Automatic physical inference

Tom Charnock[*]
*Sorbonne Universités, UPMC Université Paris 6 et CNRS, UMR 7095,
Institut d'Astrophysique de Paris, 98 bis boulevard Arago, 75014 Paris, France*

Guilhem Lavaux[†]
*Sorbonne Universités, UPMC Université Paris 6 et CNRS, UMR 7095,
Institut d'Astrophysique de Paris, 98 bis boulevard Arago, 75014 Paris, France and
Sorbonne Universités, UPMC Université Paris 6 et CNRS, UMR 7095,
Institut Lagrange de Paris, 98 bis boulevard Arago, 75014 Paris, France*

Benjamin D. Wandelt[‡]
*Sorbonne Universités, UPMC Université Paris 6 et CNRS, UMR 7095,
Institut d'Astrophysique de Paris, 98 bis boulevard Arago, 75014 Paris, France
Sorbonne Universités, UPMC Université Paris 6 et CNRS, UMR 7095,
Institut Lagrange de Paris, 98 bis boulevard Arago, 75014 Paris, France
Center for Computational Astrophysics, Flatiron Institute,
162 5th Avenue, 10010, New York, NY, USA and
Department of Physics and Astronomy, University of Illinois at Urbana-Champaign, 1002 W Green St, Urbana, IL 61801, USA*

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 reinforcement learning technique that trains artificial neural networks to find non-linear 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, non-linear 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, Euclid, and WFIRST.

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 summarising data is using the Massively Optimised Parameter Estimation and Data (MOPED) compression algorithm [3]. Summaries from MOPED are linear combinations of data that compress the number of data points down to the number of parameters of 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 [3]. The MOPED algorithm has been used on many problems in astronomy and cosmology such as studying the star formation histories of galaxies [13–15], analysing 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 maximising the possible information available, even when the likelihood is known [19].

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

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 ionising 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 summarising data whilst 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 appropriate 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 summarise in a first-step summary ? In this paper we introduce the concept of *information maximising neural networks* (IMNNs). Through the use of machine learning, we can circumvent the two step compression used in [33] and find the most informative non-linear 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 any summary of the data which increases the information further. Once the network is trained, ABC is a trivial next step. Model parameters can be drawn from a prior, used to generate simulations and once they are fed through the network, the optimal summaries of the simulation can be compared to the network summary 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 section I 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 section II we lay out the procedure for creating non-linear summaries of the data. An overview of how artificial neural networks work is presented in section III and we continue in section IV by showing how maximising the determinant of the Fisher information matrix allows a network to be trained to provide the optimal non-linear set of summaries. Next, in section V, we trace the steps to obtain parameter constraints from PMC-ABC using the network trained as prescribed in section IV. Finally, in section VI, we give some test examples. The first test model provides an example where a single linear summary of the data would provide no information about a parameter, but the non-linear 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 summarise 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 non-linear IMNN data compression introduced in this paper can avoid this peculiarity.

I. 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, can be characterised by a model with a set of $n_{\boldsymbol{\vartheta}}$ parameters, $\boldsymbol{\vartheta}$. The Fisher information describes how much information $\mathbf{d}$ contains about $\boldsymbol{\vartheta}$ 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,

$\boldsymbol{\vartheta}^{\rm fid}$,

$$\begin{aligned}\mathbf{F}_{\alpha\beta}(\boldsymbol{\vartheta}) &= \int {\rm d}\mathbf{d}\,\mathscr{L}(\mathbf{d}|\boldsymbol{\vartheta})\,\frac{\partial\ln\mathscr{L}(\mathbf{d}|\boldsymbol{\vartheta})}{\partial\vartheta_\alpha}\frac{\partial\ln\mathscr{L}(\mathbf{d}|\boldsymbol{\vartheta})}{\partial\vartheta_\beta}\bigg|_{\boldsymbol{\vartheta}=\boldsymbol{\vartheta}^{\rm fid}}\\ &= \left\langle\frac{\partial\ln\mathscr{L}(\mathbf{d}|\boldsymbol{\vartheta})}{\partial\vartheta_\alpha}\frac{\partial\ln\mathscr{L}(\mathbf{d}|\boldsymbol{\vartheta})}{\partial\vartheta_\beta}\right\rangle\bigg|_{\boldsymbol{\vartheta}\,=\,\boldsymbol{\vartheta}^{\rm fid}}. \quad (1.1)\end{aligned}$$

Equation (1.1) can be rewritten as

$$\mathbf{F}_{\alpha\beta}(\boldsymbol{\vartheta}) = -\left\langle\frac{\partial^2\ln\mathscr{L}(\mathbf{d}|\boldsymbol{\vartheta})}{\partial\vartheta_\alpha\partial\vartheta_\beta}\right\rangle\bigg|_{\boldsymbol{\vartheta}\,=\,\boldsymbol{\vartheta}^{\rm fid}} \quad (1.2)$$

when the likelihood is twice 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

$$\langle(\vartheta_\alpha - \langle\vartheta_\alpha\rangle)(\vartheta_\beta - \langle\vartheta_\beta\rangle)\rangle \geq \left(\mathbf{F}^{-1}\right)_{\alpha\beta}, \quad (1.3)$$

such that finding the maximum Fisher information, provides the minimum variance for estimators of $\boldsymbol{\vartheta}$. In the case that the likelihood of the data in a particular model is Gaussian, the logarithm of the likelihood can be written as

$$-2\ln\mathscr{L}(\mathbf{d}|\boldsymbol{\vartheta}) = (\mathbf{d}-\mu(\boldsymbol{\vartheta}))^T\mathbf{C}^{-1}(\mathbf{d}-\mu(\boldsymbol{\vartheta})) + \ln|2\pi\mathbf{C}|, \quad (1.4)$$

where $\mathbf{d}$ is the data and $\mu(\boldsymbol{\vartheta})$ is the mean of the model given parameters $\boldsymbol{\vartheta}$, which we will denote $\boldsymbol{\mu}$ for convenience. $\mathbf{C}$ is the covariance of the data and is assumed to be independent of the parameters. Using the MOPED algorithm [3], $\mathbf{d}$ 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 taking a linear combinations of the data. The MOPED compression is lossless in the sense that the Fisher information is conserved under the transformation

$$x_\alpha = \mathbf{r}_\alpha^T\mathbf{d} \quad (1.5)$$

where α labels the parameter and $\mathbf{r}_\alpha$ is calculated by maximising 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

$$\mathbf{r}_1 = \frac{\mathbf{C}^{-1}\boldsymbol{\mu}_{,1}}{\sqrt{\boldsymbol{\mu}_{,1}^T\mathbf{C}^{-1}\boldsymbol{\mu}_{,1}}}, \quad (1.6)$$

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

$$\mathbf{r}_\alpha = \frac{\mathbf{C}^{-1}\boldsymbol{\mu}_{,\alpha} - \sum_{i=1}^{\alpha-1}\left(\boldsymbol{\mu}_{,\alpha}^T\mathbf{r}_i\right)\mathbf{r}_i}{\sqrt{\boldsymbol{\mu}_{,\alpha}^T\mathbf{C}^{-1}\boldsymbol{\mu}_{,\alpha} - \sum_{i=1}^{\alpha-1}\left(\boldsymbol{\mu}_{,\alpha}^T\mathbf{r}_i\right)^2}}. \quad (1.7)$$

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, for the likelihood in equation (1.4). The Fisher information takes the form

$$\mathbf{F}_{\alpha\beta} = \frac{1}{2}{\rm Tr}\left[\boldsymbol{\mu}_{,\alpha}^T\mathbf{C}^{-1}\boldsymbol{\mu}_{,\beta} + \boldsymbol{\mu}_{,\beta}^T\mathbf{C}^{-1}\boldsymbol{\mu}_{,\alpha}\right], \quad (1.8)$$

The lossless compression of the data, $\mathbf{d} \to \mathbf{x}$, is only possible when the likelihood is exactly of the form in equation (1.4). Near, lossless compression is still possible if the peak of the likelihood is approximately Gaussian, but constraints on posterior distributions are exactly Gaussian, peaked at the most likely parameter value with a variance given by the Cramér-Rao bound.

II. NON-LINEAR FISHER INFORMATION MAXIMISING SUMMARIES

We take our influence from the MOPED algorithm to find some transformation which maps the data to compressed summaries, $f : \mathbf{d} \to \mathbf{x}$, whilst conserving Fisher information, but without the limitation that the method is only valid as a Gaussian approximation. The form of the likelihood does not need to be known as long as it can be rewritten in a compressed form as

$$-2\ln\mathscr{L}(\mathbf{x}|\boldsymbol{\vartheta}) = (\mathbf{x} - \mu_f(\boldsymbol{\vartheta}))^T\mathbf{C}_f^{-1}(\mathbf{x} - \mu_f(\boldsymbol{\vartheta})) \quad (2.1)$$

where

$$\mu_f(\boldsymbol{\vartheta}) = \frac{1}{n_\mathbf{s}}\sum_{i=1}^{n_\mathbf{s}}\mathbf{x}_i^\mathbf{s}, \quad (2.2)$$

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

$$(\mathbf{C}_f)_{\alpha\beta} = \frac{1}{n_\mathbf{s}-1}\sum_{i=1}^{n_\mathbf{s}}(\mathbf{x}_i^\mathbf{s} - \boldsymbol{\mu}_f)_\alpha(\mathbf{x}_i^\mathbf{s} - \boldsymbol{\mu}_f)_\beta. \quad (2.3)$$

Using equation (1.2) a modified Fisher information matrix can be calculated from the likelihood in equation (2.1)

$$\mathbf{F}_{\alpha\beta} = \frac{1}{2}{\rm Tr}\left[\boldsymbol{\mu}_{f,\alpha}^T\mathbf{C}_f^{-1}\boldsymbol{\mu}_{f,\beta} + \boldsymbol{\mu}_{f,\beta}^T\mathbf{C}_f^{-1}\boldsymbol{\mu}_{f,\alpha}\right]. \quad (2.4)$$

Here, the values of $\boldsymbol{\mu}_{f,\alpha}$ and $\mathbf{C}_f^{-1}$ are calculated using fixed, fiducial parameter values, $\boldsymbol{\vartheta}^{\rm fid}$, such that the simulations are $\mathbf{d}_i^{\mathbf{s},{\rm fid}} = \mathbf{d}^\mathbf{s}(\boldsymbol{\vartheta}^{\rm fid}, i)$. Although f is not specified, it can be learned using methods of machine learning.

