\boldsymbol{\mu}_{,\alpha}^T\mathbf{r}_i)\mathbf{r}_i}{\sqrt{\boldsymbol{\mu}_{,\alpha}^T\mathbf{C}^{-1}\boldsymbol{\mu}_{,\alpha} - \sum_{i=1}^{\alpha-1}(\boldsymbol{\mu}_{,\alpha}^T\mathbf{r}_i)^2}}. \tag{2.7}$$

After creating the linear summaries, $\mathbf{x} = \{x_\alpha|\alpha\in[1,n_{\boldsymbol{\vartheta}}]\}$, $\mathbf{x}$ is as informative about $\boldsymbol{\vartheta}$ as $\mathbf{d}$ is with regards to the Fisher information, for the likelihood in Eq. (2.4). The Fisher information takes the form

$$\mathbf{F}_{\alpha\beta} = \mathrm{Tr}[\boldsymbol{\mu}_{,\alpha}^T\mathbf{C}^{-1}\boldsymbol{\mu}_{,\beta}]. \tag{2.8}$$

The lossless compression of the data, $\mathbf{d}\to\mathbf{x}$, is only possible when the likelihood is exactly of the form in Eq. (2.4). Nearly lossless compression is still possible if the peak of the likelihood is approximately Gaussian. Often, this will be a good approximation in the asymptotic limit, i.e., when the data are informative about the parameters.

III. NONLINEAR FISHER INFORMATION MAXIMIZING SUMMARIES

We are influenced by the MOPED algorithm to find some transformation which maps the data to compressed summaries, $f: \mathbf{d}\to\mathbf{x}$, while conserving Fisher information, but without the limitation that the method is only valid as a Gaussian approximation. f is a function that modifies the original likelihood describing the data, which need not be known *a priori*, into the form

$$-2\ln\mathcal{L}(\mathbf{x}|\boldsymbol{\vartheta}) = (\mathbf{x}-\mu_f(\boldsymbol{\vartheta}))^T\mathbf{C}_f^{-1}(\mathbf{x}-\mu_f(\boldsymbol{\vartheta})) \tag{3.1}$$

where

$$\mu_f(\boldsymbol{\vartheta}) = \frac{1}{n_\mathbf{s}}\sum_{i=1}^{n_\mathbf{s}}\mathbf{x}_i^\mathbf{s} \tag{3.2}$$

is the mean value of $n_\mathbf{s}$ summaries, $\{\mathbf{x}_i^\mathbf{s}|i\in[1,n_\mathbf{s}]\}$, where each summary is obtained from a simulation $\mathbf{d}_i^\mathbf{s} = \mathbf{d}^\mathbf{s}(\boldsymbol{\vartheta}, i)$

using $\mathscr{f}:\mathbf{d}_i^{\mathrm{s}} \to \mathbf{x}_i^{\mathrm{s}}$. We will denote $\mu_{\mathscr{f}}(\boldsymbol{\vartheta}) \equiv \boldsymbol{\mu}_{\mathscr{f}}$ for convenience. Each i denotes a different random initialization of a simulation. Similarly $\mathbf{C}_{\mathscr{f}}^{-1}$ is the inverse of the covariance matrix which is again obtained from simulations of the data

$$(\mathbf{C}_{\mathscr{f}})_{\alpha\beta} = \frac{1}{n_{\mathrm{s}}-1}\sum_{i=1}^{n_{\mathrm{s}}}(\mathbf{x}_i^{\mathrm{s}} - \boldsymbol{\mu}_{\mathscr{f}})_\alpha(\mathbf{x}_i^{\mathrm{s}} - \boldsymbol{\mu}_{\mathscr{f}})_\beta. \tag{3.3}$$

Using Eq. (2.2) a modified Fisher information matrix can be calculated from the likelihood in Eq. (3.1)

$$\mathbf{F}_{\alpha\beta} = \mathrm{Tr}[\boldsymbol{\mu}_{\mathscr{f},\alpha}^T \mathbf{C}_{\mathscr{f}}^{-1}\boldsymbol{\mu}_{\mathscr{f},\beta}]. \tag{3.4}$$

Here, the values of $\boldsymbol{\mu}_{\mathscr{f},\alpha}$ and $\mathbf{C}_{\mathscr{f}}^{-1}$ are calculated using fixed, fiducial parameter values, $\boldsymbol{\vartheta}^{\mathrm{fid}}$, such that the simulations are $\mathbf{d}_i^{\mathrm{s\,fid}} = \mathbf{d}^{\mathrm{s}}(\boldsymbol{\vartheta}^{\mathrm{fid}}, i)$. Although $\mathscr{f}:\mathbf{d} \to \mathbf{x}$ is not specified, a subclass of $\mathscr{f}$ is accessible via a neural network, described in detail in Sec. IV. We will show how this function can be found by training a neural network in Sec. V.

IV. ARTIFICIAL NEURAL NETWORKS

Artificial neural networks are arbitrary maps from some inputs to outputs. Consider some data vector $\mathbf{d} = \{d_i | i \in [1, n_{\mathbf{d}}]\}$ with $n_{\mathbf{d}}$ data points. Each data point is regarded as an input to a network. For a deep neural network, a series of *hidden layers* are able to learn levels of abstraction from the input [42–46]. Each layer, l, of the network contains a set of *neurons* which takes some number of inputs and provides one output per neuron [47,48]. The output a neuron is *activated* by a nonlinear activation function

$$a_i^l = \phi(v_i^l) \tag{4.1}$$

where

$$v_j^l = \sum_i w_{ji}^l a_i^{l-1} + b_j^l, \tag{4.2}$$

is a weighted, biased input at each layer with weights $\boldsymbol{w}^l \equiv w_{ji}^l$ and biases $\boldsymbol{b}^l \equiv b_j^l$ [47]. i describes an element of the output vector of a collection of neurons in the $(l-1)^{\mathrm{th}}$ layer and j indexes the neuron in layer l. With these notations, the input to the network can be considered to be the output of a zeroth layer of a network, $d_i \equiv a_i^0$. Stacking several neurons into a hidden layer and stacking several hidden layers, taking the outputs from the previous layer as the inputs to each node in the next layer, allows for greater levels of abstraction from the input data [44]. These networks are often referred to as *deep* networks. Note that the addition of too many layers can lead to expensive computations and overfitting by the network so that it becomes difficult to train. The network output at the final layer can be described by $\boldsymbol{a}^L = \{a_i^L | i \in [1, n_{\mathrm{outputs}}]\}$ where n_{outputs} is the number of outputs in the final layer, labeled L, and $a_i^L = \phi(v_i^L)$.

As mentioned at the end of Sec. III, a neural network can be used as a representation of $\mathscr{f}:\mathbf{d} \to \mathbf{x}$, which compresses data to summary statistics. Formally, this subclass of functions is described, for some input $\mathbf{z}$, by

$$\mathscr{f}^l:\mathbf{z} \to \boldsymbol{a}^l = \phi\left(\sum_i w_{ji}^l[\mathscr{f}^{l-1}(\mathbf{z})]_i + b_j^l\right), \tag{4.3}$$

for $l > 0$ and

$$\mathscr{f}^0:\mathbf{z} \to \boldsymbol{a}^0 = \mathbf{m}, \tag{4.4}$$

where the compressed summary is given at $l = L$ of the recursion and the input to the function at $l = 0$ is taken to be the identity.

A. Activation functions

The activation function, $\phi(v_i^l)$, in Eq. (4.1) describes whether the artificial neuron *fires* or not, i.e. whether the inputs are informative or useful for describing the output [43,46,49,50]. It is the activation function that provides the nonlinearity necessary for the network to learn the complex map from inputs to outputs by combining the relevant combinations of inputs at each layer in a nontrivial way. As long as there are enough hidden layers, the form of the activation function is relatively unimportant since the weights and biases will be trained to combine the outputs of each hidden layer in such a way as to provide the correct map. There are many options for the choice of activation function, including tanh and sigmoid functions. Currently popular activation functions are the rectified linear unit (ReLU) [49]. We show here, as an example, an adaptation called leaky ReLU

$$\phi(x) = \begin{cases} \alpha x & x \le 0 \\ x & x > 0 \end{cases}, \tag{4.5}$$

where $\alpha = 0$ for ReLU and α is small and positive for leaky ReLU [51]. Although the ReLU family of activation functions are linear, stacking several layers of neurons provides a function which approximates a nonlinear function, and is extremely quick to calculate. It will become apparent that the derivative of the activated output with respect to the weighted, biased inputs are essential for training neural networks. The derivative of the ReLU family of activation functions can also be efficiently calculated as

$$\frac{\partial\phi(x)}{\partial x} = \begin{cases} \alpha & x \le 0 \\ 1 & x > 0 \end{cases}. \tag{4.6}$$

Although we have shown ReLU as an example, we explore various activation functions across the population of networks that we train.

B. Back propagation

A scalar loss function, $\Lambda(\boldsymbol{a}^L)$, is calculated from the outputs of the network $\boldsymbol{a}^L$. In supervised deep learning, the loss function describes how far the outputs are from a set of labels for the training data [52]. An iterative procedure, called back propagation, uses the chain rule to find how much the weights and biases need to change to minimize the loss function [52]. Using gradient descent [53] it can be seen that the weights and biases must be updated using

$$w^l_{ji} \to w^l_{ji} - \eta \frac{\partial \Lambda}{\partial w^l_{ji}} \tag{4.7}$$

and

$$b^l_i \to b^l_i - \eta \frac{\partial \Lambda}{\partial b^l_i}, \tag{4.8}$$

where η is a tunable learning rate which dictates the size of the steps that the weights and biases are able to take on each update [46]. It is very efficient to calculate the derivatives in Eqs. (4.7) and (4.8) at the last layer using

$$\frac{\partial \Lambda}{\partial v^L_i} = \frac{\partial \Lambda}{\partial a^L_i} \frac{\partial a^L_i}{\partial v^L_i}. \tag{4.9}$$

From any layer, the rate of change of the loss function with respect to the weighted, biased inputs at the previous layer can be found using

$$\frac{\partial \Lambda}{\partial v^l_i} = \sum_j w^{l+1}_{ji} \frac{\partial \Lambda}{\partial v^{l+1}_j} \frac{\partial a^l_i}{\partial v^l_i}. \tag{4.10}$$

The changes in the loss function under changes in the weights or the biases are then calculated using

$$\begin{aligned}\frac{\partial \Lambda}{\partial w^l_{ji}} &= \frac{\partial \Lambda}{\partial v^l_j} \frac{\partial v^l_j}{\partial w^l_{ji}} \\ &= \frac{\partial \Lambda}{\partial v^l_j} a^{l-1}_i \end{aligned} \tag{4.11}$$

and

$$\begin{aligned}\frac{\partial \Lambda}{\partial b^l_i} &= \frac{\partial \Lambda}{\partial v^l_i} \frac{\partial v^l_i}{\partial b^l_i} \\ &= \frac{\partial \Lambda}{\partial v^l_i}. \end{aligned} \tag{4.12}$$

Each of the a^l_i and the derivatives with respect to the weighted biased inputs [using Eq. (4.6)] are calculated on the forward pass of the network inputs. Back propagation allows the change of the loss function with respect to all of the weights or biases to be calculated in just one pass forward and one pass backwards [46]. By calculating the change in the loss function with respect to the network outputs and applying Eq. (4.9) successively, the weight and bias updates at every layer can be calculated easily.

The back propagation procedure is repeated many times using different sets of training inputs [46]. Once all of the training inputs are used, one epoch of training is complete. After one epoch of training, the order of the training inputs can be jumbled and the training procedure repeated many times until the loss function is minimized [46].

C. Overfitting

It is possible that the network weights become tuned to features in the training data which are not present in the real data. To prevent this *overfitting*, we implement *dropout* [54]. Dropout is a technique where a random fraction of the neurons are set to zero on each batch of training and after back propagation only the weights and biases of the active neurons are updated. Performing dropout during training equates to training many subnetworks, where all the neurons share weights and biases. Each of the subnetworks can learn specific features in the data, but the consensus network does not learn features too strongly.

D. Training and test data sets

When training a network, it is essential to test how well the network is learning by using a *test* set which contains data which is not present in the training set. However, it is extremely important to note that even the accuracy of prediction on the test set should not be considered to be a measure of the predictive ability of the network. It is considered normal to tune a network to achieve the minimal loss of the test set without showing signs of overfitting. A *third*, completely unseen, data set should then be used to quote network accuracies. In doing so, the irreproducible accuracy scores often quoted in the literature, arising from only considering a network that is highly tuned on the test set, are avoided. In this paper we train and test networks with a training set and a test set and use the comparison between the posterior distribution obtained using the network output and the analytically calculated distribution as our confirmation that the network is accurate.

