Received: 00 Month 0000 | Revised: 00 Month 0000 | Accepted: 00 Month 0000

DOI: xxx/xxxx

ARTICLE

Parameter inference with deep jointly-informed neural networks

K. D. Humbird*[1,2] | J. L. Peterson[1] | R. G. McClarren[3]

[1]Lawrence Livermore National Laboratory, 7000 East Ave, Livermore, CA 94550
[2]Department of Nuclear Engineering, Texas A&M University, 3133 TAMU, College Station, TX 77843
[3]Department of Aerospace and Mechanical Engineering, University of Notre Dame, 365 Fitzpatrick Hall, Notre Dame, IN, 46556

Correspondence
*K. D. Humbird, Lawrence Livermore National Laboratory, 7000 East Ave, Livermore, CA 94550, Email: humbird1@llnl.gov

Summary

A common challenge in modeling inertial confinement fusion (ICF) experiments with computer simulations is that many of the simulation inputs are unknown and cannot be directly measured. Often, parameters that are measured in the experiment are used to infer the unknown inputs by solving the inverse problem: finding the set of simulation inputs that result in outputs consistent with the experimental observations. In inertial confinement fusion, this process is often referred to as a "post-shot analysis". Post-shot analyses are challenging as the inverse problem is often highly degenerate, the input parameter space is vast, and simulations are computationally expensive.

In this work, deep neural network models equipped with model uncertainty estimates are used to train inverse models, which map directly from output to input space, to find the distribution of post-shot simulations that are consistent with experimental observations. The inverse model approach is compared to Markov chain Monte Carlo (MCMC) sampling of the forward model, which maps from input to output space, for parameter inference tasks of varying complexity. The inverse models perform best when searching vast parameter spaces for post-shot simulations that are consistent with a large number of observables, where MCMC sampling can be prohibitively expensive. We demonstrate how augmenting inverse models with autoencoders enable the inclusion of several dozen observables in the inverse mapping, reducing the degeneracy of the model and improving the accuracy of the post-shot analysis.

KEYWORDS:
parameter inference, neural network, surrogate model

1 | INTRODUCTION

The goal of inertial confinement fusion (ICF) is to compress a deuterium-tritium (DT) fuel capsule to high temperature and density to ignite fusion reactions. Impinging laser beams or x-rays generated within a hohlraum ablate material on the outer surface of the capsule, creating a rocket-like force that compresses the inner DT shell. Under ideal conditions, the gas at the center of the capsule ignites a fusion burn wave that propagates outward, consuming the fuel and releasing immense amounts of fusion energy [17, 1].

Determining the requirements for ICF ignition in a laboratory setting is challenging: the ICF design space is vast and largely unexplored, and experiments are prohibitively expensive. To better understand implosion performance and explore new designs, designers rely heavily on computationally-demanding simulations. However, validating simulations with experimental data is

difficult, as many simulation inputs are not directly controlled or measured, and thus must be inferred based on experimental observations. The process of inferring the simulation inputs that are consistent with experiential measurements is called a "post-shot" analysis [14, 23].

Post-shot analyses are critical to understanding ICF experiments as many physical quantities (such as pressure, internal energy, entropy) cannot be explicitly measured, and thus must be determined from post-shot simulations. Traditionally, post-shot analyses are performed by varying a small number of simulation inputs manually until the simulation outputs fall within the experimental uncertainties for three or four important quantities of interest (QOI). This is a slow procedure and often only one set of simulation inputs is found, when in fact there could be several combinations of simulation inputs that are consistent with the experimental measurements. Drawing conclusions from a single post-shot solution can thus lead to incorrect interpretation of the experimental data.

In this work, a neural network-based approach to post-shot analysis is presented, which finds the distribution of simulation inputs that are consistent with a large number of experimental measurements. This approach builds inverse neural network models, which map directly from the observables to distributions of simulation inputs. In section 2, the deep neural network algorithm used to train the inverse models is discussed and compared to traditional Bayesian models. In section 3, the inverse model approach to parameter inference is compared to standard Markov chain Monte Carlo (MCMC) sampling [10] for simple tasks, and is tested on a challenging ICF inference task. The benefits of inverse modeling are discussed and summarized in section 4.