III. ARTIFICIAL NEURAL NETWORKS

Artificial neural networks are arbitrary maps from some inputs to outputs. Consider some data vector $\mathbf{d} = \left\{d_i \middle| \, i \in [1, \, n_{\mathbf{d}}]\right\}$ 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 [41–45]. Each layer, l, of the network contains a set of *neurons* which takes some number of inputs and provides one output per neuron [46, 47]. In the following we set the output of a non-linear activation function to be

$$a_i^l = \phi\left(v_i^l\right) \tag{3.1}$$

where

$$v_j^l = \sum_i w_{ji}^l a_i^{l-1} + b_j^l, \tag{3.2}$$

is a weighted, biased input at each layer with weights $\boldsymbol{w}^l \equiv w_{ji}^l$ and biases $\boldsymbol{b}^l \equiv b_j^l$ [46]. i describes an element of the output vector of a collections of neurons in the $(l-1)^{\text{th}}$ layer and j indexes the neuron in layer l. The activation function, $\phi(v_i^l)$, in equation (3.1) describes whether the artificial neuron *fires* or not, i.e. whether the inputs are informative or useful for describing the output [42, 45, 48, 49]. Currently popular activation functions are the rectified linear unit (`ReLU`) [48]. We show here an adaptation, the leaky `ReLU`

$$\phi\left(x\right) = \left\{ \begin{array}{rl} \alpha x & \underline{\mathrm{x}} \leq 0 \\ x & x > 0 \end{array} \right. , \tag{3.3}$$

where $\alpha = 0$ for `ReLU` and α is small and positive for leaky `ReLU` [50]. Although the `ReLU` family of activation functions are linear, stacking several layers of neurons provides a function which approximates a non-linear 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

$$\frac{\partial\phi\left(x\right)}{\partial x} = \left\{ \begin{array}{ll} \alpha & x \leq 0 \\ 1 & x > 0 \end{array} \right. . \tag{3.4}$$

With these notations, the input to the network can be considered to be the output of a zeroth layer of the 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 [43]. The network output at the final layer can be described by $\boldsymbol{a}^L = \{a_i^L | \, i \in [1, \, n_{\text{outputs}}]\}$ where n_{outputs} is the number of outputs in the final layer, labelled L, and $a_i^L = \phi(v_i^L)$. A scalar loss function, $\Lambda(\boldsymbol{a}^L)$, is calculated from these outputs. In supervised deep learning, the loss function describes how far the outputs are from a set of labels for the training data [51]. An iterative procedure, called back propagation, uses the chain rule to find how much the weights and biases need to change to minimise the loss function [51]. Using gradient descent [52] it can be seen that the weights and biases must be updated using

$$w_{ji}^l \to w_{ji}^l - \eta\frac{\partial\Lambda}{\partial w_{ji}^l} \tag{3.5}$$

and

$$b_i^l \to b_i^l - \eta\frac{\partial\Lambda}{\partial b_i^l}, \tag{3.6}$$

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 [45]. It is very efficient to calculate the derivatives in equations (3.5) and (3.6) at the last layer using

$$\frac{\partial\Lambda}{\partial v_i^L} = \frac{\partial\Lambda}{\partial a_i^L}\frac{\partial a_i^L}{\partial v_i^L}. \tag{3.7}$$

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

$$\frac{\partial\Lambda}{\partial v_i^l} = \sum_j w_{ji}^{l+1}\frac{\partial\Lambda}{\partial v_j^{l+1}}\frac{\partial a_i^l}{\partial v_i^l}. \tag{3.8}$$

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

$$\begin{aligned} \frac{\partial\Lambda}{\partial w_{ji}^l} &= \frac{\partial\Lambda}{\partial v_j^l}\frac{\partial v_j^l}{\partial w_{ji}^l} \\ &= \frac{\partial\Lambda}{\partial v_j^l}a_i^{l-1} \end{aligned} \tag{3.9}$$

and

$$\begin{aligned} \frac{\partial\Lambda}{\partial b_i^l} &= \frac{\partial\Lambda}{\partial v_i^l}\frac{\partial v_i^l}{\partial b_i^l} \\ &= \frac{\partial\Lambda}{\partial v_i^l}. \end{aligned} \tag{3.10}$$

Each of the a_i^l and the derivatives with respect to the weighted biased inputs (using equation (3.4)) 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 [45]. By calculating the change in the loss function with respect to the network outputs and applying equation (3.7) 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 [45]. 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 minimised [45].

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*, a method called dropout can be implemented [53]. 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 sub-networks, where all the neurons share weights and biases. Each of the sub-networks can learn specific features in the data, but the consensus network does not learn features too strongly.

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. Due to tuning the network with respect to the test set, biased values of the accuracy will be obtained. The biasing arises since the test data, which may not be completely representative of the training data, may also not be representative of any *future* data. Using a third, completely unseen, data set to quote network accuracies prevents the irreproducable accuracy scores obtained when considering only a training set and a test set. For expensive simulations, creating a third data set can become very costly and so a balance needs to be considered between the amount of data which can be generated to test the accuracy, and to test for introduced biases from training whilst considering the influence of the test set on training parameters.

IV. FINDING NON-LINEAR SUMMARIES

Inspired by supervised artificial neural networks we are able to create a network capable of maximising the Fisher information to create non-linear 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, $\mathbf{x} \to \mathbf{x}\left(\boldsymbol{\vartheta}, \boldsymbol{w}^l, \boldsymbol{b}^l\right)$. The mean, $\boldsymbol{\mu}_f$, covariance, $\mathbf{C}_f$ and Fisher information matrix, $\mathbf{F}_{\alpha\beta}$, from equations (2.2), (2.3) and (2.4), each become functions of the weights and biases as well. Summaries of simulations, $\mathbf{x}^{\mathbf{s}}_i$, are obtained by passing simulations, $\mathbf{d}^{\mathbf{s}}_i$, through the network $f : \mathbf{d}^{\mathbf{s}}_i \to \mathbf{x}^{\mathbf{s}}_i$.

To compute the Fisher information matrix in equation (2.4), 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

$$\begin{aligned}\boldsymbol{\mu}_{f,\alpha} &= \frac{\partial}{\partial\vartheta_\alpha}\frac{1}{n_{\mathbf{s}}}\sum_{i=1}^{n_{\mathbf{s}}}\mathbf{x}^{\mathbf{s}\,\mathrm{fid}}_i \\ &= \frac{1}{n_{\mathbf{s}}}\sum_{i=1}^{n_{\mathbf{s}}}\left(\frac{\partial\mathbf{x}}{\partial\vartheta_\alpha}\right)^{\mathbf{s}\,\mathrm{fid}}_i . \end{aligned} \tag{4.1}$$

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, $\mathbf{d}^{\mathbf{s}\,\mathrm{fid}}_i = \mathbf{d}^{\mathbf{s}}\left(\boldsymbol{\vartheta}^{\mathrm{fid}}, i\right)$, $\mathbf{d}^{\mathbf{s}\,\mathrm{fid}-}_i = \mathbf{d}^{\mathbf{s}}\left(\boldsymbol{\vartheta}^{\mathrm{fid}} - \Delta\boldsymbol{\vartheta}^-, i\right)$, and $\mathbf{d}^{\mathbf{s}\,\mathrm{fid}+}_i = \mathbf{d}^{\mathbf{s}}\left(\boldsymbol{\vartheta}^{\mathrm{fid}} + \Delta\boldsymbol{\vartheta}^+, i\right)$ 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

$$\left(\frac{\partial\mathbf{x}}{\partial\vartheta_\alpha}\right)^{\mathbf{s}\,\mathrm{fid}}_i \approx \frac{\mathbf{x}^{\mathbf{s}\,\mathrm{fid}+}_i - \mathbf{x}^{\mathbf{s}\,\mathrm{fid}-}_i}{\Delta\vartheta^+_\alpha - \Delta\vartheta^-_\alpha}. \tag{4.2}$$

Setting the random seed, i, to the same value when generating $\mathbf{d}^{\mathbf{s}\,\mathrm{fid}-}_i$ and $\mathbf{d}^{\mathbf{s}\,\mathrm{fid}+}_i$ 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

$$\boldsymbol{\mu}_{f,\alpha} = \frac{1}{n_{\mathbf{s}}}\sum_{i=1}^{n_{\mathbf{s}}}\sum_{k=1}^{n_{\mathbf{d}}}\frac{\partial f}{\partial d_k}\left(\mathbf{d}^{\mathbf{s}\,\mathrm{fid}}_i\right)\frac{\partial d^{\mathbf{s}\,\mathrm{fid}}_{ik}}{\partial\vartheta_\alpha}, \tag{4.3}$$

where i labels the random initialisation 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 equations (4.1) and (4.2).