V. FINDING NONLINEAR SUMMARIES

Inspired by supervised artificial neural networks we are able to create a network capable of maximizing the Fisher information to create nonlinear summaries of data. The output of the network, $\mathbf{x} \equiv \boldsymbol{a}^L$ is a compressed summary of some data, $\mathbf{d}$. Since the data is a function of some parameters, $\boldsymbol{\vartheta}$, given some model, the summary can be described as a function of these parameters, as well as the weights and biases at each layer, l, of a network, $\mathbf{x} \to \mathbf{x}(\boldsymbol{\vartheta}, \boldsymbol{w}^l, \boldsymbol{b}^l)$. The mean, $\boldsymbol{\mu}_f$, covariance, $\mathbf{C}_f$ and Fisher information matrix, $\mathbf{F}_{\alpha\beta}$, from Eqs. (3.2), (3.3) and (3.4), each become functions of the weights and biases as well. Summaries of simulations, $\mathbf{x}_i^{\mathrm{s}}$, are obtained by passing simulations, $\mathbf{d}_i^{\mathrm{s}}$, through the network $f{:}\,\mathbf{d}_i^{\mathrm{s}} \to \mathbf{x}_i^{\mathrm{s}}$.

To compute the Fisher information matrix in Eq. (3.4), the derivative of the network needs to be calculated with respect to the parameters at fiducial values. It is, in principle, simple to find the derivative of the network with respect to the parameters due to partial derivatives commuting with sums

$$\begin{aligned}\boldsymbol{\mu}_{f,\alpha} &= \frac{\partial}{\partial\vartheta_\alpha}\frac{1}{n_{\mathbf{s}}}\sum_{i=1}^{n_{\mathbf{s}}}\mathbf{x}_i^{\mathbf{s}\,\mathrm{fid}}\\ &= \frac{1}{n_{\mathbf{s}}}\sum_{i=1}^{n_{\mathbf{s}}}\left(\frac{\partial\mathbf{x}}{\partial\vartheta_\alpha}\right)_i^{\mathbf{s}\,\mathrm{fid}}. \end{aligned} \tag{5.1}$$

Unfortunately, since the parameters only appear in the simulations, numerical differentiation needs to be performed. The numerical differentiation is achieved by producing three copies of the simulation, $\mathbf{d}_i^{\mathbf{s}\,\mathrm{fid}} = \mathbf{d}^{\mathrm{s}}(\boldsymbol{\vartheta}^{\mathrm{fid}}, i)$, $\mathbf{d}_i^{\mathbf{s}\,\mathrm{fid}-} = \mathbf{d}^{\mathrm{s}}(\boldsymbol{\vartheta}^{\mathrm{fid}} - \Delta\boldsymbol{\vartheta}^-, i)$, and $\mathbf{d}_i^{\mathbf{s}\,\mathrm{fid}+} = \mathbf{d}^{\mathrm{s}}(\boldsymbol{\vartheta}^{\mathrm{fid}} + \Delta\boldsymbol{\vartheta}^+, i)$ where $\Delta\boldsymbol{\vartheta}^\pm$ is some small deviation from the fiducial parameter value. The derivative of the network output with respect to the parameters is therefore given by

$$\left(\frac{\partial\mathbf{x}}{\partial\vartheta_\alpha}\right)_i^{\mathbf{s}\,\mathrm{fid}} \approx \frac{\mathbf{x}_i^{\mathbf{s}\,\mathrm{fid}+} - \mathbf{x}_i^{\mathbf{s}\,\mathrm{fid}-}}{\Delta\vartheta_\alpha^+ - \Delta\vartheta_\alpha^-}. \tag{5.2}$$

Setting the random seed, i, to the same value when generating $\mathbf{d}_i^{\mathbf{s}\,\mathrm{fid}-}$ and $\mathbf{d}_i^{\mathbf{s}\,\mathrm{fid}+}$ suppresses the sample variance in estimates of the derivative of the mean. Although the network output can vary a lot between different simulations, the derivative with respect to parameters is much more stable to changes in the parameter value, meaning relatively few extra simulations ($n_{\partial\vartheta} < n_{\mathbf{s}}$) need to be computed to calculate the gradient of the mean.

Another way of calculating the derivative of the mean of the network output is to calculate the adjoint gradient of the simulations, and calculate the derivative of the network with respect to the simulations

$$\boldsymbol{\mu}_{f,\alpha} = \frac{1}{n_{\mathbf{s}}}\sum_{i=1}^{n_{\mathbf{s}}}\sum_{k=1}^{n_{\mathbf{d}}}\frac{\partial x_{ik}^{\mathbf{s}\,\mathrm{fid}}}{\partial d_k}\frac{\partial d_{ik}^{\mathbf{s}\,\mathrm{fid}}}{\partial\vartheta_\alpha}, \tag{5.3}$$

where i labels the random initialization of the simulation and k labels the data point in the simulation. In certain situations, calculating the adjoint gradient of the simulations may be more efficient than the method described in Eqs. (5.1) and (5.2).

One simple way of obtaining the optimal nonlinear summary from some data is to maximize the determinant of the Fisher information matrix calculated from the network, $|\mathbf{F}|$,

$$\Lambda = -\frac{1}{2}|\mathbf{F}|^2. \tag{5.4}$$

The Fisher information matrix terms are produced from the second derivatives of the Kullback-Leibler divergence, i.e. the information gain, and is hence directly related to the Shannon entropy [55]. In particular, the Fisher information matrix is the Shannon information of the Gaussian probability distribution function which optimally approximates the likelihood in (3.1) near its peak. For this reason, choosing to maximize the determinant of the Fisher information is equivalent to maximizing the Shannon information of this distribution. The error is then found by taking the derivative of the loss function with respect to the network output. Normally $\mathbf{x}_i^{\mathrm{s}}$ would be considered as the network output when the input is a simulation, but since the quantity of interest in our problem is statistically calculated over a large number of network outputs, we follow the cartoon in Fig. 1 and use the determinant of the Fisher information matrix as the true network output. First, a large number of simulations at fixed fiducial parameter value and random initialization (as well as the simulations created to calculate the derivative of the mean) are fed forwards through identical networks. All the network outputs from the fixed fiducial parameter simulations are used to calculate the covariance as in Eq. (3.3). Meanwhile, the rest of the network outputs are used to find the derivative of the mean with respect to the parameter as in Eqs. (5.1) and (5.2). These are combined to give the Fisher information matrix of Eq. (3.4). If we consider the *true* network output to be $\boldsymbol{a}^L = |\mathbf{F}|$ rather than $\mathbf{x}^{\mathrm{s}}$ then the error can be defined as

$$\frac{\partial\Lambda}{\partial\boldsymbol{a}^L} = -|\mathbf{F}|. \tag{5.5}$$

Training then commences over many epochs of weight and bias updates until the Fisher information stops increasing. In practice, a problem arises when using Eq. (5.5), since the Fisher information is invariant under linear scaling of the summary. To control the magnitude of the summaries we can artificially induce a scale by adding the determinant of the covariance matrix, $|\mathbf{C}_f|$, to the error function

$$\frac{\partial\Lambda}{\partial\boldsymbol{a}^L} = -|\mathbf{F}| + |\mathbf{C}_f|. \tag{5.6}$$

The network is penalized when the determinant of the covariance is large. When using Eq. (5.6) the network provides the summary which maximizes the Fisher information while minimizing the covariance of the outputs.

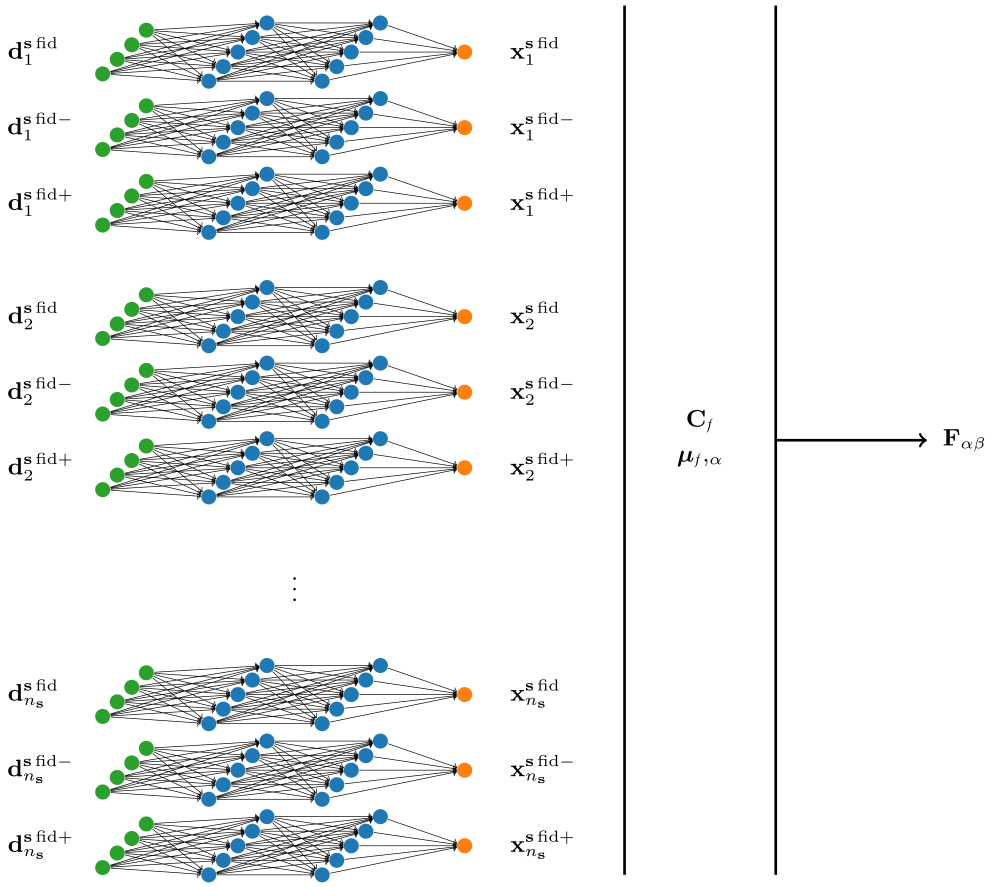
$\mathbf{d}_1^{\mathbf{s}\,\mathrm{fid}}$
$\mathbf{d}_1^{\mathbf{s}\,\mathrm{fid}-}$
$\mathbf{d}_1^{\mathbf{s}\,\mathrm{fid}+}$
$\mathbf{d}_2^{\mathbf{s}\,\mathrm{fid}}$
$\mathbf{d}_2^{\mathbf{s}\,\mathrm{fid}-}$
$\mathbf{d}_2^{\mathbf{s}\,\mathrm{fid}+}$
$\mathbf{d}_{n_\mathbf{s}}^{\mathbf{s}\,\mathrm{fid}}$
$\mathbf{d}_{n_\mathbf{s}}^{\mathbf{s}\,\mathrm{fid}-}$
$\mathbf{d}_{n_\mathbf{s}}^{\mathbf{s}\,\mathrm{fid}+}$
$\mathbf{x}_1^{\mathbf{s}\,\mathrm{fid}}$
$\mathbf{x}_1^{\mathbf{s}\,\mathrm{fid}-}$
$\mathbf{x}_1^{\mathbf{s}\,\mathrm{fid}+}$
$\mathbf{x}_2^{\mathbf{s}\,\mathrm{fid}}$
$\mathbf{x}_2^{\mathbf{s}\,\mathrm{fid}-}$
$\mathbf{x}_2^{\mathbf{s}\,\mathrm{fid}+}$
$\mathbf{x}_{n_\mathbf{s}}^{\mathbf{s}\,\mathrm{fid}}$
$\mathbf{x}_{n_\mathbf{s}}^{\mathbf{s}\,\mathrm{fid}-}$
$\mathbf{x}_{n_\mathbf{s}}^{\mathbf{s}\,\mathrm{fid}+}$
$\mathbf{C}_f$
$\boldsymbol{\mu}_{f,\alpha}$
$\mathbf{F}_{\alpha\beta}$

FIG. 1. Cartoon of the information maximizing neural network architecture. During training, each simulation $\mathbf{d}_i^{\mathrm{s,fid}}$ and each simulation made with a varied fiducial parameter, $\mathbf{d}_i^{\mathrm{s\,fid}\pm}$, is passed through the same network (all the weights and biases are shared). The output of the network for each simulation, $\mathbf{x}_i^{\mathrm{s\,fid}}$, is used to calculate the covariance, $\mathbf{C}_f$, and each of the network outputs from the varied simulations, $\mathbf{x}_i^{\mathrm{s\,fid}\pm}$, are used to calculate the derivative of the mean, $\boldsymbol{\mu}_{f,\alpha}$. The network uses $\partial\Lambda/\partial\boldsymbol{a}^L = -|\mathbf{F}| + |\mathbf{C}_f|$ as the error of a reward function which is maximized through back propagation. The reward function is back propagated only through a selection of networks which use the simulations created at the fiducial parameter value, $\mathbf{x}_i^{\mathrm{s\,fid}}$. The weights and biases are updated using the mean of the back propagated error at each weight and bias, $\partial\Lambda/\partial\boldsymbol{w}^l$ and $\partial\Lambda/\partial\boldsymbol{b}^l$. Once trained, a summary of some data can be obtained using a simple artificial neural network with the weights and biases from the training network.