2 | DEEP JOINTLY-INFORMED NEURAL NETWORKS WITH DROPOUT

The simulations used to model ICF experiments are computationally expensive, thus MCMC sampling a large parameter space using simulations alone is unfeasible. To enable rapid evaluation of the simulator, "surrogate" models are trained on a fixed data base of simulations which span the parameter space of interest. The surrogate is essentially a model that interpolates between the available data, providing fast estimates of the QOI anywhere in the parameter space. Simple examples of surrogate models include linear regression or power law models, but as the data increases in complexity and dimensionality, more sophisticated machine learning algorithms are often required to accurately fit the data.

Deep neural networks have several properties that make them attractive surrogate models; they are accurate, flexible, scalable, and are easy to update as new data becomes available. However, the predictive accuracy of deep neural networks is often sensitive to the choice of the network architecture and training hyper-parameters. There are few robust guidelines for designing and training neural networks for arbitrary data sets, and hyper-parameter optimization techniques are prohibitively expensive unless the range of hyper-parameter values is highly constrained. "Deep jointly-informed neural networks" (DJINN) is an algorithm designed to overcome the challenges of determining an appropriate deep neural network architecture and weight initialization for arbitrary data sets. DJINN builds robustly accurate neural networks by mapping decision trees trained on the data into initialized neural networks, which are subsequently tuned via back-propagation [15]. DJINN is used in this work because it displays robustly accurate performance for a variety of data sets, and does not require extensive hyper-parameter tuning. DJINN is employed as an ensemble method, in which a random forest [2] of decision trees is mapped into an ensemble of neural networks.

An advantage of using deep neural network surrogates, such as DJINN, is that estimates on prediction uncertainties are readily obtained by employing a technique called dropout. Dropout is a popular regularization technique for deep neural network training; it requires randomly removing a small subset of neurons from the network each training epoch, preventing the network from over-fitting to the training data [26]. Gal and Ghahramani [9] recently demonstrated that sufficiently large neural networks with dropout approximate deep Gaussian processes [25, 8, 7]. The method for extracting uncertainty information from the neural networks requires dropout to be employed after each layer of the network during the training and evaluation stages. Evaluating the network many times samples the neural network model space, building up a distribution of predictions which approximates variational inference of a deep Gaussian process. Since DJINN is often employed as an ensemble method, the model space that gets sampled when using dropout with DJINN is larger than a single neural network with dropout; recent work has shown that such ensemble methods, while not strictly Bayesian, yield prediction uncertainties that behave similarly to Gaussian process methods[24].

To evaluate the efficacy of DJINN with dropout as an approximately Bayesian model, the performance of Bayesian DJINN (B-DJINN) is compared to several other Bayesian surrogate models: Bayesian multivariate adaptive regression splines (BMARS) [5], Bayesian additive regression trees (BART) [4], and Gaussian processes (GP) with radial basis function (RBF)

TABLE 1 Hyperparameter for B-DJINN models.

	Architecture	Learning rate	Epochs	Batch size
Logistic	4, 6, 13, 20	0.004	500	30
Boston	13, 12, 20, 23	0.004	600	10
Diabetes	10, 13, 18, 3	0.004	300	18
Yield	5, 7, 13, 23	0.002	1000	400

kernels. The optimal length scale in the RBF kernel in the GP is found by maximizing the log-marginal likelihood using the L-BFGS-B algorithm [18]. The optimal length scales for each data set are : 0.215 for logistic, 0.138 for Boston housing, 0.152 for diabetes, and 0.223 for the yield data set. Table 1 summarizes the hyperparameters for the B-DJINN models, including a representative architecture from the ensemble of five neural networks, and the training parameters. The dropout rate was set to 5% for all of the networks, and the weight regularization coefficient is set to 10^{-6}.

The models are compared on four regression tasks: the Boston housing [11] and diabetes progression [6] data sets are common test problems for regression. The logistic data set is generated by varying x and k in the logistic function and adding Gaussian distributed noise to compute $f(x, k)$:

$$f(x,k) = \frac{1}{1-\exp(-kx)} + N(\mu = 0, \sigma^2 = 0.01). \tag{1}$$

The data set contains 1000 randomly sampled points with x sampled between (-1,1) and k sampled between (0,10). The yield data set is a set of 5000 Latin hypercube sampled [16] ICF implosion simulations that span a 4D simulation input space. The implosions are simulated with the multi-physics code HYDRA [20], and the primary QOI in the database is the yield– the thermonuclear energy produced in the implosion. Each data set is split into an 80% training set and 20% test set, and the inputs are scaled between [0,1] prior to training.