One simple way of obtaining the maximal non-linear summary of some data is to maximise the determinant of the Fisher information matrix calculated from the network, $|\mathbf{F}|$,

$$\Lambda = -\frac{1}{2}|\mathbf{F}|^2. \tag{4.4}$$

The maximal determinant of the Fisher information matrix (its logarithm to be exact) describes the Shannon information, which indicates the total information content of a Gaussian. The error is then found by taking the derivative of the loss function with respect to the network output. Normally $\mathbf{x}^{\mathbf{s}}_i$ 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 figure 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 initialisation (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 equation (2.3). Meanwhile, the rest of the network outputs are used to find the derivative of the mean with respect to the parameter as in equations (4.1) and (4.2). These are combined to give the Fisher information matrix of equation (2.4). If we consider the *true* network output to be $\boldsymbol{a}^L = |\mathbf{F}|$ rather than $\mathbf{x}^{\mathbf{s}}$ then the error can be defined as

$$\frac{\partial\Lambda}{\partial\boldsymbol{a}^L} = -|\mathbf{F}|. \tag{4.5}$$

Training then commences over many epochs of weight and bias updates until the Fisher information stops increasing. In practice, a problem arises when using equation (4.5), 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

$$\frac{\partial\Lambda}{\partial\boldsymbol{a}^L} = -|\mathbf{F}| + |\mathbf{C}_f|. \tag{4.6}$$

The network is penalised when the determinant of the covariance is large. When using equation (4.6) the network provides the summary which maximises the Fisher information whilst minimising the covariance of the outputs.

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 number of summaries already obtained. For example, if the power spectrum is a known useful summary of some data, the Fisher information for that statistic can be calculated and the network can be trained to combine the rest of the data to find any statistic which increases the Fisher information further. Any inexhausted combination in the data would become usable, even if the combination is not known.

V. APPROXIMATE BAYESIAN COMPUTATION

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 [54]. It is useful to choose an appropriate sampling procedure to quickly approach the true posterior for the parameters without creating too many simulations. Population Monte Carlo (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 [55]. 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 [30], our PMC-ABC algorithm starts by drawing N parameter values, $\{\boldsymbol{\vartheta}^t_k | k \in [1, N], t = 0\}$, 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. Simulations are made at each of the N parameter values 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, once a random initialisation i is chosen for each simulation, its value can be ignored. The distance of each simulated summary from the summary of the real data $\mathbf{x}$ is given by

$$\varrho^t_k = \sqrt{\left(\mathbf{x}^{\mathbf{s}\,t}_{ik} - \mathbf{x}\right)^T \mathbf{F}\left(\mathbf{x}^{\mathbf{s}\,t}_{ik} - \mathbf{x}\right)}, \tag{5.1}$$

where $\mathbf{F}$ is the Fisher information matrix obtained originally by the network. 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 parameters 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

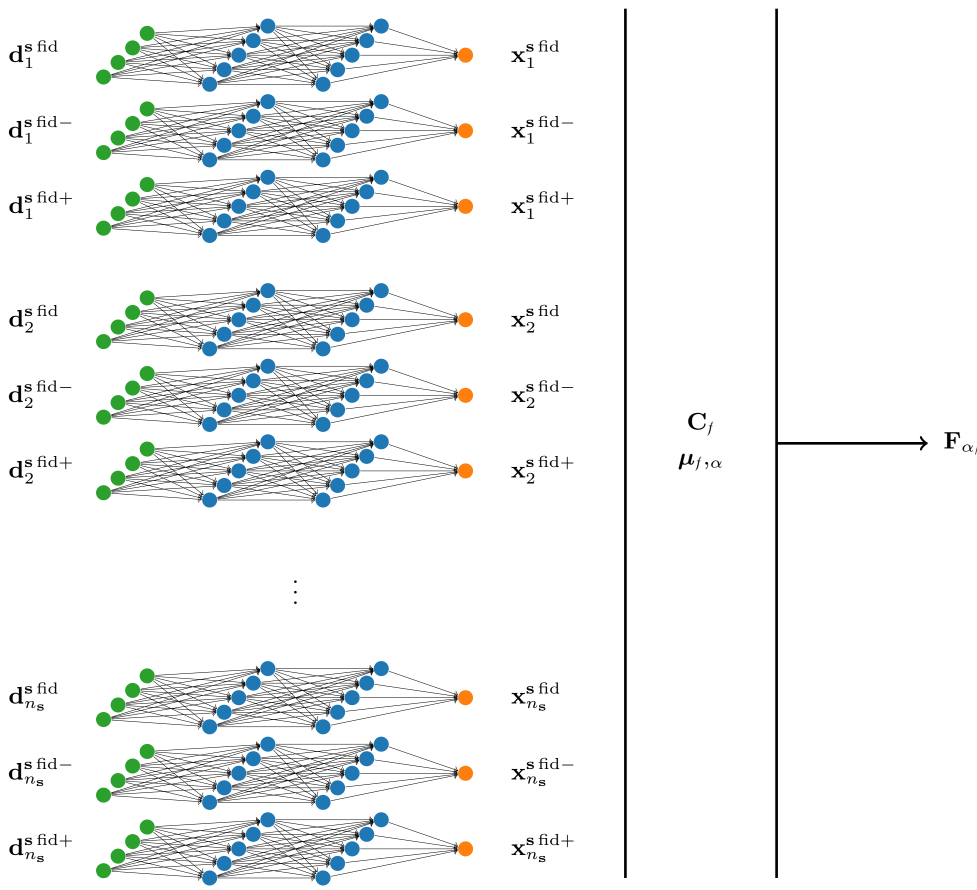
$$W^{t+1}_k = \frac{p(\boldsymbol{\vartheta}^{t+1}_k)}{\sum_{j=1}^N W^t_j \mathcal{N}(\boldsymbol{\vartheta}^{t+1}_k; \boldsymbol{\vartheta}^t_j, \mathbf{C}_t)} \tag{5.2}$$

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

$$\mathcal{N}(\boldsymbol{\vartheta}^{t+1}_k; \boldsymbol{\vartheta}^t_j, \mathbf{C}_t) = \frac{\exp\left[-\frac{1}{2}\left(\boldsymbol{\vartheta}^{t+1}_k - \boldsymbol{\vartheta}^t_j\right)^T \mathbf{C}_t^{-1}\left(\boldsymbol{\vartheta}^{t+1}_k - \boldsymbol{\vartheta}^t_j\right)\right]}{\sqrt{|2\pi\mathbf{C}_t|}} \tag{5.3}$$

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 $\varrho^{t+1}_k \leq \varepsilon^t$ 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 [30] 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



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

FIGURE 1: Cartoon of the information maximising neural network architecture. During training each simulation $\mathbf{d}_i^{\mathbf{s},\mathrm{fid}}$ and each simulation made with a varied fiducial parameter, $\mathbf{d}_i^{\mathbf{s}\,\mathrm{fid}\pm}$, is passed through the same network (all the weights and biases are shared). The output of the network for each simulation, $\mathbf{x}_i^{\mathbf{s}\,\mathrm{fid}}$, is used to calculate the covariance, $\mathbf{C}_f$, and each of the network outputs from the varied simulations, $\mathbf{x}_i^{\mathbf{s}\,\mathrm{fid}\pm}$, 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 maximised 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, $\mathbf{x}_i^{\mathbf{s}\,\mathrm{fid}}$. 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.

$\boldsymbol{\vartheta}_k^{t+1}$ used to find $\mathbf{C}_{t+1}$. The next acceptance condition, ε^{t+1}, is again calculated from the 75th percentile of $\{\varrho_k^{t+1}|\,k\in[1,\,N]\}$ and the selection procedure is repeated. Iterations can be performed until the number of draws from $\mathcal{N}(\boldsymbol{\vartheta}_k^t,\mathbf{C}_t)$ 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.

VI. TESTING THE NETWORK

In this section we apply the information maximising neural network to summarise data allows use to perform parameter inference on a range of test models. In section VI A we use the network to summarise Gaussian noise with unknown variance, as well as Gaussian noise with unknown variance where some element of the variance comes from noise. In section VI B we constrain the amplitude of scalar perturbations using simulations of quasar absorption spectra which can be summarised by a single statistic provided by the network. Finally, in section VI C, we address a concern in [34] which shows that a linear summary of data is the time domain can be misleading about a parameter in the frequency domain, but we are able to show that the non-linear summary is more informative.