Although the network is capable of extracting all necessary summaries of the data without any prior knowledge of what the parameters represent, we can imagine the IMNNs would be better suited to extending the heuristic first-step summaries. For example, if the power spectrum is a known useful summary of some data, the network can be trained to find any statistic which increases the Fisher information further. With the power spectrum and the network summary, a second stage compression as described in [19] can be used for efficient parameter inference. This way, inexhausted information of the data can be unlocked, even when the form of the data combination that probes it is not known.

VI. APPROXIMATE BAYESIAN COMPUTATION

ABC is a technique of finding an approximate posterior distribution for some model parameters by accepting or rejecting samples dependent on how similar simulations created using the sample parameters are to the real data [56]. It is useful to choose an appropriate sampling procedure to quickly approach the true posterior for the parameters without creating too many simulations. PMC is an algorithm by which samples can be obtained by iterating through weighted draws from a prior, even when the likelihood is not accessible [57]. Although PMC has a variety of uses, such as filtering, we are going to couple it to ABC (PMC-ABC) to effectively approach the true posterior [20,21].

Similar to the method in [31], our PMC-ABC algorithm starts by drawing N parameter vectors, $\{\boldsymbol{\vartheta}_k^t | k \in [1, N], t = 0\}$, from the prior, $p(\boldsymbol{\vartheta})$. N is the final number of posterior samples wanted, k labels the sample and t describes the number of sampling iterations. In each sampling iteration, samples are drawn from a prior, used to create simulations, and then weighted by the distance of the simulation from the real data. The weighted samples are used to obtain a new proposal distribution with which to resample from in the next iteration. This allows the PMC-ABC to gradually hone in on the true probability distribution. Simulations are made at each of the N parameter vectors and fed through the trained network to obtain a collection of network summaries $\{\mathbf{x}^{\mathbf{s}t}_{ik}|k \in [1, N]\}$ where i labels the simulation. Only the value of $\boldsymbol{\vartheta}$ is important for ABC and so the random initialization, i, can be ignored once chosen for each simulation. We choose to define the distance of each simulated summary from the summary of the real data $\mathbf{x}$ by

$$\varrho^t_k = \sqrt{(\mathbf{x}^{\mathbf{s}t}_{ik} - \mathbf{x})^T \mathbf{F} (\mathbf{x}^{\mathbf{s}t}_{ik} - \mathbf{x})}, \qquad (6.1)$$

where $\mathbf{F}$ is the Fisher information matrix obtained originally by the network. Equation (6.1) is the optimal distance measure [19], although it is not unique. On each iteration, an acceptance condition, ϵ^t, for the samples is defined by the 75th percentile of $\{\varrho^t_k|k \in [1, N]\}$ such that the 75% of samples which have the smallest distances from the summary of the real data are kept. $\boldsymbol{\vartheta}^t_k$ then corresponds to the remaining 25% of the samples, which are used to draw parameter vectors for the next iteration, $\boldsymbol{\vartheta}^{t+1}_k$. $\boldsymbol{\vartheta}^{t+1}_k$ are selected from a Gaussian with mean $\boldsymbol{\vartheta}^t_k$ and covariance, $\mathbf{C}_t$, from the weighted parameter values. The weighting for $\boldsymbol{\vartheta}^{t+1}_k$ is given by

$$W^{t+1}_k = \frac{p(\boldsymbol{\vartheta}^{t+1}_k)}{\sum_{j=1}^N W^t_j \mathcal{N}(\boldsymbol{\vartheta}^{t+1}_k; \boldsymbol{\vartheta}^t_j, \mathbf{C}_t)} \qquad (6.2)$$

with $p(\boldsymbol{\vartheta}^{t+1}_k)$ as the value of the prior at $\boldsymbol{\vartheta}^{t+1}_k$,

$$\mathcal{N}(\boldsymbol{\vartheta}^{t+1}_k; \boldsymbol{\vartheta}^t_j, \mathbf{C}_t) = \frac{\exp\left[-\frac{1}{2}(\boldsymbol{\vartheta}^{t+1}_k - \boldsymbol{\vartheta}^t_j)^T \mathbf{C}^{-1}_t (\boldsymbol{\vartheta}^{t+1}_k - \boldsymbol{\vartheta}^t_j)\right]}{\sqrt{|2\pi \mathbf{C}_t|}} \qquad (6.3)$$

and where the initial weighting is equal for all k, $W^0_k = 1/N$. $\boldsymbol{\vartheta}^{t+1}_k$ is drawn repeatedly from the Gaussian with mean $\boldsymbol{\vartheta}^t_k$ and covariance $\mathbf{C}_t$ until $\varrho^{t+1}_k \leq \epsilon^t$ for each of the rejected k samples. Once complete, the first iteration of sampling finishes, allowing W^{t+1}_k to be calculated.

Unlike the method in [31] the accepted $\boldsymbol{\vartheta}^t_k$ are instantly promoted to $\boldsymbol{\vartheta}^{t+1}_k$ rather than being redrawn. The accepted ϱ^t_k can also be promoted to ϱ^{t+1}_k, and the new $\boldsymbol{\vartheta}^{t+1}_k$ used to find $\mathbf{C}_{t+1}$. The next acceptance condition, ϵ^{t+1}, is again calculated from the 75th percentile of $\{\varrho^{t+1}_k|k \in [1, N]\}$ and the selection procedure is repeated. Iterations can be performed until the number of draws from $\mathcal{N}(\boldsymbol{\vartheta}^t_k, \mathbf{C}_t)$ in a particular iteration, t, is much larger than the number of wanted samples from the posterior, N. A large number of draws compared to the number of accepted parameter values is a sign that the approximate posterior has stopped changing considerably between iterations.

VII. TESTING INFERENCE WITH INFORMATION MAXIMISING NEURAL NETWORKS

In this section we use the information maximizing neural network on a range of test models. In Sec. VII A we use the network to summarize a Gaussian signal with unknown variance, as well as Gaussian signal with unknown variance that was contaminated by noise, first of known variance and then of unknown variance. We consider the same problem in Sec. VII B showing that the network provides nearly optimal, informative summaries in spite of a poorly chosen fiducial parameter value by learning the correct map. In Sec. VII C we constrain the amplitude of scalar perturbations using simulations of quasar absorption spectra which can be summarized by a single statistic provided by the network. Finally, in Sec. VII D, we demonstrate the performance of IMNN compression for the case estimating the central frequency of a LISA gravitational wave chirp. This example addresses a concern raised in [34] where the authors show that a linear summary of data in the time domain can be misleading about a parameter in the frequency domain. We are show that the nonlinear summary avoids this problem and is more informative.

A. Summarizing Gaussian signals

A simple toy model can be constructed where linear combinations of the data are unable to provide information about parameters.

Consider an experiment which measures $n_\mathbf{d} = 10$ data points which are drawn from a zero-mean Gaussian where the variance, $\vartheta = \sigma^2$, is not perfectly known, $\mathbf{d} = \{d_i \curvearrowleft \mathcal{N}(0, \vartheta)|i \in [1, n_\mathbf{d}]\}$. The likelihood is written

$$\mathcal{L}(\mathbf{d}|\vartheta) = \prod_{i=1}^{n_\mathbf{d}} \frac{1}{\sqrt{2\pi\vartheta}} \exp\left[-\frac{1}{2\vartheta} d_i^2\right] = \frac{1}{(2\pi\vartheta)^{n_\mathbf{d}/2}} \exp\left[-\frac{1}{2\vartheta} \sum_{i=1}^{n_\mathbf{d}} d_i^2\right], \qquad (7.1)$$

such that

$$-2\ln\mathcal{L}(\mathbf{d}|\vartheta) = \frac{1}{\vartheta}\sum_{i=1}^{n_\mathbf{d}} d_i^2 + n_\mathbf{d} \ln[2\pi\vartheta]. \qquad (7.2)$$

From here it can be seen that a single number, the sum of the square of the data

$$x = \sum_{i=1}^{n_\mathbf{d}} d_i^2, \qquad (7.3)$$

is a minimal sufficient statistic. Maximizing the (logarithm of the) likelihood with respect to the variance relates the value of the statistic to the variance

$$\frac{\partial \ln \mathcal{L}(x|\vartheta)}{\partial \vartheta} = \frac{x}{2\vartheta^2} - \frac{n_{\mathbf{d}}}{2\vartheta} = 0 \tag{7.4}$$

so that

$$x = n_{\mathbf{d}}\vartheta. \tag{7.5}$$

The Fisher information is calculated using Eq. (2.2)

$$\mathbf{F} = \frac{x}{(\vartheta^{\rm fid})^3} - \frac{n_{\mathbf{d}}}{2(\vartheta^{\rm fid})^2} = \frac{n_{\mathbf{d}}}{2(\vartheta^{\rm fid})^2}. \tag{7.6}$$

For $n_{\mathbf{d}} = 10$ and a fiducial variance of $\vartheta^{\rm fid} = 1$ the Fisher information is

$$\mathbf{F} = 5. \tag{7.7}$$

Since the single summary is a nonlinear combination (squared sum) of the data, linear combinations will not be able to provide a single sufficient statistic.

Now consider training a network to maximize the Fisher information while summarizing the data, as laid out in Sec. V. We show the progress an example network makes until it extracts the full information in Fig. 2.

The fully connected network has two hidden layers with 256 neurons in each. We denote this configuration [256, 256]. The network uses leaky ReLU activation with $\alpha = 0.01$ and a learning rate of $\eta = 0.01$. Each of the weights, $\boldsymbol{w}^l$, are initialized with a value drawn from a normal distribution with mean $\mu = 0$ and standard deviation $\sigma = \sqrt{2/\kappa^{l-1}}$ where κ^l is the number of neurons in layer l [49]. As is usual when using the ReLU family of activation functions, the biases $\boldsymbol{b}^l$ are initialized with a slightly positive value [58], where $\boldsymbol{b}^l = 0.1$ has been chosen here. To mimic the small number of simulations which would be available for complex data sets, we limit the total number of simulations to 1000 (+100 simulations created above and below the fiducial parameter value to calculate the numerical derivatives). These are divided into $n_{\rm train} = 2$ training batches per epoch, such that $n_{\mathbf{s}} = 500$ and $n_{\partial\vartheta} = 50$. The training batches are split to provide variation in the statistical quantities $\boldsymbol{\mu}_{\ell,\alpha}$ and $\mathbf{C}_{\ell}$ when jumbling the simulations at the beginning of each epoch of training. We train the network for 800 epochs. To prevent overfitting, where the network learns features in the training set which are not present in the test data, 50% of the neurons are dropped from the network on each batch of training.

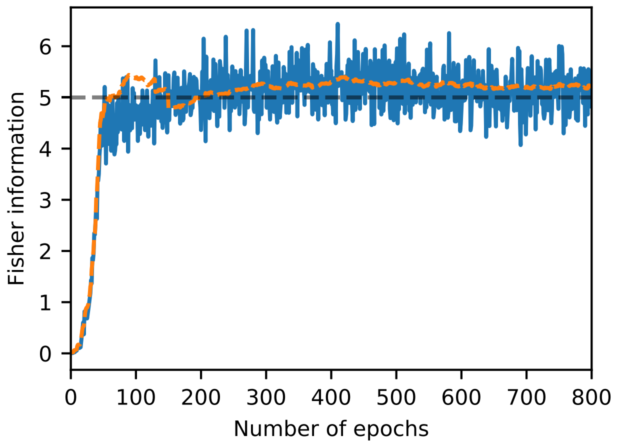

FIG. 2. Value of the Fisher information obtained by the network at the end of each epoch of training. The solid blue line shows the Fisher information obtained by running a set of 500 simulations (and 50 partial derivatives), which are contained in the training set, through the network. The dashed orange line shows the Fisher information obtained by running the same number of simulations through the network, but where none of the simulations are present in the training set. The maximum amount of Fisher information expected is $\mathbf{F} = 5$, shown as a black dashed line. It is clear that the network manages to extract the entirety of the information given the data.