To evaluate the accuracy of each model, the mean squared error (MSE), mean absolute error (MAE), and explained variance ratio (EV) are computed using the mean predictions for the test data set. These metrics do not take into account the uncertainties in the predictions, thus to compare the overall accuracy and precision of each model, the normalized sum of absolute errors is computed using the following equation:

$$\frac{1}{N_{data} \cdot N_{pred}} \sum_{i=1}^{N_{data}} \sum_{j=1}^{N_{pred}} |Y_j(x_i) - T_i|, \tag{2}$$

where N_{data} and N_{pred} are the number of data points and predictions, respectively, $Y_j(x_i)$ is the jth prediction for the data point x_i, and T_i is the true value at point x_i. This metric is large for models that are inaccurate and have wide distributions of predictions, and small for accurate models with low uncertainties. Table 2 summarizes the performance metrics for each of the regression data sets.

B-DJINN with dropout displays similar performance to other Bayesian regression algorithms, both in terms of its average accuracy and according to integrated metrics which take into account the uncertainty in the model predictions. B-DJINN offers some distinct advantages over models such as BMARS and BART – it does not required MCMC sampling and thus B-DJINN can be trained more efficiently, and it is easier to save and evaluate at a later time as it does not require the full distribution of models to be saved.

3 | INVERSE MODELS WITH B-DJINN

In complex experiments there are often input conditions to the system that are not directly controlled or measured, and must be inferred with physical models that relate the unobservable and observable quantities. Determining a set of unknown inputs which correspond to a set of observed outputs is a common task in Bayesian analysis; unknown parameters are typically inferred by MCMC sampling the "forward" surrogate model– the model that maps from input to output space– to identify sets of inputs that are consistent with observations.

Rather than MCMC sampling the forward model, we propose the use of inverse B-DJINN models, which are trained to map directly from output to input space. The inverse mapping is often degenerate; this is reflected in the uncertainty of the B-DJINN

TABLE 2 Performance metrics for B-DJINN and several Bayesian regression models for four data sets. The MSE, MAE, and EV are computed using the mean predictions for the test data sets. B-DJINN shows similar performance to the other Bayesian surrogates.

	Logistic				Boston			
	MSE	MAE	EV	NSAE	MSE	MAE	EV	NSAE
B-DJINN	3.171e-4	0.014	0.998	0.038	5.359	1.743	0.913	2.068
GP	9.023e-4	0.008	0.992	0.098	22.62	3.042	0.716	3.122
BART	1.292e-4	0.008	0.999	0.098	15.42	2.574	0.690	3.274
BMARS	2.495e-6	0.002	0.999	0.001	18.57	2.867	0.637	2.602
	Diabetes				Yield			
	MSE	MAE	EV	NSAE	MSE	MAE	EV	NSAE
B-DJINN	2935	40.82	0.405	45.82	0.002	0.034	0.998	0.065
GP	5451	60.65	0.257	60.66	0.001	0.02145	0.996	0.127
BART	3379	47.11	0.466	47.02	0.002	0.032	0.995	0.038
BMARS	2142	38.47	0.638	49.70	0.001	0.032	0.998	0.015

predictions. The degeneracy of the inverse model can be reduced by including a large number of observables in the output space–knowing more about the experimental observations aids in constraining where the experiment is located in simulation input space. Although the inverse models sometimes suffer from large uncertainties, they provide rapid estimates of the simulation inputs consistent with the observables, which can then be confirmed by evaluating the forward surrogate.

In the following subsections, the benefits of using inverse models are illustrated with two inference tasks. In section 3.1, inverse models are compared to MCMC sampling to infer the exponent k in the logistic function of Eq. 1. In section 3.2, inverse B-DJINN models are used to perform a post-shot analysis for ICF data, in which eight inputs are simultaneously inferred using several observables.

3.1 | Parameter inference with inverse models

To demonstrate the utility of training inverse models for inference tasks, MCMC sampling the forward model with B-DJINN and GP models is compared to the inverse model predictions for the logistic function data set. Given ten random values of x and the corresponding value of $f(x,k)$, the task is to infer the value of k. The forward B-DJINN and GP models are MCMC sampled using the Metropolis-Hastings algorithm [12] to find the values of k consistent with the observations of x, $f(x,k)$; the resulting distributions are shown in blue (B-DJINN) and green (GP) for various true values of k in the left side of Fig. 1 . These distributions can be compared to the inverse model predictions of k that are generated by training a B-DJINN model to map directly from $(x, f(x,k))$ to k, then evaluating the inverse model with the ten available observations and collecting the predicted values of k into a distribution. The black points on the plot indicate the true value of k.