A. Summarising 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, $\mathbf{d} = \left\{d_i \curvearrowleft \mathcal{N}(0, \vartheta) \,\middle|\, i \in [1, n_{\mathbf{d}}]\right\}$. The likelihood is written

$$\begin{aligned}\mathcal{L}(\mathbf{d}|\vartheta) &= \prod_{i=1}^{n_{\mathbf{d}}} \frac{1}{\sqrt{2\pi\vartheta}} \exp\left[-\frac{1}{2\vartheta}d_i^2\right] \\ &= \frac{1}{(2\pi\vartheta)^{n_{\mathbf{d}}/2}} \exp\left[-\frac{1}{2\vartheta}\sum_{i=1}^{n_{\mathbf{d}}} d_i^2\right], \qquad (6.1)\end{aligned}$$

such that

$$-2\ln\mathcal{L}(\mathbf{d}|\vartheta) = \frac{1}{\vartheta}\sum_{i=1}^{n_{\mathbf{d}}} d_i^2 + n_{\mathbf{d}} \ln[2\pi\vartheta]. \qquad (6.2)$$

From here it can be seen that there is a single sufficient statistic, i.e. the sum of the square of the data

$$x = \sum_{i=1}^{n_{\mathbf{d}}} d_i^2. \qquad (6.3)$$

Maximising the (logarithm of the) likelihood with respect to the variance relates the value of the statistic to the variance

$$\begin{aligned}\frac{\partial \ln\mathcal{L}(x|\vartheta)}{\partial\vartheta} &= \frac{x}{2\vartheta^2} - \frac{n_{\mathbf{d}}}{2\vartheta} \\ &= 0 \qquad (6.4)\end{aligned}$$

so that

$$x = n_{\mathbf{d}}\vartheta. \qquad (6.5)$$

The Fisher information is calculated using equation (1.2)

$$\begin{aligned}\mathbf{F} &= \frac{x}{(\vartheta^{\rm fid})^3} - \frac{n_{\mathbf{d}}}{2(\vartheta^{\rm fid})^2} \\ &= \frac{n_{\mathbf{d}}}{2(\vartheta^{\rm fid})^2}. \qquad (6.6)\end{aligned}$$

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

$$\mathbf{F} = 5. \qquad (6.7)$$

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

Now consider training a network to maximise the Fisher information whilst summarising the data, as laid out in section IV. The training of an example network is shown in figure 2. The fully connected network has two hidden layers with 256 neurons in each, $[256, 256]$, using leaky `ReLU` activation with $\alpha = 0.01$ and a learning rate of $\eta = 0.01$. Each of the weights, $\boldsymbol{w}^l$, are initialised 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 [48]. As is usual when using the `ReLU` family of activation functions, the biases $\boldsymbol{b}^l$ are initialised with a slightly positive value [56], where $\boldsymbol{b}^l = 0.1$ has been chosen here. To echo 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 $\boldsymbol{\mu}_{f,\alpha}$ and $\mathbf{C}_f$ 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.

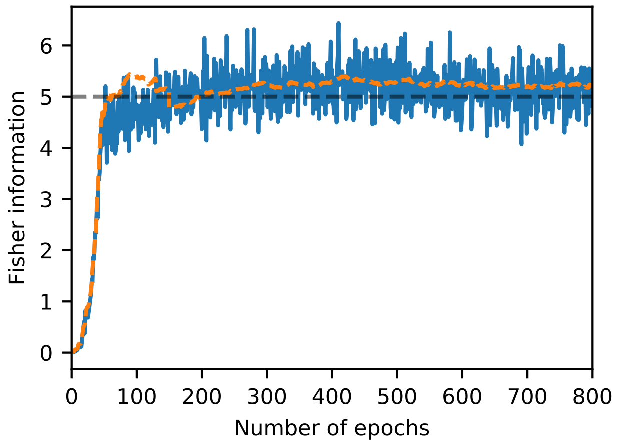


FIGURE 2: Value of the Fisher information obtained by the network at the end of each epoch of training. The solid blue line shows the Fisher information obtained by running a set of 500 simulations (and 50 partial derivatives), which are contained in the training set, through the network. The dashed 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 $\mathbf{F} = 5$, show as a black dashed line. It is clear that the network manages to extract the entirety of the information given the data.

From equation (6.7), 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 figure 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), whilst 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

Data	Value
d_1	-0.91903399
d_2	-0.37322515
d_3	-0.05613342
d_4	1.20816746
d_5	0.07649269
d_6	-0.47171141
d_7	-1.4756571
d_8	-0.62946463
d_9	-1.30334079
d_{10}	-0.41441639

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

straight black dashed line since the training and test data are biased, with a statistically slightly smaller variance than $\vartheta = 1$. The network interprets the bias in the data as an indication that more information about the parameters is available from the network than is truly available. We do not use a third *future* data set as specified at the end of section III to quote the final Fisher information value as the Fisher information from the training data is a product we use in the PMC-ABC stage.

It should be noted that although the architecture will be important, we have found that a very large variety of hyperparameters 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 maximise 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 *real* data $\mathbf{d}$ (shown in table I) can be found and is plotted as the solid orange curve in figure 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 the asymptotic likelihood by expanding equation (2.1) about the fiducial variance with $\Delta\vartheta = (-1, 9]$[1]. The asymptotic likelihood result is plotted in figure 3 in dashed blue. It can be seen that the peak of the posterior found using the asymptotic likelihood corresponds with

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

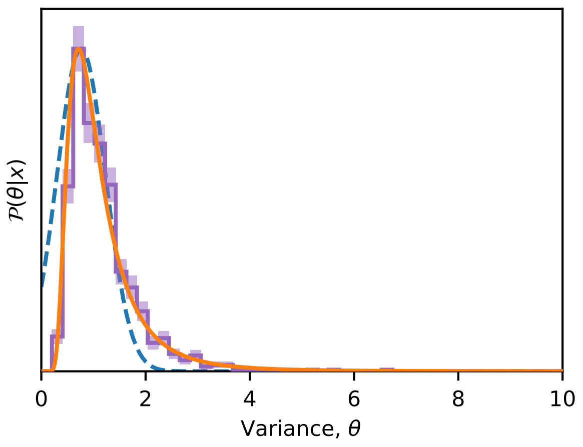


FIGURE 3: The posterior distribution for the variance of the real data. The solid orange curve is the analytic posterior distribution using Bayes' theorem and the likelihood in equation (6.1). The dashed blue curve shows the posterior calculated from the asymptotic likelihood from the network summary and in purple is the ABC posterior obtained through PMC with the purple shaded error bars showing the 1-σ Poisson width. The curves are normalised to peak at the highest bins of the purple histogram. 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 summarise 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 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, $p(\boldsymbol{\vartheta})$, 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 $\varepsilon^T = 0.086$ at the last iteration, T, meaning that the network summary of each of the accepted network summaries are within a band of $x_{ik}^{\mathbf{s}\,T} = x \pm 0.086$ of the network summary of the real data, x. The histogram of the accepted points are shown in figure 3 in purple. The PMC-ABC posterior distribution follows the analytic posterior distribution exactly, showing that the network has successfully learned how to summarise the data.

It is interesting to see the network outputs as a function of ϑ, without using the PMC procedure. By performing naïve ABC we can plot the network output as a function of the variance drawn from the prior, shown in figure 4. The green points show the rejected samples and the purple points (under the black dashed line) show

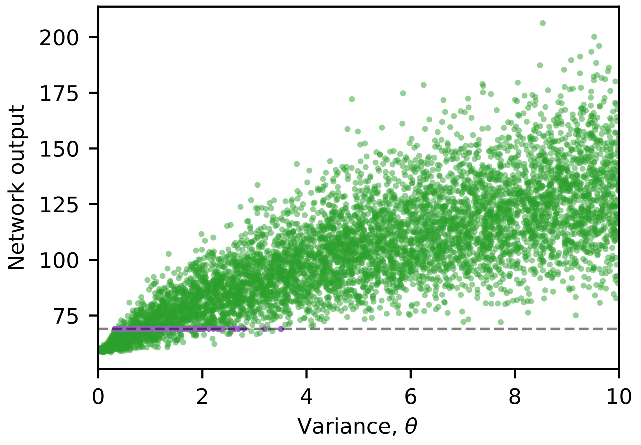


FIGURE 4: Network output as a function of the variance used to create the simulations. The green points are the network summaries of a selection of the simulations created from random draws from $\vartheta = (0, 10]$ for the random ABC procedure. The purple points are the accepted network summaries of the 1000 simulations within $x_{ik}^{\mathbf{s}\,T} = x \pm \varepsilon^T$ with $\varepsilon^T = 0.086$. The black dotted line indicates the network output of the real data.

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 $x_{ik}^{\mathbf{s}\,T} = x \pm \varepsilon^T$, where $\varepsilon^T = 0.086$, necessitates more than 600,000 draws from the prior, 50 times more 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 initialisation 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 equation (4.6).

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

$$x_i^{\mathbf{s}} = \sum_{j=1}^{n_{\mathbf{d}}} \left(d_{ij}^{\mathbf{s}}\right)^2 \tag{6.8}$$

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

$$\text{network summary} \approx \sum_{j=1}^{n_{\mathbf{d}}} \left(d_{ij}^{\mathbf{s}}\right)^2 + 58, \tag{6.9}$$

without a linear scaling of the true summary, 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 summary 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 sub-optimal 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 figure 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. *Summarising Gaussian signals with known noise variance*

Now consider some noisy data where the real data $\mathbf{d} = \left\{d_i \backsim \mathcal{N}\left(0, \vartheta + \sigma^2_{\text{noise}}\right) \middle| \, i \in [1, n_{\mathbf{d}}]\right\}$ has a signal variance of $\vartheta^{\text{true}} = 1$ and the variance of the noise is taken to be known $\sigma^2_{\text{noise}} = 1$. 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 equation (6.6) 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 neurons in two hidden layers, [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 figure 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 *real* 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

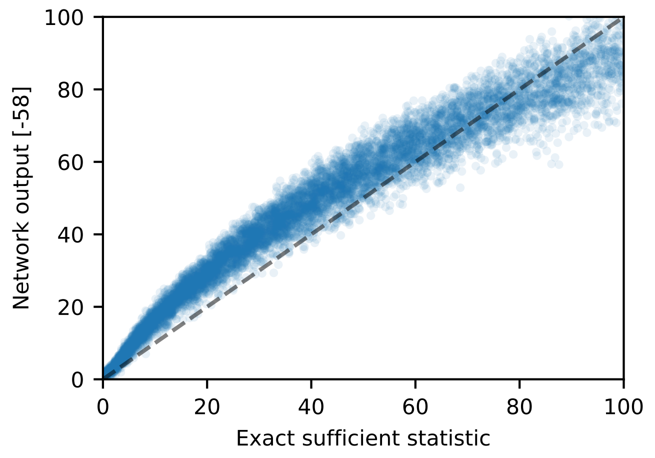


FIGURE 5: Rescaled network output for a given real summary. The blue dots show a scaled value of the network output at the exact sufficient statistic of simulations obtained at a range of ϑ during ABC. The black dashed line shows the expected value of the network summary if the network had learned the map from data to true summary 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 summary. 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 map which shows that the map from data to summary is only approximately correct.