From Eq. (7.7), it can be seen that the maximum Fisher information attainable for this problem is $\mathbf{F} = 5$. Figure 2 shows that $\mathbf{F} = 5.15 \pm 0.39$ is obtained by the network over the last 10% of the training epochs. The solid blue line in Fig. 2 is the value of the Fisher information obtained from the network summaries of $n_{\mathbf{s}} = 500$ and $n_{\partial\vartheta} = 50$ simulations from the training set (a single batch with no dropout), while the dashed orange line is the same for simulations which are not contained in the training set. We find that we are able to obtain a Fisher information slightly above $\mathbf{F} = 5$ as indicated by the straight black dashed line. This is because the data sets fluctuated to have a smaller variance than $\vartheta = 1$ and therefore the Fisher information for these sets is higher than their expectation. The network interprets the fluctuation in the data as an indication that more information about the parameters is available from the network than is truly available.

We have found that a very large variety of hyper-parameters will provide us with approximately $\mathbf{F} = 5$. Most notably we can use very deep networks with few neurons such as [5, 5, 5, 5, 5] to extremely simple networks with large numbers of neurons, i.e. [2048, 2048], each with very similar outcomes. The main difference with different architectures, that we have found, is the number of epochs necessary to maximize the Fisher information matrix. We have chosen a simple network of [256, 256] since it seems to converge more quickly than other networks.

Since the test model can be written down analytically, the true posterior distribution for some simulated test data $\mathbf{d}$ (shown in Table I) can be found and is plotted as the solid orange curve in Fig. 3. The prior distribution used here is uniform between $\vartheta = (0, 10]$. A first approximation of the posterior distribution using the network, without creating any additional simulations can be found using

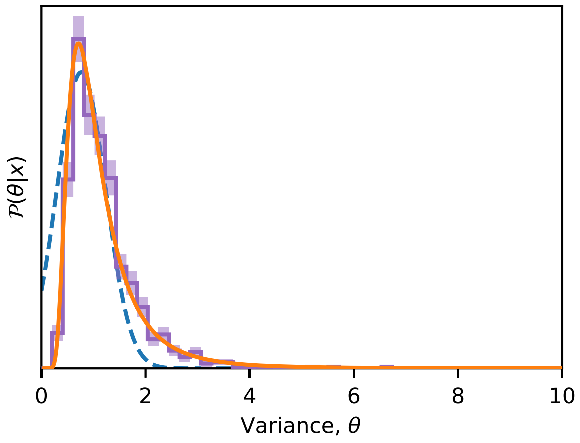

FIG. 3. The posterior distribution for the variance of the real data. The solid orange curve is the analytic posterior distribution using Bayes' theorem and the likelihood in Eq. (7.1). The dashed blue curve shows the posterior calculated from the asymptotic likelihood from the network summary and in purple is the ABC posterior obtained through PMC with the purple shaded error bars showing the 1-σ Poisson width. Each distribution is normalized such that its integral is unity in the interval $\vartheta = [0, 10]$. We can see that the analytic posterior in the solid orange curve overlaps the PMC-ABC posterior in the purple histogram showing that the network has successfully learned to summarize the data. The blue dashed curve peaks at the same place as the analytic posterior with a similar width, which shows that the first order approximation of the posterior is also correct.

the asymptotic likelihood by expanding Eq. (3.1) about the fiducial variance with $\Delta\vartheta = (-1, 9]$.[1] The asymptotic likelihood result is plotted in Fig. 3 in dashed blue. It can be seen that the peak of the posterior found using the asymptotic likelihood corresponds with the peak of the analytic posterior, although as expected the rest of the distribution quickly deviates from the analytic result. To perform PMC-ABC, $N = 1000$ parameter values, $\{\vartheta_k^0 | k \in [1, N]\}$, are drawn from the uniform prior distribution, $p(\vartheta)$, between $\vartheta = (0, 10]$. The PMC procedure, described above, is then carried out to obtain 1000 samples from the approximate posterior. Using a criterion that there needs to be 2000 draws of ϑ_k^t in iteration t to be convinced that the approximate posterior has converged requires a total of 10232 simulations. The width of the acceptance parameter is $\epsilon^T = 0.086$ at the last iteration, T, meaning that the network summary of each of the accepted network summaries are within a band of $x_{ik}^{sT} = x \pm 0.086$ of the network summary of the real data, x. The histogram of the accepted points are shown in Fig. 3 in purple. The PMC-ABC posterior distribution follows the analytic posterior distribution exactly, showing that the network has successfully learned how to summarize the data.

[1]This approximation is only true for $\mathrm{Abs}[\Delta\vartheta] \ll 1$. The interval chosen here is used only for plotting purposes.

TABLE I. Values of the input parameters for the original simulated true data set where the data is Gaussian noise $\mathbf{d} = \{d_i \curvearrowleft \mathcal{N}(0, 1) | i \in [1, n_\mathbf{d}]\}$.

Data	Value
d_1	−0.919 033 99
d_2	−0.373 225 15
d_3	−0.056 133 42
d_4	1.208 167 46
d_5	0.076 492 69
d_6	−0.471 711 41
d_7	−1.475 657 1
d_8	−0.629 464 63
d_9	−1.303 340 79
d_{10}	−0.414 416 39

It is interesting to see the network outputs as a function of ϑ, without using the PMC procedure. By performing ABC by randomly drawing from the whole prior, and not honing in on the true distribution, we can plot the network output as a function of the variance drawn from the prior, shown in Fig. 4. The green points show the rejected samples and the purple points (under the black dashed line) show the accepted draws. The black dashed line shows the network output of the real data. There is a strong correlation between the network summary of the simulations and the value of ϑ used to create the simulation. Requiring that there are 1000 samples whose summaries are within $x_{ik}^{sT} = x \pm \epsilon^T$, where $\epsilon^T = 0.086$, necessitates more than 600 000 draws from the prior, 50 times more

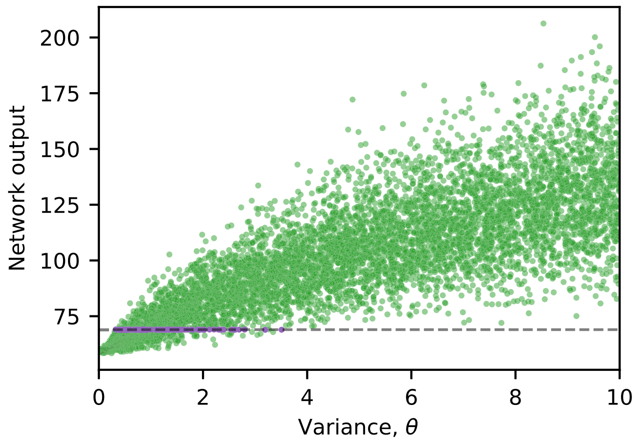

FIG. 4. Network output as a function of the variance used to create the simulations. The green points are the network summaries of a selection of the simulations created from random draws from $\vartheta = (0, 10]$ for the random ABC procedure. The purple points are the accepted network summaries of the 1000 simulations within $x_{ik}^{sT} = x \pm \epsilon^T$ with $\epsilon^T = 0.086$. The black dotted line indicates the network output of the real data. There is a strong correlation between the network output and the value of ϑ which suggests that the network has learned how to summarize the network input with respect to the model parameters.

draws than the PMC needs. It should be noted that the network summary is not equal to the value of ϑ and, in general can vary a lot by changing the network architecture, the initialization of the weights or even just changing the order of the simulations used to train the network. The variation in the network summary is a manifestation of how the Fisher information is invariant under linear scalings of a sufficient statistic, although the scale of the statistic is able to be constrained somewhat by coupling the Fisher information matrix to the covariance of the outputs, as in Eq. (5.6).

When creating simulations during the ABC procedure we can calculate the true sufficient statistic, i.e.

$$x_i^{\mathbf{s}} = \sum_{j=1}^{n_{\mathbf{d}}} (d_{ij}^{\mathbf{s}})^2 \tag{7.8}$$

where i labels the random initialization of the simulation and the j labels the data point in the data set $\mathbf{d}$. Plotting the exact sufficient statistic against the network output allows us to see how well the correct function is learned by maximizing the IMNN, as seen in Fig. 5. The blue points show the values of exact sufficient statistics of the simulations and scaled values of the network outputs of the same simulations. The network output must be scaled due to the allowed linear scaling of the sufficient statistic. We actually found that network output is approximately

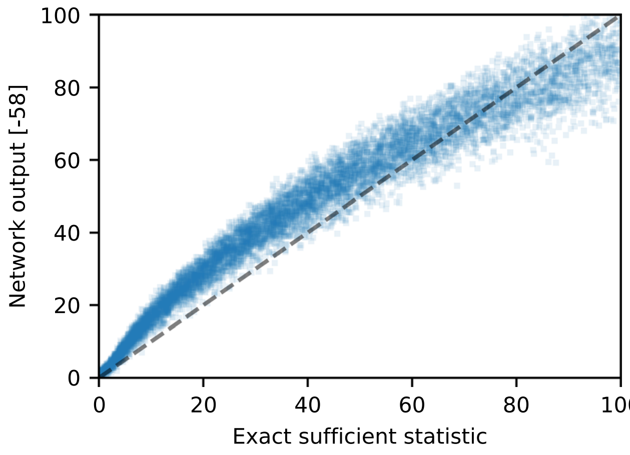

FIG. 5. Rescaled network output for a given exact sufficient statistic. The blue dots show a scaled value of the network output at the exact sufficient statistic of simulations obtained at a range of ϑ during ABC. The black dashed line shows the expected value of the network output if the network had learned the map from data to the sufficient statistic perfectly. Since the scatter of the exact sufficient statistic to the network output closely follows the black dashed line, we know the network has approximately learned the correct map from data to sufficient statistic. There is a slight curve which arises from the fact that any one-to-one function of the sufficient statistic is still a sufficient statistic and so is of no concern. There is also a superficial broadening of the curve which shows that the map is only approximately correct.

$$\text{network output} \approx \sum_{j=1}^{n_{\mathbf{d}}} (d_{ij}^{\mathbf{s}})^2 + 58, \tag{7.9}$$

without a linear scaling of the exact sufficient statistic, but with an offset. The black dashed line shows what would be expected if the exact map was learned by the network. We can see that the network output generally follows the sum of the square of the data closely with hints of a slight bend and superficial broadening at larger exact sufficient statistics. The bending is of no concern since any one-to-one function of the sufficient statistic is still a sufficient statistic, and we can see that the network output is clearly a monotonic function of the real summary. The broadening indicates that only an *approximate* map is learned because the training of the network is incomplete due to lack of diversity within simulations and perhaps a suboptimal choice of network hyperparameters. With greater variety within the simulations or, likewise, a greater number of simulations, the optimal map could be learned even more precisely. Nevertheless, we can see how minor an effect the broadening of the exact sufficient statistic is by looking at the results in Fig. 3. The resulting posterior distribution is equivalent to the analytic posterior, which is the real proof that the network has found the correct summary statistic.

1. Summarizing Gaussian signals with known noise variance

Now consider some noisy data where the real data $\mathbf{d} = \{d_i \curvearrowleft \mathcal{N}(0, \vartheta + \sigma_{\text{noise}}^2) | i \in [1, n_{\mathbf{d}}]\}$ has a signal variance of $\vartheta^{\text{true}} = 1$ and the variance of the noise is taken to be known $\sigma_{\text{noise}}^2 = 1$. Simulations of the noisy data can be created and used to train the network, as before. The addition of the noise makes the likelihood less peaked about the *true* parameter value and so the Fisher information is expected to be less than in original problem. Since the likelihood is known analytically, using Eq. (7.6) it can be seen that $\mathbf{F} = 1.25$. The network manages to achieve $\mathbf{F} \approx 1.25$ by the end of training, suggesting the network is capable of extracting close to the maximum amount of information possible. We have used a slightly less complex network here with [128, 128], but all other parameters the same. Again, many different architectures work equally well, but do not necessarily converge as quickly. We train the network for 2000 epochs before the Fisher information saturates to its maximum value.

In Fig. 6, it can be seen that the PMC-ABC posterior distribution, shown in the purple histogram with shaded 1-σ Poisson regions, when given some simulated test data, $\mathbf{d}$, is very similar to the analytic result shown by the solid orange line. The dashed blue approximate posterior distribution from the asymptotic likelihood again peaks very close to the maximum of the analytic posterior. The posterior distribution becomes maximal at the most likely parameter value given the data, with the variance given by the inverse

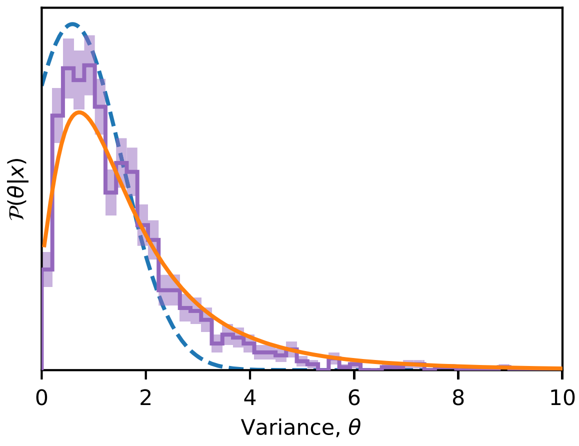

FIG. 6. The posterior distribution of the signal variance of the Gaussian noise when the data is contaminated with known noise of $\sigma^2_{\mathrm{noise}} = 1$. The solid orange line shows the analytic posterior distribution, while the posterior distribution from the asymptotic likelihood is shown in dashed blue and the purple histogram with 1-σ Poisson shaded error bars shows the approximate posterior distribution from PMC-ABC. Each distribution is normalized such that its integral is unity in the interval $\vartheta = [0, 10]$. We can see that the solid orange curve and the purple histogram overlap along the entire range of ϑ suggesting that the network has learned the correct way to summarize the data.