The MCMC sampled DJINN model provides the most accurate and precise predictions for k. Although the inverse model displays high uncertainties, the mean value of its prediction is close to the true value of k away from the boundaries of the data set. The inverse model prediction is acquired in a fraction of the time required for MCMC inference, and although uncertain, the mean prediction is accurate enough to be used as a starting point for the MCMC sampling to accelerate convergence. The right side of Fig. 1 demonstrates the utility of using the inverse model prediction to warm-start sampling for k=9. The MCMC chain converges within 3000 MCMC samples when initializing the chain at the inverse model mean prediction for k; randomly initializing the chain takes over 100,000 samples to converge.

Although this is a simple inference task in which only a single parameter is unknown, the methodology is readily extended to high-dimensional inference tasks. As the number of parameters that need to be inferred increases, the advantages of using an inverse model to approximate the values of the unknown parameters become significant. MCMC sampling in high-dimensional spaces can be prohibitively expensive, and starting the chain near the true value can greatly reduce the time spent searching the space.

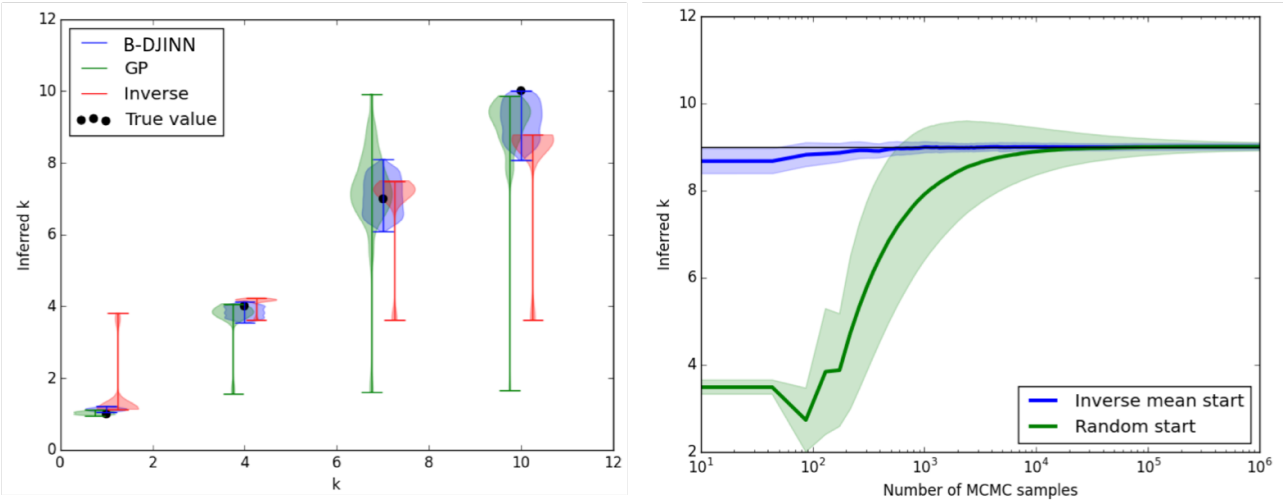

FIGURE 1 Parameter inference with inverse models and MCMC sampling forward models. Left: B-DJINN, Gaussian process (GP), and an inverse model are used to infer k given values of x, $f(x, k)$. Right: Inverse model estimates accelerate the convergence of a Markov chain by initializing the search near the true value. The shaded region indicates the 25th and 75th percentile from the distribution of inferred values of k, and the solid line indicates the median value.

3.2 | Post-shot analysis of ICF experiments with inverse models

Inverse models are next tested on a more challenging inference task: post-shot analysis of an ICF implosion [27, 3]. The database used to train the B-DJINN models contains 60,000 HYDRA simulations that are Latin hypercube sampled from an 8D input space. The inputs include several engineering features that are present in the experiment: the amplitude and width of a fill tube, a small tube used to fill the capsule with DT gas [19], and a tent– a thin membrane that holds the capsule in place within the hohlraum [22]. The inputs also include effects that could cause performance degradation in the implosion, including carbon mix of the ablator into the DT fuel, and extra energy in the fuel prior to the implosion (preheat). The final three inputs are characteristics of the experiment that are controllable– the energy scale, which is a scale factor for the laser energy and fuel capsule size, the peak multiplier, which describe the energy and power of the laser, and the amount of dopant added to the ablator.