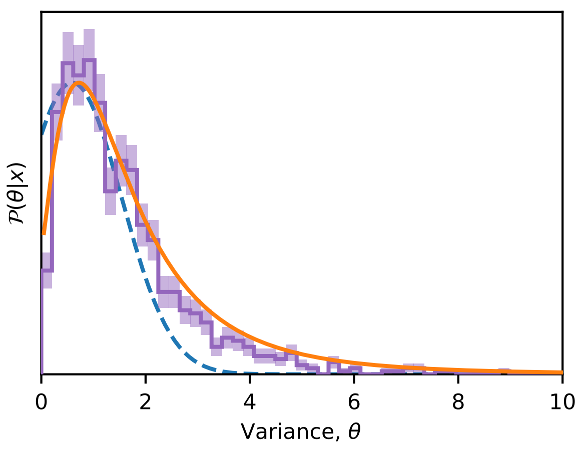


FIGURE 6: The posterior distribution of the signal variance of the Gaussian noise when the data is contaminated with known noise of $\sigma^2_{\rm noise} = 1$. The solid orange line shows the analytic posterior distribution, whilst the posterior distribution from the asymptotic likelihood is shown in dashed blue and the purple histogram with 1-σ Poisson shaded error bars shows the approximate posterior distribution from PMC-ABC. The dashed blue and solid orange curves are normalised such that their maxima coincides with the average height over the 4 highest bins. 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 summarise the data.

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 Fisher information at the end of training. There are 1000 samples used to create the histogram of the PMC-ABC posterior which required 16231 simulations to be created during the PMC, where all samples are within $x^{\mathbf{s}\,T}_{ik} = x \pm \varepsilon^T$, with $\varepsilon^T = 0.109$. 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 summarise the data.

2. Summarising Gaussian signals with wrong fiducial variance

Since the network trained in the known noise problem, in section VI A 1, is akin to a network trained at a fiducial parameter $\vartheta^{\rm fid} = 2$, 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 expected that data with $\vartheta = 1$ would be underrepresented 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 section VI A. However, figure 7 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 section VI 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 figure 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 the network from section VI 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 summarise 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

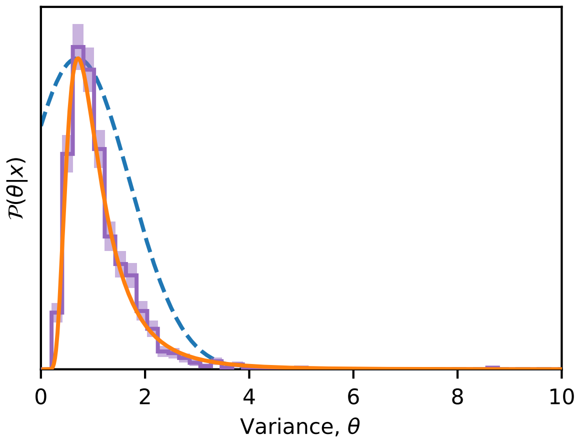


FIGURE 7: The posterior distribution of parameter ϑ where the real data is that of table I, but the network has been trained with a fiducial $\vartheta^{\rm fid} \neq \vartheta^{\rm true}$. The solid orange line shows the analytic posterior distribution and the purple histogram, with shaded 1-σ Poisson widths, shows the approximate posterior distribution from PMC-ABC. The dashed blue curve shows the posterior distribution from the asymptotic likelihood. The dashed blue and solid orange curves are normalised to the average over the maximum 2 bins of the purple histogram. 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.

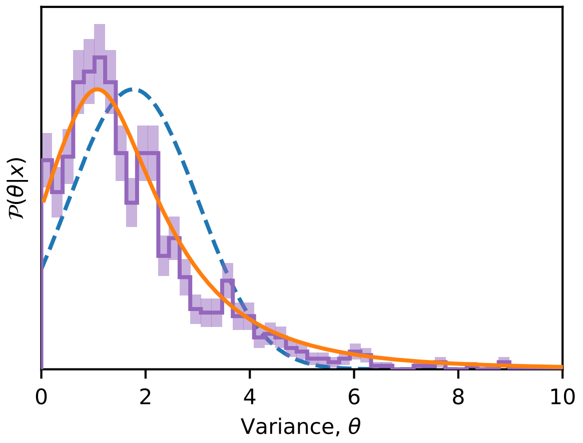


FIGURE 8: The posterior distribution of the signal variance when the data is contaminated with unknown noise $\sigma^2_{\rm noise} = (0, 2]$. The exact posterior is shown by the solid orange curve, the posterior distribution obtained using the asymptotic likelihood is in dashed blue and the PMC-ABC posterior with samples drawn using PMC is indicated by the purple histogram with shaded 1-σ Poisson error bars. The solid orange and dashed blue curve are normalised to the average over the five maximum bins of the purple histogram. Even with unknown noise the network can summarise 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.

sum of the square of the input.

3. *Summarising Gaussian signals with unknown noise variance*

Now consider the problem where, again, the real data $\mathbf{d} = \{d_i \curvearrowleft \mathcal{N}(0, \vartheta + \sigma^2_{\rm noise}) | i \in [1, n_{\mathbf{d}}]\}$ has a signal variance of $\vartheta = 1$ and the variance of the noise is also unknown with a uniform prior $\sigma^2_{\rm noise} \in (0, 2]$. We train using 1000 simulations (+100 for each of the derivatives) at a fiducial $\vartheta^{\rm fid} = 1$ each with a different $\sigma^2_{\rm noise}$ 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 $\sigma^2_{\rm noise} = 2$ then the maximum Fisher available, as calculated from equation (6.6) would be $\mathbf{F} = 5/9$. The posterior distributions for ϑ are shown in figure 8. 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 $\sigma^2_{\rm noise} = (0, 2]$ 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 11892 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 *real* 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 summarise data independent of the input, i.e. the network calculates the sum of the square of the input.

B. Summarising 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 [57] in `python` to generate the 3D power spectrum $P^{\rm 3D}(k)$, evolved using the method of Eisenstein and Hu [58], 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 polarisation results [59], $H_0 = 67.7$ km Mpc^{-1}s^{-1}, $\Omega_\mathrm{m} = 0.307$, $\Omega_\mathrm{b} = 0.0486$, $n_\mathrm{s} = 0.9667$, $\sigma_8 = 0.8159$, $T_\mathrm{CMB} = 2.725$ K, $N_\mathrm{eff} = 3.05$, and $\sum m_\nu = 0.06$ eV, calculated using `astropy` [60]. The power spectrum is calculated between $\ln k_\mathrm{min}/(1h\mathrm{Mpc}^{-1}) = -18.42$ and $\ln k_\mathrm{max}/(1h\mathrm{Mpc}^{-1}) = 9.90$ in steps of $\Delta \ln k/(1h\mathrm{Mpc}^{-1}) = 0.005$. The correlation function can be found using

$$\xi(r) = \int_0^\infty \frac{dk}{2\pi^2} \exp\left[-R_w^2 k^2\right] k^2 P^\mathrm{3D}(k)\mathrm{sinc}(kr) \quad (6.10)$$

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

$$P^\mathrm{1D}(k) = \int_{-\infty}^{\infty} dr \exp[ikr]\xi(r). \quad (6.11)$$

The Lyman-α peak in the rest frame of an emitter is $\lambda^\alpha_\mathrm{RF} = 121.567$nm [61] and we use the fact that BOSS can measure absorbers in the redshift range $1.96 < z < 3.44$ [62]. Using

$$z = \frac{\lambda}{\lambda_\mathrm{RF}} - 1 \quad (6.12)$$

the minimum observed wavelength of the Lyman-α peak is $\lambda_\mathrm{min} = 359.838$ nm (at $z = 1.96$) and the maximum wavelength is $\lambda_\mathrm{max} = 539.757$ nm (at $z = 3.44$) [62]. 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 $k = (0, 14.6]h\,\mathrm{Mpc}^{-1}$ 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 [63],

$$P^\mathrm{1D}_\mathrm{g}(k) = \beta D(k, \mu) P^\mathrm{1D}(k) \quad (6.13)$$

where β is a free parameter, set for a given realisation of noise which ensures that $\langle F\rangle = 0.8$ [64]. $D(k, \mu)$ is a term which modifies the small-scale power spectrum [63] and is of the form

$$D(k, \mu) = \exp\left[\left(\frac{k}{k_\mathrm{NL}}\right)^{\alpha_\mathrm{NL}} - \left(\frac{k}{k_\mathrm{P}}\right)^{\alpha_\mathrm{P}} - \left(\frac{k_\|}{k_\mathrm{V}}\right)^{\alpha_\mathrm{V}}\right], \quad (6.14)$$