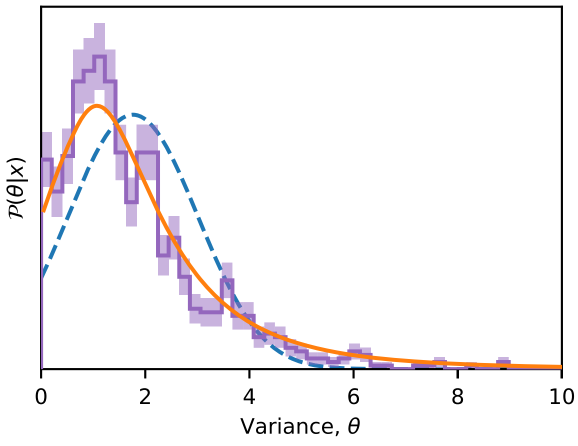

FIG. 7. The posterior distribution of the signal variance when the data is contaminated with unknown noise $\sigma^2_{\mathrm{noise}} = (0, 2]$. The exact posterior is shown by the solid orange curve, the posterior distribution obtained using the asymptotic likelihood is in dashed blue and the PMC-ABC posterior with samples drawn using PMC is indicated by the purple histogram with shaded 1-σ Poisson error bars. Each distribution is normalized such that its integral is unity in the interval $\vartheta = [0, 10]$. Even with unknown noise the network can summarize the data equally as well as a Rao-Blackwell estimate of the analytic case, leading to equivalent posterior distributions. The posterior distribution obtained from the asymptotic likelihood does not agree with the other distributions since the training simulations are not representative of the real data.

Fisher information at the end of training. There are 1000 samples used to create the histogram of the PMC-ABC posterior which required approximately 2×10^5 simulations to be created during the PMC, where all samples are within $x^{sT}_{ik} = x \pm \epsilon^T$, with $\epsilon^T = 0.109$. Since the analytic posterior distribution is so similar to the PMC-ABC posterior we can see that, even though the network is only given noisy simulations, it is capable of finding the true function to summarize the data.

2. *Summarizing Gaussian signals with unknown noise variance*

Now consider the problem where, again, the real data $\mathbf{d} = \{d_i \curvearrowleft \mathcal{N}(0, \vartheta + \sigma^2_{\mathrm{noise}}) | i \in [1, n_{\mathbf{d}}]\}$ has a signal variance of $\vartheta = 1$ and the variance of the noise is also unknown with a uniform prior $\sigma^2_{\mathrm{noise}} \in (0, 2]$. We train using 1000 simulations (+100 for each of the derivatives) at a fiducial $\vartheta^{\mathrm{fid}} = 1$ each with a different $\sigma^2_{\mathrm{noise}}$ randomly drawn from the uniform prior on the noise. The final value of the Fisher information from the network is less than in either of the two previous cases at $\mathbf{F} = 0.9$ using a slightly more complex network than in the previous section with an architecture of [128, 128, 64] but all other parameters the same. If the noise were assumed to be known at $\sigma^2_{\mathrm{noise}} = 2$ then the maximum Fisher available, as calculated from Eq. (7.6) would be $\mathbf{F} = 5/9$. The posterior distributions for ϑ are shown in Fig. 7. Since the noise is unknown, a Rao-Blackwell estimate of the analytic distribution is made. Here, the posterior distribution is calculated for a range of given noise values from $\sigma^2_{\mathrm{noise}} = (0, 2]$ and their results summed at each value of ϑ, plotted with a solid orange line. The PMC-ABC posterior is given by the purple histogram consisting of 1000 samples, which required approximately 10^5 simulations using the PMC. Again, as before, the constraints on ϑ are incredibly similar to the analytic result, confirming that the network can approximate the exact summary very well. The Rao-Blackwell estimation procedure is also carried out to obtain the posterior calculated from the asymptotic likelihood, in dashed blue, although the result does not agree with the exact or PMC-ABC posteriors. The lack of agreement arises because the simulated test data is not well represented in the training simulations. Even though there is an under representation in the data, the network has learned the correct way to summarize data independent of the input, i.e. the network calculates the sum of the square of the input.

B. Summarizing Gaussian signals with wrong fiducial variance

Since the network trained in the known noise problem, in Sec. VII A 1, is akin to a network trained at a fiducial parameter $\vartheta^{\mathrm{fid}} = 2$, we can use it to test how well the network can predict the variance when the fiducial value does not coincide with the true parameter. It would be

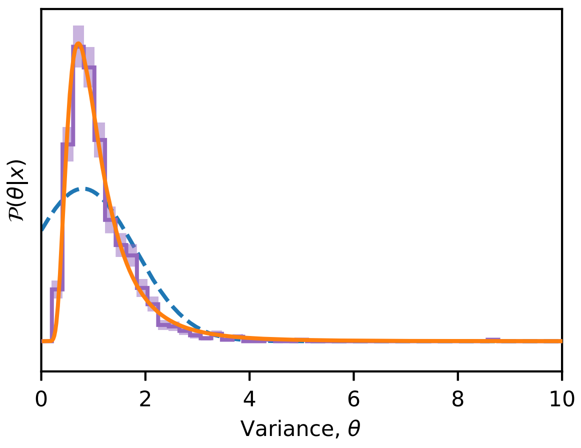

FIG. 8. The posterior distribution of parameter ϑ where the real data is that of Table I, but the network has been trained with a fiducial $\vartheta^{\rm fid} \neq \vartheta^{\rm true}$. The solid orange line shows the analytic posterior distribution and the purple histogram, with shaded 1-σ Poisson widths, shows the approximate posterior distribution from PMC-ABC. The dashed blue curve shows the posterior distribution from the asymptotic likelihood. Each distribution is normalized such that its integral is unity in the interval $\vartheta = [0, 10]$. The analytic posterior distribution and the PMC-ABC posterior are again identical, which shows how the network is able to find the correct function to map data to summaries, even when the fiducial training parameter value is incorrect.

expected that data with $\vartheta = 1$ would be under-represented in a training data set where the fiducial value is $\vartheta^{\rm fid} = 2$. Naïvely, one would assume that the network would not perform as well as a network trained using simulations created at $\vartheta^{\rm fid} = 1$, especially since the Fisher information available from this network is $\mathbf{F} = 1.25$ and not $\mathbf{F} = 5$ as in Sec. VII A. However, Fig. 8 shows that the parameter constraints given the same real data as in Table I are equally as strong as when using the trained network from Sec. VII A. It is promising that the training of the network seems fairly insensitive to the choice of fiducial parameter. The posterior distribution from the asymptotic likelihood, in dashed blue, is much wider than the same curve in Fig. 3 since the variance of the distribution is given by the Cramér-Rao bound, i.e. $\mathbf{F}^{-1} = 0.8$, rather than $\mathbf{F}^{-1} = 0.2$ when using the network from Sec. VII A 1. The fact that the purple histogram matches the analytic solid orange distribution so well indicates that the network has learned the correct way to summarize data, rather than learning an algorithm for mapping simulations to an output which specifically depends on the fiducial parameter value. For example, in the problem considered here, we know that the correct summary of the data is the sum of the square of the data (or at least a linear scaling of the sum of the square of the data). The network is trained in such a way that the abstract function of weights, biases and inputs that the network represents closely approximates the sum of the square of the input. Once abstract function is learned, it does not matter what parameter value is used to create the simulations, even if that parameter is far from the fiducial value, because the network will still output the sum of the square of the input. It is extremely encouraging to see that the network can extrapolate beyond its training data by depending on the robustness of the learned patterns.

C. Summarizing quasar spectra

Beyond the elementary test case on variance estimation, we can consider models that are of more astronomical interest. Here we attempt to generate constraints on the amplitude of scalar perturbations, $A_{\rm s}$, using a simplistic 1D model of the Lyman-α forest from a single quasar. To generate simulations we begin by using the halo mass function calculator hmf module [59] in python to generate the 3D power spectrum $P^{\rm 3D}(k)$, evolved using the method of Eisenstein and Hu [60], at a redshift of $z = 2.25$ with fixed cosmological parameters (at $z = 0$). The cosmological parameters come from the Planck 2015 temperature and low-ℓ polarization results [61], $H_0 = 67.7\ {\rm km\,Mpc^{-1}\,s^{-1}}$, $\Omega_{\rm m} = 0.307$, $\Omega_{\rm b} = 0.0486$, $n_{\rm s} = 0.9667$, $\sigma_8 = 0.8159$, $T_{\rm CMB} = 2.725$ K, $N_{\rm eff} = 3.05$, and $\sum m_\nu = 0.06$ eV, calculated using astropy [62]. The power spectrum is calculated between $\ln k_{\rm min}/(1h\,{\rm Mpc}^{-1}) = -18.42$ and $\ln k_{\rm max}/(1h\ {\rm Mpc}^{-1}) = 9.90$ in steps of $\Delta \ln k/(1h\ {\rm Mpc}^{-1}) = 0.005$. The correlation function can be found using

$$\xi(r) = \int_0^\infty \frac{dk}{2\pi^2} \exp{[-R_w^2 k^2]} k^2 P^{\rm 3D}(k) {\rm sinc}(kr) \qquad (7.10)$$

where the exponential term is a smoothing function where we use $R_w = 5h^{-1}$ Mpc. We calculate the value of $\xi(r)$ between $-200 < r < 200h^{-1}$ Mpc in $N = 8192$ bins. To simulate the density fluctuations along the line of sight, we calculate the 1D power spectrum using

$$P^{\rm 1D}(k) = \int_{-\infty}^\infty dr \exp[ikr] \xi(r). \qquad (7.11)$$

The Lyman-α peak in the rest frame of an emitter is $\lambda_{\rm RF}^\alpha = 121.567$ nm [63] and we use the fact that BOSS can measure absorbers in the redshift range $1.96 < z < 3.44$ [64]. Using

$$z = \frac{\lambda}{\lambda_{\rm RF}} - 1 \qquad (7.12)$$

the minimum observed wavelength of the Lyman-α peak is $\lambda_{\rm min} = 359.838$ nm (at $z = 1.96$) and the maximum wavelength is $\lambda_{\rm max} = 539.757$ nm (at $z = 3.44$) [64]. The length L of the survey in comoving space is calculated between these redshifts, yielding $L = 1122.9h^{-1}$ Mpc. The frequency spacing is given by the inverse of the survey length, so we consider a range of $k = (0, 14.6]h\ {\rm Mpc}^{-1}$ with $N = 8192$ bins. We modify the 1D power spectrum such that it more closely follows the gas power spectrum as seen from Lyman-α absorptions [65],

$$P_{\rm g}^{\rm 1D}(k) = \beta D(k,\mu)P^{\rm 1D}(k) \tag{7.13}$$

where β is a free parameter, set for a given realization of noise which ensures that $\langle F\rangle = 0.8$ [66]. $D(k,\mu)$ is a term which modifies the small-scale power spectrum [65] and is of the form

$$D(k,\mu) = \exp\left[\left(\frac{k}{k_{\rm NL}}\right)^{\alpha_{\rm NL}} - \left(\frac{k}{k_{\rm P}}\right)^{\alpha_{\rm P}} - \left(\frac{k_\parallel}{k_{\rm V}}\right)^{\alpha_{\rm V}}\right], \tag{7.14}$$

where

$$k_{\rm V} = k_{{\rm V}_0}\left(1+\frac{k}{k'_{\rm V}}\right)^{\alpha'_{\rm V}} \tag{7.15}$$