As a demonstration, four observables commonly matched in post-shot analyses are used to infer the simulation inputs. The observables include: neutron yield (log10(Neutrons)), which is the number of fusion neutrons produced during the implosion, ion temperature (T_ion), which is inferred using the width of the neutron birth spectra [21], bang time, which is the time of peak neutron production, and the down scatter ratio (DSR), which is the ratio of lower to higher energy neutrons in the observed spectrum [13]. The observables are used to train inverse models that map from the 4D output vector to the 8D simulation input vector. The mapping is degenerate, but it serves as an illustrative example as to how unconstrained the simulation inputs are when only matching a small number of experimental observables.

The inverse models are tested on a simulation for which the true inputs and outputs are known. The four observables are Gaussian distributed about the true value with uncertainties typical for experiments; these distributions are shown in blue in Fig. 2 . The distributions of observables map into distributions of 8D input vectors via the inverse B-DJINN model. Figure 3 illustrates the resulting simulation inputs from the inverse model, shown in red; the black lines in the figure indicate the true values of the simulation inputs. The inverse model has non-negligible error in its predictions; it is therefore expected that some of the simulation inputs are not actually consistent with the known outputs. The red input distributions are passed through the forward model, resulting in the red output distributions shown in Fig. 2 . As expected, the red "inverse + forward" distributions are broader than the true output distributions due to the error in the surrogate models.

An alternative approach to inverse modeling is MCMC sampling the forward distribution to locate simulations whose outputs fall within the blue distributions indicated in Fig. 2 ; the resulting set of inputs are shown in blue in Fig. 3 . The inputs are much more constrained in the inverse mapping predictions, which is attributed to the error of the inverse surrogate. Due to the degeneracy of the problem, it is challenging to train an accurate model to predict eight inputs given values of four observables, and the model predictions trend toward the mean of the training data when relationships between the inputs and outputs are not

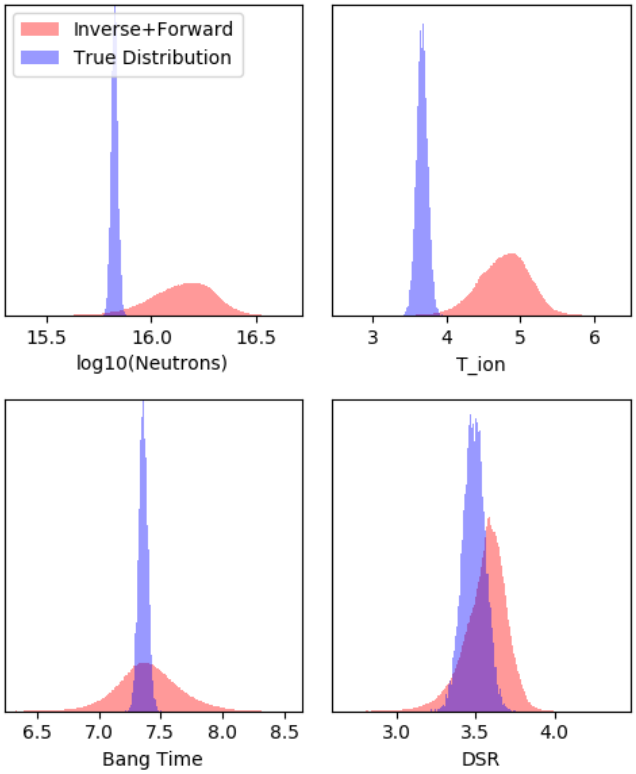

FIGURE 2 Distributions of simulation outputs from a simulation-only inference test. Blue distributions indicate the true values, red distributions are those found by first using the inverse model to map the blue distributions to simulation inputs, then using the forward model to map back to simulation outputs; error in the inverse model leads to broadening of the output distributions.

constrained. This is shown clearly in the inference of the fill tube width, which is unconstrained according to the MCMC result, but is predicted to be the mean value of the training data by the neural network.

The inverse model approach to post-shot analysis provides rapid estimates of simulation inputs that are consistent with experimental observations without requiring potentially expensive MCMC sampling of the forward model. The inverse models suffer from large uncertainties in highly degenerate problems, however they can provide a good starting location for an MCMC sampling algorithm. The utility of inverse models becomes more obvious when a large number of observables is included in the inverse mapping; if more information is known about the experiment, the distributions of inferred simulation inputs are better constrained. However, including a large number of observables in the inverse models increases the number of degrees of freedom in the neural network, and therefore increases the amount of data required to train the model. Many observables available in ICF experiments contain redundant information; X-ray images and neutron spectra are recorded from various lines of sight, temperatures are measured using various diagnostics, etc. Thus it is reasonable to expect that a collection of several dozen observables can be compressed to a smaller number of metrics that efficiently summarize the data. There are many dimensional reduction algorithms for data compression; in this work, we consider the use of autoencoder neural networks to compress a large set of observables from ICF experiments into a low dimensional "latent space".