where

$$k_\mathrm{V} = k_{\mathrm{V}_0}\left(1 + \frac{k}{k'_\mathrm{V}}\right)^{\alpha'_\mathrm{V}} \quad (6.15)$$

and $k_\mathrm{NL} = 6.40h\,\mathrm{Mpc}^{-1}$, $\alpha_\mathrm{NL} = 0.569$, $k_\mathrm{P} = 15.3h\,\mathrm{Mpc}^{-1}$, $\alpha_\mathrm{P} = 2.01$, $k_{\mathrm{V}_0} = 1.220$, $k'_\mathrm{V} = 0.923h\,\mathrm{Mpc}^{-1}$, $\alpha'_\mathrm{V} = 0.451$, $\alpha_\mathrm{V} = 1.50$ [65] and we choose to use $\mu = k_{||}/k = 1$ 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 [63]. For the purpose of demonstration we keep the same k dependence here. With the gas power spectrum in equation (6.13), normalised by the length of the survey, we can generate 1D random Gaussian fields, δ_g. The Gaussian fields are generated by multiplying unit variance, zero mean Gaussian noise with $\left(P^\mathrm{1D}_\mathrm{g}(k)/2\right)^{1/2}$ and Fourier transforming into real space, including the normalisation 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 over-densities in the density field, and can be calculated from the fluctuating Gunn-Peterson approximation [66] as

$$F = \exp\left[-\tau\right] \quad (6.16)$$

where we consider the form of the optical depth to be

$$\tau = 1.54\left(\frac{T_0}{10^4\mathrm{K}}\right)^{-0.7} \frac{10^{-12}\mathrm{s}^{-1}}{\Gamma_\mathrm{UV}}\left(\frac{1+z}{1+3}\right)^6 \quad (6.17)$$

$$\times \frac{0.7}{h}\left(\frac{\Omega_\mathrm{b}h^2}{0.02156}\right)^2 \frac{4.0927}{H(z)/H_0}\rho^{2-0.7\gamma} \quad (6.18)$$

where $T_0 = 18400$K is the normalisation to the power-law temperature-density relation $T = T_0(1+\delta_g)^{\gamma-1}$ with $\gamma = 0.29$ (both here and in equation (6.18)) and $\Gamma_\mathrm{UV} = 4 \times 10^{-12}s^{-1}$ is the photo-ionisation rate due to the ambient UV background [66]. The gas density field is normalised such that its mean is unity,

$$\rho = \frac{\exp\left[\delta_\mathrm{g}\right]}{\langle\exp\left[\delta_\mathrm{g}\right]\rangle}. \quad (6.19)$$

The continuum flux can be calculated between the Lyman-α and Lyman-β peaks at $\lambda^\alpha_\mathrm{RF} = 121.567$ nm and $\lambda^\beta_\mathrm{RF} = 102.572$ nm using the PCA formulation of [67]. The continuum flux in the rest frame of the emitter is calculated using

$$r(\lambda) = \mu(\lambda) + \sum_i c_i(\lambda)\xi_i(\lambda) \quad (6.20)$$

where $\mu(\lambda)$ is the mean flux over many quasars, $\xi_i(\lambda)$ are the i^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 equation (6.12). 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 equation (6.12). The continuum modulated flux from the quasar is simply

$$f(\lambda) = F(\lambda)C(\lambda) \quad (6.21)$$

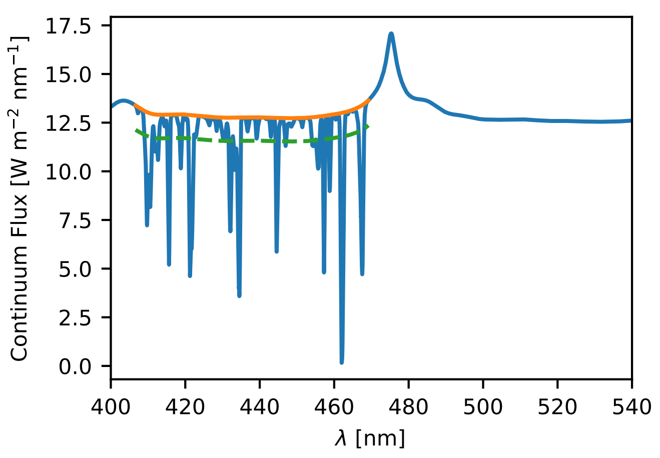


Figure 9: Simulated spectrum of a quasar at $z = 2.91$ in blue, with the value of the continuum in orange (light) and the mean flux in the Lyman-α forest in dashed green.

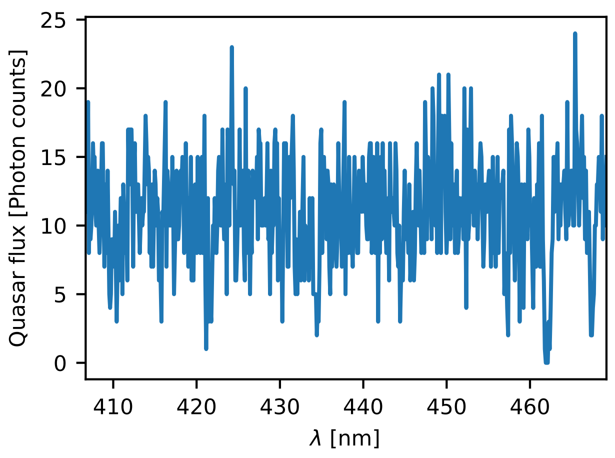


Figure 10: Simulated observation of a quasar spectrum from a quasar at $z = 2.91$ between the Lyman-α and Lyman-β peaks in the rest frame of the observer. We use this data as our *real* data for the PMC-ABC.

where $C(\lambda)$ is $r(\lambda)$ from equation (6.20) in the rest frame of the observer [61]. 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 406.6nm $< \lambda <$ 469.2nm which is 104nm $< \lambda <$ 120nm in the rest frame of the quasar [62]. We bin the wavelengths using the resolution from the BOSS coadded spectra of $\Delta \log_{10} \lambda/1\text{nm} = 10^{-4}$ [61] which gives a flux in Wm^{-2}nm^{-1}, but needs to be measured in photon counts. Using the method[2] in [68] 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) [68]. 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 *real* data input to the network is shown in figure 10.

For any set of fixed cosmological parameters the value of amplitude of scalar perturbations, A_s, is a scaling of $P_\text{g}^\text{1D}(k)$. 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 *real* data. In fact, for simplicity, we can consider the parameter ϑ to be some multiplicative scaling of the amplitude, $A_\text{s} = \vartheta A_\text{cosmo}$ with A_cosmo the amplitude of the power spectrum found in equation (6.13). We use $\vartheta^\text{fid} = \exp[0]$ as the fiducial parameter, i.e. $A_\text{s}^\text{fid} = A_\text{cosmo}$.

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 10000 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 *real* data was created away from the fiducial parameter value of $\vartheta^\text{fid} = \exp[0]$ at $\vartheta^\text{real} = \exp[3]$, i.e. $A_\text{s} = \exp[3]A_\text{cosmo}$, and is shown in figure 10. The posterior distribution for the value of ϑ can be found in figure 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 $A_\text{s} \approx \exp[3]A_\text{cosmo}$, i.e. $\ln \vartheta \approx 3$, which confirms that the correct *real* parameter can be recovered, shown as the vertical black dashed line in figure 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

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.

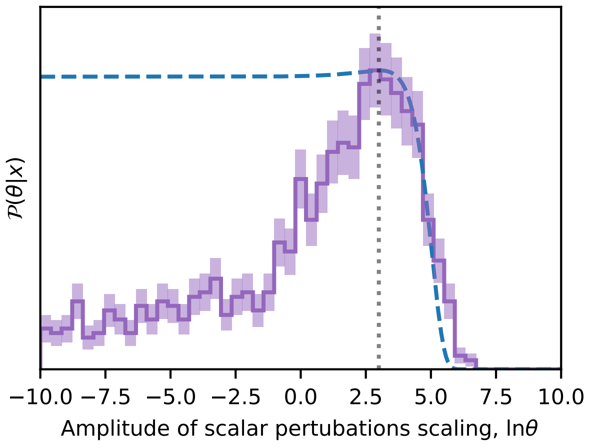


FIGURE 11: Posterior distribution for the scaling of the amplitude of scalar perturbations, $\ln\vartheta$. The dashed blue curve shows the Gaussian approximation to the true constraints as estimated with the training simulations and the purple histogram, with shaded 1-σ Poisson error bars, is calculated 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 summarise extremely noisy data by solely maximising the Fisher information.

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 low ϑ is even clearer in the posterior from the leading order expansion. The constraints on $A_{\rm s}$ 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 summarise it in such a way that we can perform PMC-ABC to get a posterior distribution for parameters of interest.