and $k_{\rm NL} = 6.40h$ Mpc^{-1}, $\alpha_{\rm NL} = 0.569$, $k_{\rm P} = 15.3h$ Mpc^{-1}, $\alpha_{\rm P} = 2.01$, $k_{{\rm V}_0} = 1.220$, $k'_{\rm V} = 0.923h$ Mpc^{-1}, $\alpha'_{\rm V} = 0.451$, $\alpha_{\rm V} = 1.50$ [67] and we choose to use $\mu = k_\parallel/k = 1$ since we only consider independent quasar lines, i.e. the flux is completely decorrelated from one line to the next. The above numbers are computed for the log-flux explicitly described in [65]. For the purpose of demonstration we keep the same k dependence here. With the gas power spectrum in Eq. (7.13), normalized by the length of the survey, we can generate 1D random Gaussian fields, $\delta_{\rm g}$. The Gaussian fields are generated by multiplying unit variance, zero-mean Gaussian noise with $(P_{\rm g}^{\rm 1D}(k)/2)^{1/2}$ and Fourier transforming into real space, including the normalization of $N/(2L)$ due to the discrete nature and finite period of the discrete Fourier transform. The flux from quasars is absorbed by neutral hydrogen in overdensities in the density field, and can be calculated from the fluctuating Gunn-Peterson approximation [68] as

$$F = \exp[-\tau] \tag{7.16}$$

where we consider the form of the optical depth to be

$$\tau = 1.54\left(\frac{T_0}{10^4\ {\rm K}}\right)^{-0.7}\frac{10^{-12}\ {\rm s}^{-1}}{\Gamma_{\rm UV}}\left(\frac{1+z}{1+3}\right)^6 \tag{7.17}$$

$$\times\frac{0.7}{h}\left(\frac{\Omega_{\rm b}h^2}{0.02156}\right)^2\frac{4.0927}{H(z)/H_0}\rho^{2-0.7\gamma} \tag{7.18}$$

where $T_0 = 18400$ K is the normalization to the power-law temperature-density relation $T = T_0(1+\delta_g)^{\gamma-1}$ with $\gamma = 0.29$ [both here and in Eq. (7.18)] and $\Gamma_{\rm UV} = 4\times10^{-12}$ s^{-1} is the photoionization rate due to the ambient UV background [68]. The gas density field is normalized such that its mean is unity,

$$\rho = \frac{\exp[\delta_{\rm g}]}{\langle\exp[\delta_{\rm g}]\rangle}. \tag{7.19}$$

The continuum flux can be calculated between the Lyman-α and Lyman-β peaks at $\lambda_{\rm RF}^\alpha = 121.567$ nm and $\lambda_{\rm RF}^\beta = 102.572$ nm using the PCA formulation of [69]. The continuum flux in the rest frame of the emitter is calculated using

$$r(\lambda) = \mu(\lambda) + \sum_i c_i(\lambda)\xi_i(\lambda) \tag{7.20}$$

where $\mu(\lambda)$ is the mean flux over many quasars, $\xi_i(\lambda)$ are the $i^{\rm th}$ principal components and $c_i(\lambda)$ are the amplitudes of the principal components which we consider to be $c_i(\lambda) = 1$ for simplicity. The continuum can be transformed into the observer's wavelength space by assuming a redshift for the quasar and inverting Eq. (7.12). We choose the redshift of the simulated (and real) quasar to be $z = 2.91$. The flux, which is currently in real space, is transformed into wavelength space by interpolating the comoving distance, r, along given redshift values, z, using the hmf comoving_distance(z) function and then using Eq. (7.12). The continuum modulated flux from the quasar is simply

$$f(\lambda) = F(\lambda)C(\lambda) \tag{7.21}$$

where $C(\lambda)$ is $r(\lambda)$ from Eq. (7.20) in the rest frame of the observer [63]. Figure 9 shows the generated flux from a single mock quasar at $z = 2.91$ in blue. The orange (lighter) line shows the continuum flux between the Lyman-α and Lyman-β peaks and the dashed green line shows the mean of the transmitted flux. We only consider the flux between $406.6\ {\rm nm} < \lambda < 469.2$ nm which is $104\ {\rm nm} < \lambda < 120$ nm in the rest frame of the quasar [64]. We bin the wavelengths using the resolution from the BOSS coadded spectra of $\Delta\log_{10}\lambda/1\ {\rm nm} = 10^{-4}$ [63] which gives a flux

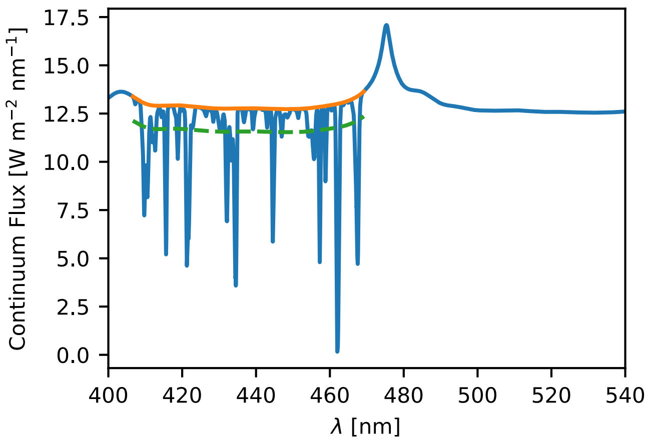

FIG. 9. Simulated spectrum of a quasar at $z = 2.91$ in blue, with the value of the continuum in orange (light) and the mean flux in the Lyman-α forest in dashed green.

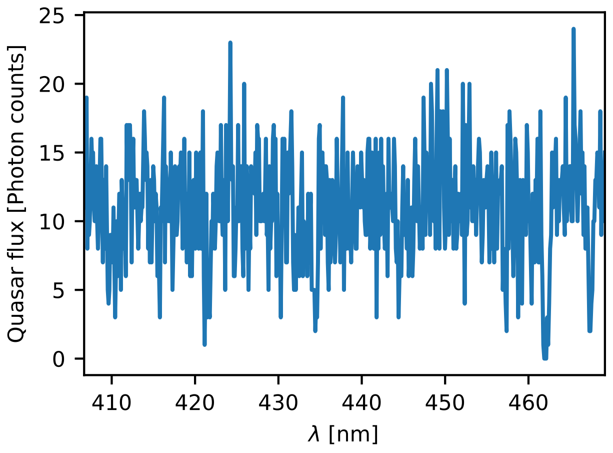

FIG. 10. Simulated observation of a quasar spectrum from a quasar at $z = 2.91$ between the Lyman-α and Lyman-β peaks in the rest frame of the observer. We use this data as our simulated test data for the PMC-ABC.

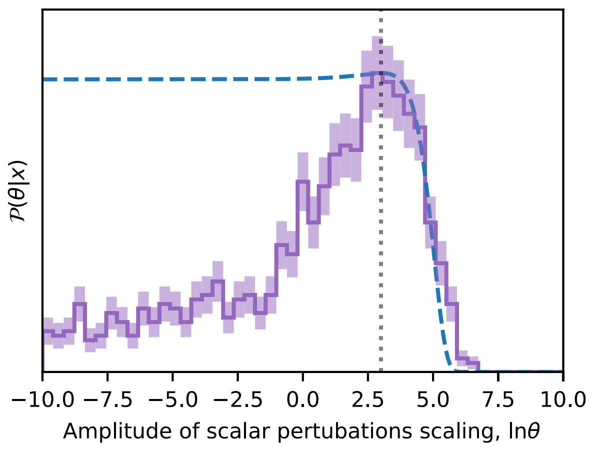

FIG. 11. Posterior distribution for the scaling of the amplitude of scalar perturbations, $\ln\vartheta$. The dashed blue curve shows the Gaussian approximation to the true constraints as estimated with the training simulations and the purple histogram, with shaded 1-σ Poisson error bars, is calculated from samples from PMC-ABC. The peak of both posterior distributions occur at the vertical black dotted line which shows the true value of the parameter. Although the constraints span several orders of magnitude, we expect these kinds of constraints from a single observation of a quasar absorption spectrum. Most importantly, we have shown that we can summarize extremely noisy data by solely maximizing the Fisher information.

in $\mathrm{Wm}^{-2}\ \mathrm{nm}^{-1}$, but needs to be measured in photon counts. Using the method[2] in [70] we see that for a quasar such as the one we are generating the spectra for, there is an almost one-to-one correspondence between flux and photon count (albeit the photon count is integer) [70]. Therefore, we make the assumption that making the flux into integer values and then applying Poisson noise satisfactorily represents real quasar spectra. Our binned, noisy spectra have 581 data points, each of which can be used as an input to an IMNN.

An example of the simulated test data input to the network is shown in Fig. 10.

For any set of fixed cosmological parameters the value of amplitude of scalar perturbations, A_s, is a scaling of $P_g^{1D}(k)$. To get constraints on A_s we can train a network at a fiducial A_s and then use the PMC to find the posterior distribution of A_s compared to some simulated test data. In fact, for simplicity, we can consider the parameter ϑ to be some multiplicative scaling of the amplitude, $A_s = \vartheta A_{\mathrm{cosmo}}$ with A_{cosmo} the amplitude of the power spectrum found in Eq. (7.13). We use $\vartheta^{\mathrm{fid}} = \exp[0]$ as the fiducial parameter, i.e. $A_s^{\mathrm{fid}} = A_{\mathrm{cosmo}}$.

A relatively simple network, such as [256, 256], is able to obtain a Fisher information of $\mathbf{F} = 0.015$, which is the maximum Fisher information that could be found over a large range of different network architectures and hyperparameters. However, the network which was most resilient to incorrect fiducial values was more complex than those networks previously considered. The network with the largest Fisher information by the final epoch of training, which could handle incorrect fiducial parameters was a network four hidden layers shaped like [1024, 512, 256, 128], using 1000 simulations (with 100 simulations each for the upper and lower components of the derivative) which were split into two batches, an initial bias of $\boldsymbol{b} = 0.1$, where the activation function is leaky ReLu with $\alpha = 0.1$, a dropout of 20% and a learning rate of $\eta = 1 \times 10^2$ when training for 10 000 epochs.

As before, once the network was trained, PMC-ABC could be performed. We used a uniform prior in logarithmic space of $\vartheta = \exp[-10, 10]$. The simulated test data was created away from the fiducial parameter value of $\vartheta^{\mathrm{fid}} = \exp[0]$ at $\vartheta^{\mathrm{real}} = \exp[3]$, i.e. $A_s = \exp[3]A_{\mathrm{cosmo}}$, and is shown in Fig. 10. The posterior distribution for the value of ϑ can be found in Fig. 11. Here, we required 1000 samples in the posterior requiring at least 2500 draws in the final iteration of the PMC to be convinced that the posterior had converged. The histogram peak, and the tentative peak of the leading order expansion of the likelihood, are at their maximum at $A_s \approx \exp[3]A_{\mathrm{cosmo}}$, i.e. $\ln\vartheta \approx 3$, which confirms that the correct test parameter can be recovered, shown as the vertical black dashed line in Fig. 11. There is a large, degenerate tail in the PMC-ABC posterior which arises due to the amplitude of the random Gaussian noise, used to create the quasar spectrum, being so small that the features in the generated flux become negligible. Since the network output of the random fluctuations are still reasonably close to the network output from the real data, they cannot be constrained. The lack of constraining power at

[2]In particular we use the method described in http://www.sdss.org/dr12/algorithms/spectrophotometry/ in the section called "DR9 Flux to Photons." We use quasar 024918.47 + 025035.6 as a guideline.

low ϑ is even clearer in the posterior from the leading order expansion. The constraints on A_s span approximately 5 orders of magnitude or more using the PMC-ABC posterior, which seems poor, but is due to using only one quasar spectrum to constrain cosmology with. Joint inference using several quasars would provide a much stronger constraint, as is done when using cosmological surveys. Although the constraints are not particularly strong, we have shown that we can learn to extract information from highly noisy data, and summarize it in such a way that we can perform PMC-ABC to get a posterior distribution for parameters of interest.

D. Gravitational waveform frequency

Reference [34] showed that the MOPED algorithm, described in Sec. II, was unable to summarize the central oscillation frequency of a gravitational waveform from LISA without introducing spurious features [34]. By using nonlinear summaries of the data, the problem in [34] can be avoided.