An autoencoder is a neural network designed to non-linearly compress data with minimal information loss. Autoencoders are straightforward variations of traditional feed forward neural networks that have a characteristic hourglass-like shape. The network takes as input a large vector of data, and non-linearly compresses the data through hidden layers with a decreasing number of neurons per layer until a bottleneck layer is reached. This bottleneck layer is often referred to as the "latent space"; it is a low-dimensional representation of the original vector of data that was input to the network. The network from the input layer to the bottleneck layer is the "encoder"; it encodes the large vector of data into the low-dimensional latent space. The second half of the network decompresses the data one layer at a time until the output layer, which is the same size as the input layer. This half of the network is the "decoder"; it takes the latent space vector and decompresses it to get back the original vector of

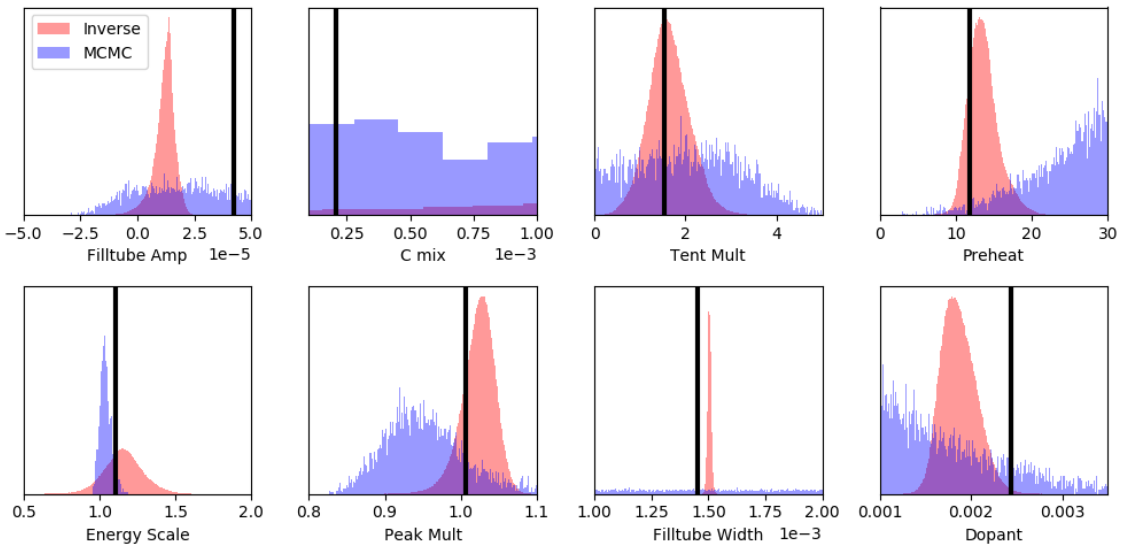

FIGURE 3 Distributions of simulation inputs. Red distributions are those found via the inverse model, blue distributions are those found by MC sampling the forward model, and accepting simulations whose outputs fall within the black dashed lines in Fig.2 . The bold black line indicates the true value of the input.

data. The autoencoder is trained by minimizing the reconstruction error between the input vector and the output vector, while forcing the data to go through the low-dimensional latent space.

Autoencoders enable the inclusion of many observables in the ICF inverse models without adding a significant number of degrees of freedom, keeping the training data demand moderate. Rather than training inverse models to map from yield, ion temperature, bang time, and DSR to the eight simulation inputs, the inverse models are now trained to map from the latent space to the inputs. The autoencoder latent space is a 10D representation of 45 scalar outputs from the simulation databases. The outputs include: neutron yield, ion temperature, DSR, and the skew and kurtosis of the neutron spectra measured from four lines of sight, X-ray emissions measured at several energies, temperature, pressure, and volume of the hot spot of the implosion, and bang time. The median reconstruction error of the autoencoder is less than 2% for all 45 observables.

To illustrate the improvements autoencoders offer for post-shot analysis, consider the same test simulation from Figures 2 -3 . Rather than using inverse models to find post-shot simulations which are consistent with only four observables, inverse models built with the autoencoder find simulations which are consistent with all 45 observables included in the latent space. Figure 4 shows the distributions of four of the observables, and Fig. 5 shows the corresponding post-shot simulation inputs.