C. Gravitational waveform frequency

[34] showed that the MOPED algorithm, described in section I, was unable to summarise the central oscillation frequency of a gravitational waveform from LISA without introducing spurious features [34]. By using non-linear 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 [69] of

$$\overline{h}(f) = \frac{A}{2}\sqrt{\frac{Q^2}{2\pi f_{\rm c}^2}}\exp\left[-\frac{Q^2}{2}\left(\frac{f-f_{\rm c}}{f_{\rm c}}\right)^2\right]\exp\left[2\pi i f_{\rm c} t_{\rm c}\right], \tag{6.22}$$

where A is some amplitude, Q is the width of the gravitational wave burst, $t_{\rm c}$ is the time of the burst and $f_{\rm c}$ is the central oscillation frequency. We fix $Q = 5$ and $t_{\rm c} = 1\times 10^5 s$ and scale A such that the signal-to-noise of the burst is $S/N = 10$ [34]. Note that we are using a S/N less than in [34] to introduce more noise. More noise in the simulations allows the IMNN to find greater sample variance in the summaries, which is useful for training. We are interested in summarising and constraining the parameter $f_{\rm c}$. To generate a simulation of the gravitational wave signal, we use the one-sided noise power spectral density of the LISA detector [69], which is

$$S_h(f) = 16\sin^2\left[2\pi f t_{\rm L}\right]\Big(2S_{\rm pn}\big(1+\cos\left[2\pi f t_{\rm L}\right] + \cos^2\left[2\pi f t_{\rm L}\right]\big) + \big(\cos\left[2\pi f t_{\rm L}\right]/2+1\big)S_{\rm sn}f^2\Big), \tag{6.23}$$

$$S_{\rm pn}(f) = \left(1+\left(\frac{10^{-4}{\rm Hz}}{f}\right)^2\right)\frac{S_{\rm acc}}{f^2}, \tag{6.24}$$

where $S_{\rm sn} = 1.8\times10^{-37}{\rm Hz}^{-1}$ is the shot noise, $S_{\rm acc} = 2.5\times10^{-48}{\rm Hz}^{-1}$ is the proof acceleration mass and $t_{\rm L} = 16.678s$ 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 $\overline{h}(f)$ and detector noise $\overline{n}(f)$ and then Fourier transform them into real space

$$\overline{h}(f) = \int_{-\infty}^{\infty} dt\, h(t)\exp[2\pi i f t], \tag{6.25}$$

$$\overline{n}(f) = \int_{-\infty}^{\infty} dt\, n(t)\exp[2\pi i f t]. \tag{6.26}$$

We perform the Fourier transform at 2048 time steps from $t = 9.9\times10^4 s$, sampled at $1s$ intervals. The output of the LISA detector is then given by

$$\mathbf{d} = \mathbf{h}(\boldsymbol{\vartheta}^{\rm true}) + \mathbf{n} \tag{6.27}$$

where $\mathbf{h}(\boldsymbol{\vartheta}^{\rm true})$ is the values of the gravitational waveform at the true parameter values, $\boldsymbol{\vartheta}^{\rm true} = \{A^{\rm true}, Q^{\rm true}, t_{\rm c}^{\rm true}, f_{\rm c}^{\rm true}\}$ at the sampled time and $\mathbf{n}$ is a random realisation of the noise. The logarithm of the likelihood is particularly simple [69] and is given by

$$\ln\mathcal{L} = C - \frac{||\mathbf{d}-\mathbf{h}(\boldsymbol{\vartheta})||^2}{2}, \tag{6.28}$$

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

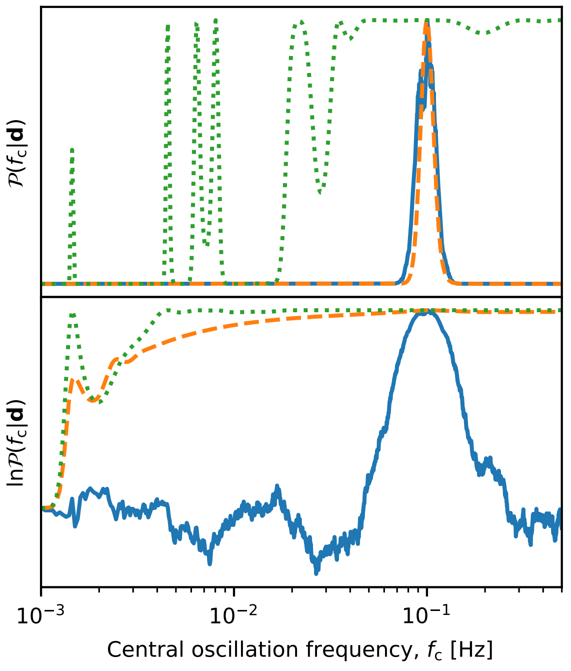


FIGURE 12: (Upper panel) The posterior of the central oscillation frequency, f_c. (Lower panel) The logarithm of the posterior for the central oscillation frequency, f_c. In both subplots, the orange dashed line shows the true posterior, or its logarithm, calculated using equation (6.28) without summarising the data. The solid blue curve shows the posterior, or its logarithm, from the summary from the network, calculated using equation (6.29). The dotted green line shows the posterior, or its logarithm, using (6.29), when the summary comes from the MOPED algorithm. All curves are normalised at $f_c = 1 \times 10^{-3}$Hz in the lower subplot and are normalised to peak at maximum of one in the upper subplot. The plot shows that, although the logarithm of the posterior for f_c using the MOPED summary is more similar to the logarithm of the posterior with all the data than when using the network summary, the network summary is better suited to constraining the parameter since there is no aliasing.

we set to zero. We are interested in summarising 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 and has five hidden layers, each decreasing in size, [1024, 512, 256, 128, 64]. The network has no dropout and, different to the previous networks, we use a tangent function as the activation function for each neuron. The learning rate is fixed at $\eta = 0.01$ and the biases are initialised slightly positively at $\boldsymbol{b}^l = 0.1$. We train for 2000 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 summarise the data. In fact, we can also use equation (6.28) to calculate the logarithm of the likelihood from the summary. We pass the true data, $\mathbf{d}$, with a given realisation of the noise, through the network $f : \mathbf{d} \to x$ as well as the waveform at a given f_c, $f : \mathbf{h}(f_c) \to x^{\mathbf{h}}(f_c)$. The likelihood we compute is

$$\ln\mathcal{L} = C - \frac{||x - x^{\mathbf{h}}(f_c)||^2}{2}. \qquad (6.29)$$

We calculate equations (6.28) and (6.29) for $f_c^{\rm true} = 0.1$Hz between $1 \times 10^{-3} < f_c < 0.5$Hz. The posterior and logarithm of the posterior for f_c calculated using all the data, $\mathbf{d}$, and the network summary, x, are shown in figure 12 in dashed orange and solid blue respectively, normalised to the same scale. It can be seen that the true logarithm of the posterior and logarithm of the posterior from the summaries, in the lower subplot, are not similar. However, the posterior distributions in the upper subplot are extremely similar, peaking at the true central oscillation frequency. The network has found a summary which describes the value of the parameter well. When using MOPED to find linear summaries of the data, aliasing peaks in the likelihood arise from unsuccessfully undoing the Fourier transform, shown by the green dotted line in figure 12. We use equations (1.5) and (1.6) to calculate the MOPED summary. Although the logarithm of the posterior distributions looks similar to the orange dashed curve obtained when using all the data, it can be seen in the upper subplot that aliasing peaks appear. Using the IMNN avoids the aliasing problem.

When the fiducial value is substantially wrong when training the network the logarithm of the posterior using the network summary has many aliasing-looking peaks, but the posterior only has one, Gaussian-like, peak around the true parameter value. Although the network may not have learned exactly the correct procedure to summarise the data, it is better suited to summarising the data than the MOPED algorithm. It is also probable that the network we have chosen has saturated to a function in a false maximum of the Fisher information, and by training many different networks with different initial conditions, the false maximum could be avoided and the *true* optimal summary of the network be found.

VII. CONCLUSIONS

We have shown how information maximising neural networks (IMNNs) can perform automatic physical inference. Automatic physical inference begins by training a neural network to find the optimal non-linear 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] generalises 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, firstly summarising 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 maximising neural network can replace both steps in [33] by taking the raw data and providing non-linear, 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 use 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 reionisation, τ, and recovering B-mode polarisation from probes of the large-angle cosmic microwave background polarisation 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 non-linear summaries are suited to situations where linear summaries can be misleading.

Information maximising 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, whilst 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 at `https://doi.org/10.5281/zenodo.1119070`.

Acknowledgements

This work was supported by the ANR grant number ANR-16-CE23-0002 and the Simons Foundation. Benjamin Wandelt is supported by the Labex ILP (reference ANR-10-LABX-63).

[1] J. R. Bond, A. H. Jaffe, and L. Knox, Phys. Rev. D **57**, 2117 (1998), astro-ph/9708203.

[2] M. Tegmark, A. Taylor, and A. Heavens, Astrophys. J. **480**, 22 (1997), astro-ph/9603021.

[3] A. Heavens, R. Jimenez, and O. Lahav, Mon. Not. Roy. Astron. Soc. **317**, 965 (2000), astro-ph/9911102.

[4] F. Murtagh and A. Heck, eds., *Multivariate Data Analysis*, vol. 131 of *Astrophysics and Space Science Library* (1987).

[5] P. J. Francis, P. C. Hewett, C. B. Foltz, and F. H. Chaffee, Astrophys. J. **398**, 476 (1992).

[6] A. J. Connolly, A. S. Szalay, M. A. Bershady, A. L. Kinney, and D. Calzetti, Astron. J. **110**, 1071 (1995), astro-ph/9411044.

[7] D. S. Madgwick, O. Lahav, I. K. Baldry, C. M. Baugh, J. Bland-Hawthorn, T. Bridges, R. Cannon, S. Cole, M. Colless, C. Collins, et al., Monthly Notices to the Royal Astronomical Society **333**, 133 (2002), astro-ph/0107197.

[8] O. Lahav, *Data Compression, Classification and Parameter Estimation. Methods : Examples from Astronomy* (Springer Berlin Heidelberg, Berlin, Heidelberg, 2009), pp. 73–76, ISBN 978-3-540-44767-2, URL `https://doi.org/10.1007/978-3-540-44767-2_3`.