We start by considering a sine-Gaussian gravitational wave signal as could be seen in LISA, with a short burst duration and frequency space waveform [71] of

$$\bar{h}(f) = \frac{AQ}{f}\exp\left[-\frac{Q^2}{2}\left(\frac{f-f_c}{f_c}\right)^2\right]\exp\left[2\pi i f_c t_c\right], \qquad (7.22)$$

where A is some amplitude, Q is the width of the gravitational wave burst, t_c is the time of the burst and f_c is the central oscillation frequency. We fix $A = 3.5$, $Q = 5$ and $t_c = 1 \times 10^5 s$ and require that the signal-to-noise of the burst is $S/N = 34$ [34]. We are interested in summarizing and constraining the parameter f_c. To generate a simulation of the gravitational wave signal, we use the one-sided noise power spectral density of the LISA detector [71], which is

$$S_h(f) = 16\sin^2[2\pi f t_L](2S_{pn}(1+\cos[2\pi f t_L] + \cos^2[2\pi f t_L]) + (\cos[2\pi f t_L]/2+1)S_{sn}f^2), \qquad (7.23)$$

$$S_{pn}(f) = \left(1+\left(\frac{10^{-4}\ \mathrm{Hz}}{f}\right)^2\right)\frac{S_{acc}}{f^2}, \qquad (7.24)$$

where $S_{sn} = 1.8 \times 10^{-37}\ \mathrm{Hz}^{-1}$ is the shot noise, $S_{acc} = 2.5 \times 10^{-48}\ \mathrm{Hz}^{-1}$ is the proof acceleration mass and $t_L = 16.678$ s is the light travel time along one arm of the LISA constellation. To generate the real space gravitational wave burst, we calculate the frequency space waveform $\bar{h}(f)$ and detector noise $\bar{n}(f)$ and then Fourier transform them into real space

$$\bar{h}(f) = \int_{-\infty}^{\infty} dt h(t)\exp[2\pi i f t], \qquad (7.25)$$

$$\bar{n}(f) = \int_{-\infty}^{\infty} dt n(t)\exp[2\pi i f t]. \qquad (7.26)$$

We perform the Fourier transform at 2048 time steps from $t = 9.9 \times 10^4$ s, sampled at 1 s intervals. The output of the LISA detector is then given by

$$\mathbf{d} = \mathbf{h}(\boldsymbol{\vartheta}^{\mathrm{true}}) + \mathbf{n} \qquad (7.27)$$

where $\mathbf{h}(\boldsymbol{\vartheta}^{\mathrm{true}})$ is the values of the gravitational waveform at the true parameter values, $\boldsymbol{\vartheta}^{\mathrm{true}} = \{A^{\mathrm{true}}, Q^{\mathrm{true}}, t_c^{\mathrm{true}}, f_c^{\mathrm{true}}\}$ at the sampled time and $\mathbf{n}$ is a random realization of the noise. When assuming a noise covariance which is independent of the signal, $\sigma_n^2 = \mathbb{I}$, the logarithm of the likelihood is particularly simple [71] and is given by

$$\ln\mathcal{L} = C - \frac{\|\mathbf{d}-\mathbf{h}(\boldsymbol{\vartheta})\|^2}{2}, \qquad (7.28)$$

where $\mathbf{h}(\boldsymbol{\vartheta})$ is the real space gravitational wave at, not-necessarily-true, parameters $\boldsymbol{\vartheta}$ and C is a constant which we set to zero.

We are interested in summarizing the data to constrain the central oscillation frequency, f_c, of the gravitational wave. To do so, we use a network which takes in the 2048 inputs from the data with the architecture [10, 10, 10, 10, 10]. The network has a 10% dropout and leaky ReLU activation with $\alpha = 0.01$. The learning rate is fixed at $\eta = 10^{-5}$ and the biases are initialized slightly positively at $\boldsymbol{b}^l = 0.1$. We train for 1200 epochs using 1000 fiducial simulations and 100 simulations each for the positive and negative parts of the numerical derivative, all of which is split into two combinations. Once trained, we can use the network to summarize the data. We also use Eq. (7.28) to calculate the logarithm of the likelihood from the summary by passing the simulated test data, $\mathbf{d}$, with a given realization of the noise and generated at $f_c^{\mathrm{true}} = 0.1$ Hz, through the network $\mathscr{f}:\mathbf{d} \to x$ and comparing it to the waveform at a given f_c, $\mathscr{f}:\mathbf{h}(f_c) \to x^{\mathbf{h}}(f_c)$. However, since the noise is included in the realizations which is passed through the network, the noise variance needs to be transformed as well. Assuming the variance is small, so that the likelihood remains Gaussian near the peak, the error propagation gives the new variance as

$$\sigma_n'^2 = \left|\frac{\partial x^{\mathbf{h}}(f_c)}{\partial f_c}\right|^2 \sigma_n^2 \qquad (7.29)$$

where the gradient should be evaluated at or near the true mean. The modified approximate likelihood, assuming Gaussian noise, for the IMNN summary evaluated at different parameters is therefore given by

$$\ln\mathcal{L} = C - \frac{\|x - x^{\mathbf{h}}(f_c)\|^2}{2\sigma_n'^2}. \qquad (7.30)$$

We calculate Eqs. (7.28) and (7.30) using simulated test data, $\mathbf{d}$, generated at $f_c^{\mathrm{true}} = 0.1$ Hz between $1 \times 10^{-2} < f_c < 0.5$ Hz. The logarithm of the likelihood of f_c

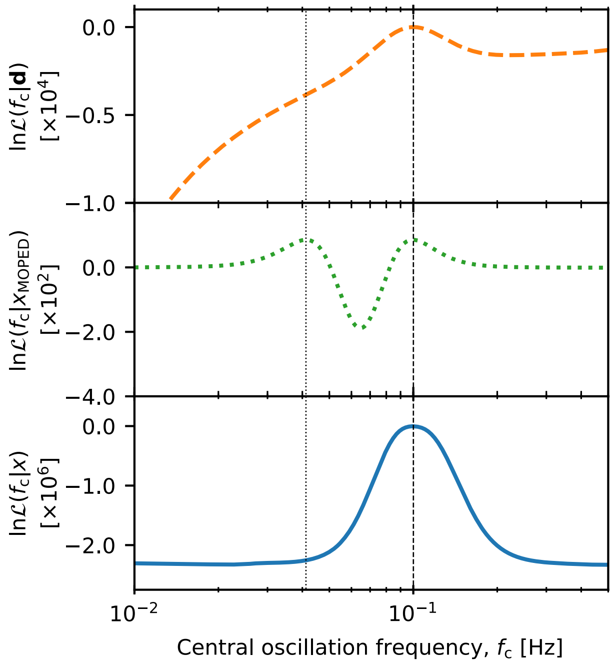

FIG. 12. Logarithm of the likelihood for the central oscillation frequency, $f_{\rm c}$. The dashed orange line in the upper panel shows the likelihood using all the data, while the green dotted line in the middle panel and the blue solid line in the bottom panel show the approximate Gaussian likelihoods when using compression, MOPED and IMNN respectively. All three likelihoods have peaks at the correct $f_{\rm c}^{\rm true} = 0.1$ Hz, but an aliasing peak arises in the MOPED likelihood due to a nonmonotonic mapping from $\mathbf{h}(f_{\rm c}) \to x^{\mathbf{h}}_{\rm MOPED}(f_{\rm c})$. The compression using the IMNN on the other hand does not have any aliasing peaks since the network has learned the nonlinear map from data to frequency.

calculated using all the data, $\ln\mathcal{L}(f_{\rm c}|\mathbf{d})$ is shown in the upper subplot of Fig. 12 as a dashed orange line. This is compared to $\ln\mathcal{L}(f_{\rm c}|x_{\rm MOPED})$ using the MOPED summary as the dotted green line in the middle subplot and $\ln\mathcal{P}(f_{\rm c}|x)$ using the network summary as the solid blue line in the bottom subplot. The MOPED summary assumes a noise covariance which is independent of the signal such that the compression parameter is simply

$$\mathbf{r}_{f_{\rm c}} \propto \boldsymbol{\mu}_{,f_{\rm c}}. \tag{7.31}$$

For the network summary, the noise is automatically included through the random initialization of the simulations used to train the network. It can be seen that each of the likelihoods in Fig. 12 agree with $f_{\rm c}^{\rm true} = 0.1$ Hz, shown with the dashed black line, but a false aliasing peak appears, shown with the dotted black line, when using the MOPED summary. This false maximum in the likelihood arises from unsuccessfully undoing the Fourier transform which leaves the mapping from $\mathbf{h}(f_{\rm c}) \to x^{\mathbf{h}}_{\rm MOPED}(f_{\rm c})$ not being one-to-one. On the other hand, the IMNN compression does not suffer this problem. There is a clear unique summary which, when used to calculate the approximate likelihood assuming Gaussian noise and evaluated at different $f_{\rm c}$, results in a single peak at the $f_{\rm c}^{\rm true}$. Full inference on $f_{\rm c}$ is then possible using PMC-ABC.

This test shows that, through the use of the nonlinear function provided by the IMNN, we are able to surpass the capability of linear compression. Not only can the summary from the network be at least as informative as the MOPED summary, it is also more robust since it is able to avoid misleading parameter inference due to nontrivial mappings.

VIII. CONCLUSIONS

We have shown how IMNNs can perform automatic physical inference. Automatic physical inference begins by training a neural network to find the optimal nonlinear summaries of data supplied only with simulations and no other knowledge about how to best compress data. Once the network is trained, its output is used to perform PMC-ABC and find the approximate posterior distribution of any parameter that the network is sensitive to. We have also shown that the network is insensitive to poor choice in fiducial parameter value when generating simulations.

We consider the technique presented in this paper as an extension or replacement to other massive optimal data compression procedures. The MOPED algorithm is able to optimally compress data using linear combinations under the assumption that the likelihood is known and is, to first order, Gaussian. Further, the method in [19] generalizes MOPED to any given likelihood function, where the compressed statistics no longer need to be linear. In [33], the likelihood does not need to be known at all, first summarizing simulations of real data heuristically and then compressing these summaries using an appropriate likelihood in the same way as [19]. Although a powerful technique, the first step in [33] can potentially be lossy and the likelihood in the second step should be well known to achieve optimal compression of the first-step summaries. The information maximizing neural network can replace both steps in [33] by taking the raw data and providing nonlinear, likelihood-free summaries directly from the simulations. Likewise, and perhaps more conveniently, the network introduced here is ideally placed to squeeze additional information out of the data after all of the more obvious summaries, such as the power spectrum, have been exhausted.

In this paper, we have focussed on a few test models used to illustrate the method and its abilities. The first set of tests uses the network to find a summary of Gaussian signal, without noise, with known noise variance and with unknown noise variance. This is a useful example since it can be solved analytically and linear compression, such as MOPED would fail to provide useful summaries of the data. We showed that PMC-ABC is able to recover the analytic posterior distribution for the variance of the Gaussian noise nearly exactly, which means that the network has correctly learned the sufficient statistic for this problem. It is useful to consider variance inference as there are many examples in astronomy and cosmology where the variance is informative about the underlying parameters. Although the details of the input data and simulations will be more complex, variance estimation appears in cases such as estimating the value of the optical depth to reionization, τ, and recovering B-mode polarization from probes of the large-angle cosmic microwave background polarization anisotropies.

Following the success of the first set of tests, the next two examples show further tests on astronomically motivated problems. The first shows how extremely noisy raw data can be directly input to the network to constrain cosmological parameters and the second shows how using nonlinear summaries are suited to situations where linear summaries can be misleading.

Information maximizing neural networks are designed to deal with raw data. We can see IMNNs being useful, or even essential, when trying to calculate posterior distributions of model parameters where the likelihood, describing the distribution of some large number of data points, is unknown. For example, the raw data from large scale structure surveys is infeasibly large. Even the number of summary statistics is $\sim 10^4$ and a likelihood cannot be written to describe the physics, the selection bias and the instrument—but the data *can* in principle be simulated from initial conditions. The IMNNs presented in this paper to illustrate and explore the concept used a fully connected architecture. When considering very large data sets we will need to consider network architectures that are adapted to the problem at hand and computationally efficient. For example, assuming the isotropy of the universe transverse to the line of sight, while looking radially in redshift space suggests that stacks of convolutional neural networks could be used to deal with raw LSS data. As long as patches of the large scale structure (and the instrument) can be simulated to train the convolutional filter, IMNNs should make it possible to extract cosmologically interesting information directly from the raw data—automatically.

The code used in this paper is available [72].

ACKNOWLEDGMENTS

We would like to thank the referee for their excellent comments and useful input. This work was supported by the ANR BIG4 Grant No. ANR-16-CE23-0002 of the French Agence Nationale de la Recherche as well as the Simons Foundation and we acknowledge that the work has been done within the Labex 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.

[1] J. R. Bond, A. H. Jaffe, and L. Knox, Estimating the power spectrum of the cosmic microwave background, Phys. Rev. D **57**, 2117 (1998).

[2] A. Heavens, R. Jimenez, and O. Lahav, Massive lossless data compression and multiple parameter estimation from galaxy spectra, Mon. Not. R. Astron. Soc. **317**, 965 (2000).

[3] M. Tegmark, A. Taylor, and A. Heavens, Karhunen-Loeve eigenvalue problems in cosmology: How should we tackle large data sets?, Astrophys. J. **480**, 22 (1997).

[4] A. J. Connolly, A. S. Szalay, M. A. Bershady, A. L. Kinney, and D. Calzetti, Spectral classification of galaxies: An Orthogonal approach, Astron. J. **110**, 1071 (1995).