Requiring post-shot simulations to be consistent with a set of 45 observables results in more accurate and precise predictions of the simulation inputs from the inverse models, as shown in Fig. 5 . Although there is still error accumulation from the forward and inverse models, resulting in some post-shot simulations which have observable values slightly outside the range of the true experimental uncertainties, shown in Fig. 4 , the error is significantly lower than that in Fig 2 .

The inverse modeling approach to post-shot analysis performs best when a large number of observables are used to constrain the simulation inputs; when matching only a small number of observations the mapping is highly degenerate, and MCMC sampling the forward model yields better results. Although including a large number of observables in the inverse models does not remove all of the degeneracy, it does improve the inverse model accuracy significantly. Furthermore, when matching several dozen observables MCMC sampling can be prohibitively expensive, thus inverse models augmented with autoencoders offer a promising alternative to MCMC sampling when searching high dimensional spaces for inverse solutions.

4 | CONCLUSIONS

Ensembles of deep neural networks with dropout are used as approximate Bayesian models to perform post-shot analyses for inertial confinement fusion experiments. Inverse models, which map directly from output to input space, are compared to traditional

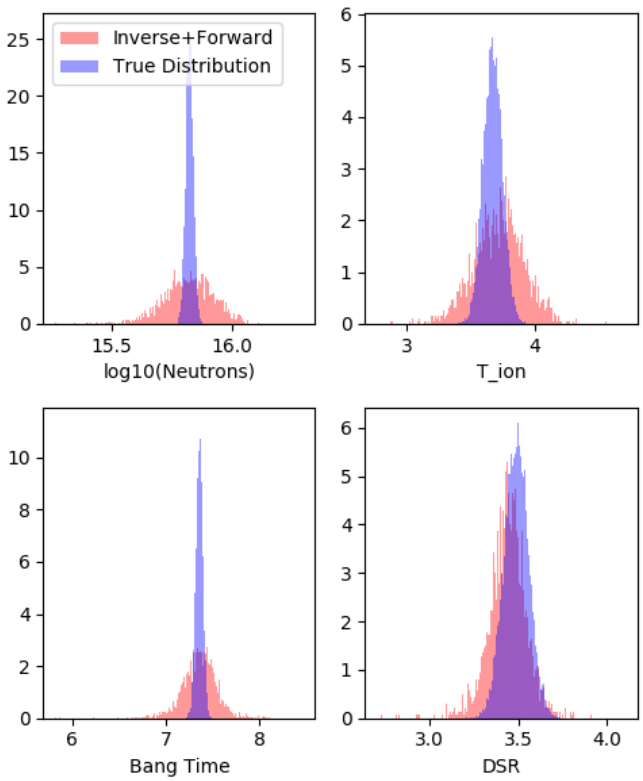

FIGURE 4 Distributions of simulation outputs from a simulation-only inference test. Blue distributions indicate the true values, red distributions are those found by first using the latent space inverse model to map the blue distributions to simulation inputs, then using the forward model to map back to the latent space, which is decoded to produce simulation outputs. The latent space post-shot analysis locates simulations which agree well with experimental observations, without requiring expensive MCMC sampling.

MCMC sampling of the forward model for post-shot analysis. The inverse models suffer from large uncertainties when constrained by only a few outputs, but the benefits of using inverse models become evident when searching for post-shot simulations which match a large number of experimental outputs.

Autoencoder neural networks are used to compress a large number of observables into a lower dimensional representation of the data, and inverse models are trained to map from this low dimensional "latent" space to the simulation inputs directly. Autoencoder augmented post-shot analyses enable the best post-shot simulations to be found rapidly, by simply evaluating a neural network multiple times. This approach is much more efficient than MCMC sampling a forward model, which can be prohibitively expensive when searching high dimensional parameter spaces for solutions that are consistent with several dozen experimental outputs.

Neural network-based post-shot analyses offer a promising approach to interpreting ICF experiments, as they provide uncertainty estimates on the inferred simulation inputs, and thus can be used to establish the relative likelihoods of certain hypotheses explaining the experimental data.

ACKNOWLEDGMENTS

The authors would like to thank Ryan Nora, Brian Spears, John Field, Jim Gaffney, and Michael Kruse for fruitful discussions, and for providing access to the ICF simulation and experimental data.