[9] Belmon, L., Benoit-Cattin, H., Baskurt, A., and Bougeret, J.-L., Astronomy & Astrophysics **386**, 1143 (2002), URL `https://doi.org/10.1051/0004-6361:20020225`.

[10] J. Betancort-Rijo, *Structures in Random Fields* (Springer New York, 2012), pp. 397–399, ISBN 9781461219682, URL `https://books.google.fr/books?id=fjn0BwAAQBAJ`.

[11] I. Segal, *Modern Statistical Methods for Cosmological Testing* (Springer New York, 2012), pp. 67–81, ISBN 9781461219682, URL `https://books.google.fr/books?id=fjn0BwAAQBAJ`.

[12] A. Heavens, E. Sellentin, D. de Mijolla, and A. Vianello (2017), 1707.06529.

[13] C. Reichardt, R. Jimenez, and A. Heavens, Mon. Not. Roy. Astron. Soc. **327**, 849 (2001), astro-ph/0101074.

[14] A. Heavens, B. Panter, R. Jimenez, and J. Dunlop, Nature **428**, 625 EP (2004), URL `http://dx.doi.org/10.1038/nature02474`.

[15] B. Panter, R. Jimenez, A. F. Heavens, and S. Charlot, Mon. Not. Roy. Astron. Soc. **378**, 1550 (2007), astro-ph/0608531.

[16] S. Gupta and A. F. Heavens, Monthly Notices of the Royal Astronomical Society **334**, 167 (2002), URL `http://dx.doi.org/10.1046/j.1365-8711.2002.05499.x`.

[17] A. Zablocki and S. Dodelson, Phys. Rev. D **93**, 083525 (2016), URL `https://link.aps.org/doi/10.1103/PhysRevD.93.083525`.

[18] P. Protopapas, R. Jimenez, and C. Alcock, Mon. Not. Roy. Astron. Soc. **362**, 460 (2005), astro-ph/0502301.

[19] J. Alsing and B. Wandelt (2017), 1712.00012.

[20] J. Pritchard, M. Seielstad, A. Perez-Lezaun, and F. M.W., Mol Biol Evol. **16**, 1791 (1999).

[21] S. Tavaré, D. J. Balding, R. C. Griffiths, and P. Donnelly, Genetics **145**, 505âĂŞ518 (1997).

[22] C. Schafer and F. P.E., in *Statistical Challenges in Modern Astronomy V*, edited by E. Feigelson and J. Babu (Springer-Verlag New York, New York, USA, 2012), chap. 1, pp. 3–19.

[23] E. Cameron and A. N. Pettitt, Monthly Notices of the Royal Astronomical Society **425**, 44 (2012), 1202.1426.

[24] A. Weyant, C. Schafer, and W. M. Wood-Vasey, The Astrophysical Journal **764**, 116 (2013), 1206.2563.

[25] A. C. Robin, C. Reylé, J. Fliri, M. Czekaj, C. P. Robert, and A. M. M. Martins, Astronomy and Astrophysics **569**, A13 (2014), 1406.5384.

[26] C. Hahn, M. Vakili, K. Walsh, A. P. Hearin, D. W. Hogg, and D. Campbell, Mon. Not. Roy. Astron. Soc. **469**, 2791 (2017), 1607.01782.

[27] T. Kacprzak, J. Herbel, A. Amara, and A. Réfrégier (2017), 1707.07498.

[28] S. Carassou, V. de Lapparent, E. Bertin, and D. Le Borgne, Astronomy and Astrophysics **605**, A9 (2017), 1704.05559.

[29] F. B. Davies, J. F. Hennawi, A.-C. Eilers, and Z. LukiÄĞ (2017), 1703.10174.

[30] 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 (COIN), Astron. Comput. **13**, 1 (2015), 1504.06129.

[31] J. Akeret, A. Refregier, A. Amara, S. Seehars, and C. Hasner, Journal of Cosmology and Astroparticle Physics **2015**, 043 (2015), URL `http://stacks.iop.org/1475-7516/2015/i=08/a=043`.

[32] E. Jennings, R. Wolf, and M. Sako (2016), 1611.03087.

[33] J. Alsing, B. Wandelt, and S. Feeney (2018), 1801.01497.

[34] P. Graff, M. Hobson, and A. Lasenby, Mon. Not. Roy. Astron. Soc. **413**, L66 (2011), 1010.5907.

[35] R. Fisher, *Statistical Methods for Research Workers*, Biological monographs and manuals (Oliver and Boyd, The University of California, USA, 1925).

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

[37] M. Kendall and A. Stuart, *The advanced theory of statistics*, no. vol. 2 in The Advanced Theory of Statistics (Griffin, The University of California, USA, 1969).

[38] E. Lehmann and G. Casella, *Theory of Point Estimation*, Springer Texts in Statistics (Springer New York, 2003), ISBN 9780387985022.

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

[40] C. R. Rao, Bulletin of the Calcutta Mathematical Society **37**, 81âĂŞ89 (1945).

[41] Y. Bengio, Foundations and Trends in Machine Learning **2**, 1 (2009), ISSN 1935-8237.

[42] G. Cybenko, Mathematics of Control, Signals and Systems **2**, 303 (1989), ISSN 1435-568X.

[43] L. Deng and D. Yu, Foundations and Trends in Signal Processing **7**, 197 (2014), ISSN 1932-8346.

[44] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning* (MIT Press, 2016), `http://www.deeplearningbook.org`.

[45] M. A. Nielsen, *Neural Networks and Deep Learning* (Determination Press, 2015), URL `http://neuralnetworksanddeeplearning.com`.

[46] W. S. McCulloch and W. Pitts, The Bulletin of Mathematical Biophysics **5**, 115 (1943), ISSN 1522-9602.

[47] W. Pitts and W. S. McCulloch, The Bulletin of Mathematical Biophysics **9**, 127 (1947), ISSN 1522-9602.

[48] K. He, X. Zhang, S. Ren, and J. Sun, ArXiv e-prints (2015), 1502.01852.

[49] A. Krizhevsky, I. Sutskever, and G. E. Hinton, in *Advances in Neural Information Processing Systems* (2012).

[50] A. L. Maas, A. Y. Hannun, and A. Y. Ng, in *Proceedings of the International Machine Learning Society* (2013), vol. 30.

[51] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, Nature **323**, 533 (1986).

[52] K. C. Kiwiel, Mathematical Programming **90**, 1 (2001), ISSN 1436-4646, URL `https://doi.org/10.1007/PL00011414`.

[53] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, The Journal of Machine Learning Research **15**, 1929 (2014), URL `http://jmlr.org/papers/v15/srivastava14a.html`.

[54] D. B. Rubin, Ann. Statist. **12**, 1151 (1984), URL `https://doi.org/10.1214/aos/1176346785`.

[55] G. Kitagawa, Journal of Computational and Graphical Statistics **5**, 1 (1996), ISSN 10618600, URL `http://www.jstor.org/stable/1390750`.

[56] X. Glorot and Y. Bengio, in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, edited by Y. W. Teh and M. Titterington (PMLR, Chia Laguna Resort, Sardinia, Italy, 2010), vol. 9 of *Proceedings of Machine Learning Research*, pp. 249–256, URL `http://proceedings.mlr.press/v9/glorot10a.html`.

[57] S. Murray, C. Power, and A. Robotham (2013),

1306.6721.

[58] D. J. Eisenstein and W. Hu, Astrophys. J. **496**, 605 (1998), astro-ph/9709112.

[59] P. A. R. Ade et al. (Planck), Astron. Astrophys. **594**, A13 (2016), 1502.01589.

[60] Astropy Collaboration, T. P. Robitaille, E. J. Tollerud, P. Greenfield, M. Droettboom, E. Bray, T. Aldcroft, M. Davis, A. Ginsburg, A. M. Price-Whelan, et al., Astron. Astrophys. **558**, A33 (2013), 1307.6212.

[61] J. E. Bautista, S. Bailey, A. Font-Ribera, M. M. Pieri, N. G. Busca, J. Miralda-EscudÃČÂľ, N. Palanque-Delabrouille, J. Rich, K. Dawson, Y. Feng, et al., Journal of Cosmology and Astroparticle Physics **2015**, 060 (2015), URL `http://stacks.iop.org/1475-7516/2015/i=05/a=060`.

[62] J. E. Bautista et al., Astron. Astrophys. **603**, A12 (2017), 1702.00176.

[63] P. McDonald, The Astrophysical Journal **585**, 34 (2003), URL `http://stacks.iop.org/0004-637X/585/i=1/a=34`.

[64] A. Font-Ribera, P. McDonald, and J. Miralda-Escudé, Journal of Cosmology and Astrophysics **1**, 001 (2012), 1108.5606.

[65] 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, et al., Journal of Cosmology and Astrophysics **11**, 034 (2015), 1504.06656.

[66] M. S. Peeples, D. H. Weinberg, R. Dave, M. A. Fardal, and N. Katz, Mon. Not. Roy. Astron. Soc. **404**, 1281 (2010), 0910.0256.

[67] N. Suzuki, D. Tytler, D. Kirkman, J. M. OÃćÂĂÂŹ-Meara, and D. Lubin, The Astrophysical Journal **618**, 592 (2005), URL `http://stacks.iop.org/0004-637X/618/i=2/a=592`.

[68] 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 Astrophysical Journal Supplement **203**, 21 (2012), 1207.7137.

[69] F. Feroz, J. R. Gair, P. Graff, M. P. Hobson, and A. Lasenby, Class. Quant. Grav. **27**, 075010 (2010), 0911.0288.