[5] P. J. Francis, P. C. Hewett, C. B. Foltz, and F. H. Chaffee, An objective classification scheme for QSO spectra, Astrophys. J. **398**, 476 (1992).

[6] O. Lahav, *Data Compression, Classification and Parameter Estimation. Methods: Examples from Astronomy* (Springer, Berlin, 2009), pp. 73–76.

[7] D. S. Madgwick *et al.*, The 2dF Galaxy Redshift Survey: galaxy luminosity functions per spectral type, Mon. Not. R. Astron. Soc. **333**, 133 (2002).

[8] F. Murtagh and A. Heck, *Multivariate Data Analysis*, Astrophysics and Space Science Library, Vol. 31, edited by (Springer Netherlands, Dordrecht, 1987), pp. 13–53.

[9] L. Belmon, H. Benoit-Cattin, A. Baskurt, and J.-L. Bougeret, Lossy compression of scientific spacecraft data using wavelets. application to the cassini spacecraft data compression, Astron. Astrophys. **386**, 1143 (2002).

[10] J. Betancort-Rijo, *Structures in Random Fields* (Springer, New York, 2012), pp. 397–399.

[11] I. Segal, *Modern Statistical Methods for Cosmological Testing* (Springer, New York, 2012), pp. 67–81.

[12] A. Heavens, E. Sellentin, D. de Mijolla, and A. Vianello, Massive data compression for parameter-dependent covariance matrices (unpublished).

[13] A. Heavens, B. Panter, R. Jimenez, and J. Dunlop, The star-formation history of the universe from the stellar populations of nearby galaxies, Nature (London) **428**, 625 (2004).

[14] B. Panter, R. Jimenez, A. F. Heavens, and S. Charlot, The star formation histories of galaxies in the Sloan Digital Sky Survey, Mon. Not. R. Astron. Soc. **378**, 1550 (2007).

[15] C. Reichardt, R. Jimenez, and A. Heavens, Recovering physical parameters from galaxy spectra using MOPED, Mon. Not. R. Astron. Soc. **327**, 849 (2001).

[16] S. Gupta and A. F. Heavens, Fast parameter estimation from the cosmic microwave background power spectrum, Mon. Not. R. Astron. Soc. **334**, 167 (2002).

[17] A. Zablocki and S. Dodelson, Extreme data compression for the cmb, Phys. Rev. D **93**, 083525 (2016).

[18] P. Protopapas, R. Jimenez, and C. Alcock, Fast identification of transits from light-curves, Mon. Not. R. Astron. Soc. **362**, 460 (2005).

[19] J. Alsing and B. Wandelt, Generalized massive optimal data compression (unpublished).

[20] J. Pritchard, M. Seielstad, A. Perez-Lezaun, and M. W. Feldman, Population growth of human y chromosomes: A study of y chromosome microsatellites, Mol. Biol. Evol. **16**, 1791 (1999).

[21] S. Tavaré, D. J. Balding, R. C. Griffiths, and P. Donnelly, Inferring coalescence times from dna sequence data, Genetics **145**, 505 (1997).

[22] C. Schafer and P. Freeman, Likelihood-free inference in cosmology: Potential for the estimation of luminosity functions, in *Statistical Challenges in Modern Astronomy V*, edited by E. Feigelson and J. Babu (Springer-Verlag, New York, 2012), Chap. 1, pp. 3–19.

[23] E. Cameron and A. N. Pettitt, Approximate Bayesian Computation for astronomical model analysis: a case study in galaxy demographics and morphological transformation at high redshift, Mon. Not. R. Astron. Soc. **425**, 44 (2012).

[24] A. Weyant, C. Schafer, and W. M. Wood-Vasey, Likelihood-free cosmological Inference with Type Ia Supernovae: Approximate Bayesian computation for a complete treatment of uncertainty, Astrophys. J. **764**, 116 (2013).

[25] A. C. Robin, C. Reylé, J. Fliri, M. Czekaj, C. P. Robert, and A. M. M. Martins, Constraining the thick disc formation scenario of the Milky Way, Astron. Astrophys. **569**, A13 (2014).

[26] C. Hahn, M. Vakili, K. Walsh, A. P. Hearin, D. W. Hogg, and D. Campbell, Approximate Bayesian computation in large-scale structure: Constraining the galaxy-halo connection, Mon. Not. R. Astron. Soc. **469**, 2791 (2017).

[27] T. Kacprzak, J. Herbel, A. Amara, and A. Réfrégier, Accelerating approximate Bayesian computation with quantile regression: Application to cosmological redshift distributions (unpublished).

[28] S. Carassou, V. de Lapparent, E. Bertin, and D. Le Borgne, Inferring the photometric and size evolution of galaxies from image simulations. I. Method, Astron. Astrophys. **605**, A9 (2017).

[29] F. B. Davies, J. F. Hennawi, A.-C. Eilers, and Z. Lukić, A new method to measure the post-reionization ionizing background from the joint distribution of Lyman-α and Lyman-β forest transmission (unpublished).

[30] J. Akeret, A. Refregier, A. Amara, S. Seehars, and C. Hasner, Approximate bayesian computation for forward modeling in cosmology, J. Cosmol. Astropart. Phys. 08 (2015) 043.

[31] E. E. O. Ishida, S. D. P. Vitenti, M. Penna-Lima, J. Cisewski, R. S. de Souza, A. M. M. Trindade, E. Cameron, and V. C. Busti, COSMOABC: Likelihood-free inference via population Monte Carlo approximate Bayesian computation, Astron. Comput. **13**, 1 (2015).

[32] E. Jennings, R. Wolf, and M. Sako, A new approach for obtaining cosmological constraints from type Ia supernovae using approximate Bayesian computation (unpublished).

[33] J. Alsing, B. Wandelt, and S. Feeney, Massive optimal data compression and density estimation for scalable, likelihood-free inference in cosmology (unpublished).

[34] P. Graff, M. Hobson, and A. Lasenby, An investigation into the Multiple Optimised Parameter Estimation and Data compression algorithm, Mon. Not. R. Astron. Soc. **413**, L66 (2011).

[35] R. Fisher, *Statistical Methods for Research Workers*, Biological Monographs and Manuals (Oliver and Boyd, Edinburgh, 1925).

[36] M. Kendall and A. Stuart, *The Advanced Theory of Statistics* (Griffin, The University of California, 1969), Vol. 2.

[37] J. F. Kenney and E. S. Keeping, *Mathematics of Statistics, Part II*, 2nd ed. (Van Nostrand, New York, 1951).

[38] E. Lehmann and G. Casella, *Theory of Point Estimation*, Springer Texts in Statistics (Springer, New York, 2003).

[39] H. Cramér, *Mathematical Methods of Statistics* (Princeton University Press, Princeton, 1946).

[40] C. R. Rao, Information and the accuracy attainable in the estimation of statistical parameters, Bull. Calcutta Math. Soc. **37**, 81 (1945).

[41] E. Sellentin, M. Quartin, and L. Amendola, Breaking the spell of Gaussianity: Forecasting with higher order Fisher matrices, Mon. Not. R. Astron. Soc. **441**, 1831 (2014).

[42] Y. Bengio, Learning deep architectures for ai, Found. Trends Netw. **2**, 1 (2009).

[43] G. Cybenko, Approximation by superpositions of a sigmoidal function, Math. Control Signals Syst. **2**, 303 (1989).

[44] L. Deng and D. Yu, Deep learning: Methods and applications, FnT Signal Processing, **7**, 197 (2013).

[45] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning* (MIT Press, Cambridge, 2016).

[46] M. A. Nielsen, *Neural Networks and Deep Learning* (Determination Press, Mountain View, 2015).

[47] W. S. McCulloch and W. Pitts, A logical calculus of the ideas immanent in nervous activity, Bull. Math. Biophys. **5**, 115 (1943).

[48] W. Pitts and W. S. McCulloch, How we know universals the perception of auditory and visual forms, Bull. Math. Biophys. **9**, 127 (1947).

[49] K. He, X. Zhang, S. Ren, and J. Sun, Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification, arXiv:1502.01852.

[50] A. Krizhevsky, I. Sutskever, and G. E. Hinton, Imagenet classification with deep convolutional neural networks, in *Advances in Neural Information Processing Systems* (Curran Associates, Inc., Red Hook, 2012).

[51] A. L. Maas, A. Y. Hannun, and A. Y. Ng, Rectifier nonlinearities improve neural network acoustic models, in *Proceedings of the 30th International Conference on Machine Learning* (2013), Vol. 30, pp. 1–6.

[52] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, Learning representations by back-propagating errors, Nature (London) **323**, 533 (1986).

[53] K. C. Kiwiel, Convergence and efficiency of subgradient methods for quasiconvex minimization, Math. Program. **90**, 1 (2001).

[54] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, Dropout: A simple way to prevent neural

networks from overfitting, J. Mach. Learn. Res. **15**, 1929 (2014).

[55] S. Kullback, *Information Theory and Statistics*, Dover Books on Mathematics (Dover Publications, New York, 1968).

[56] D. B. Rubin, Bayesianly justifiable and relevant frequency calculations for the applied statistician, Ann. Stat. **12**, 1151 (1984).

[57] G. Kitagawa, Monte carlo filter and smoother for non-Gaussian nonlinear state space models, J. Comput. Graph. Stat. **5**, 1 (1996).

[58] X. Glorot and Y. Bengio, Understanding the difficulty of training deep feedforward neural networks, Proc. Mach. Learn. Res. **9**, 249 (2010).

[59] S. Murray, C. Power, and A. Robotham, HMFcalc: An online tool for calculating dark matter halo mass functions (unpublished).

[60] D. J. Eisenstein and W. Hu, Baryonic features in the matter transfer function, Astrophys. J. **496**, 605 (1998).

[61] P. A. R. Ade *et al.*, Planck 2015 results. XIII. Cosmological parameters, Astron. Astrophys. **594**, A13 (2016).

[62] T. P. Robitaille *et al.* (Astropy Collaboration), Astropy: A community Python package for astronomy, Astron. Astrophys. **558**, A33 (2013).

[63] J. E. Bautista, S. Bailey, A. Font-Ribera, M. M. Pieri, N. G. Busca, J. Miralda-Escud, N. Palanque-Delabrouille, J. Rich, K. Dawson, Y. Feng, J. Ge, S. G. A. Gontcho, S. Ho, J. M. L. Goff, P. Noterdaeme, I. Pâris, G. Rossi, and D. Schlegel, Mock quasar-lyman-α forest data-sets for the sdss-iii baryon oscillation spectroscopic survey, J. Cosmol. Astropart. Phys. 05 (2015) 060.

[64] J. E. Bautista *et al.*, Measurement of baryon acoustic oscillation correlations at $z = 2.3$ with SDSS DR12 Lyα-Forests, Astron. Astrophys. **603**, A12 (2017).

[65] P. McDonald, Toward a measurement of the cosmological geometry at z 2: Predicting ly-α forest correlation in three dimensions and the potential of future data sets, Astrophys. J. **585**, 34 (2003).

[66] A. Font-Ribera, P. McDonald, and J. Miralda-Escudé, Generating mock data sets for large-scale Lyman-α forest correlation measurements, J. Cosmol. Astropart. Phys. 01 (2012) 001.

[67] M. Blomqvist, D. Kirkby, J. E. Bautista, A. Arinyo-i-Prats, N. G. Busca, J. Miralda-Escudé, A. Slosar, A. Font-Ribera, D. Margala, D. P. Schneider, and J. A. Vazquez, Broadband distortion modeling in Lyman-α forest BAO fitting, J. Cosmol. Astropart. Phys. 11 (2015) 034.

[68] M. S. Peeples, D. H. Weinberg, R. Dave, M. A. Fardal, and N. Katz, Pressure support vs thermal broadening in the Lyman-alpha forest I: Effects of the equation of state on longitudinal structure, Mon. Not. R. Astron. Soc. **404**, 1281 (2010).

[69] N. Suzuki, D. Tytler, D. Kirkman, J. M. O'Meara, and D. Lubin, Predicting qso continua in the ly forest, Astrophys. J. **618**, 592 (2005).

[70] C. P. Ahn, R. Alexandroff, C. Allende Prieto, S. F. Anderson, T. Anderton, B. H. Andrews, É. Aubourg, S. Bailey, E. Balbinot, R. Barnes *et al.*, The ninth data release of the Sloan Digital Sky Survey: First spectroscopic data from the SDSS-III baryon oscillation spectroscopic survey, Astrophys. J. Suppl. Ser. **203**, 21 (2012).

[71] F. Feroz, J. R. Gair, P. Graff, M. P. Hobson, and A. Lasenby, Classifying LISA gravitational wave burst signals using Bayesian evidence, Classical Quantum Gravity **27**, 075010 (2010).

[72] Automatic physical inference with information maximising neural networks, https://doi.org/10.5281/zenodo.1175196.