This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344. Released as LLNL-PROC-752360. This document was prepared as an account of

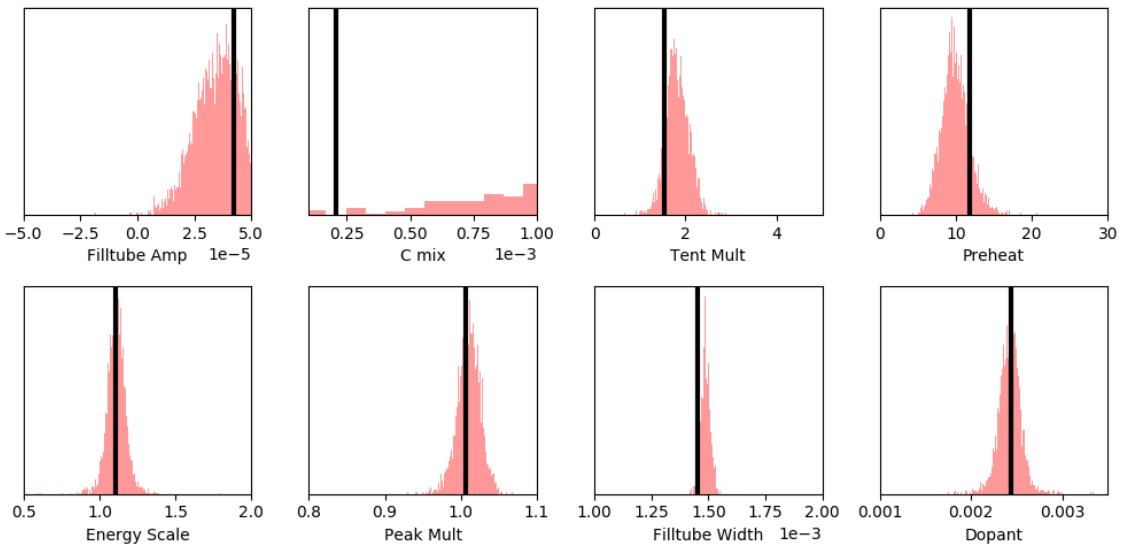

FIGURE 5 Distributions of simulation inputs. Red distributions are those found via the inverse model which maps from the latent space (composed of 45 observables compressed to 10 latent parameters) to the simulation input space. The bold black line indicates the true value of the input. The latent space post-shot analysis results in better-constrained and more accurate values for the simulation inputs which are consistent with experimental observations, and does not require expensive MCMC sampling of the forward model.

work sponsored by an agency of the United States government. Neither the United States government nor Lawrence Livermore National Security, LLC, nor any of their employees makes any warranty, expressed or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States government or Lawrence Livermore National Security, LLC. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States government or Lawrence Livermore National Security, LLC, and shall not be used for advertising or product endorsement purposes.

References

[1] Atzeni, S. and J. Meyer-ter-Vehn, 2004: *The Physics of Inertial Fusion*. Oxford Science Publications.

[2] Breiman, L., 2001: Random forests. *Mach. Learn.*, **45**, no. 1, 5–32.

[3] Casey, D. T. and et. al., 2018: The high velocity, high adiabat, "Bigfoot" campaign and tests of indirect-drive implosion scaling. *Physics of Plasmas*, **25**, no. 5, 056308, doi:10.1063/1.5019741.

[4] Chipman, H. A., E. I. George, and R. E. Mcculloch, 2006: *BART: Bayesian Additive Regression Trees*.

[5] Denison, D. G. T., B. K. Mallick, and A. F. M. Smith, 1998: Bayesian MARS. *Statistics and Computing*, **8**, no. 4, 337–346.

[6] Efron, B. and et al., 2004: Least angle regression. *Annals of Statistics*, 407–499.

[7] Gal, Y., 2016: Uncertainty in deep learning. *University of Cambridge*.

[8] Gal, Y. and Z. Ghahramani, 2015: Dropout as a Bayesian approximation: Insights and applications. *Deep Learning Workshop, ICML*.

[9] Gal, Y. and Z. Ghahramani, 2015: Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning. *ArXiv e-prints*.

[10] Gilks, W. R., S. Richardson, and D. Spiegelhalter, 1995: *Markov chain Monte Carlo in practice*. CRC press.

[11] Harrison, D. and D. Rubinfeld, 1978: Hedonic prices and the demand for clean air. *J. Environ. Economics & Management*, **5**, 81–102.

[12] Hastings, W. K., 1970: Monte carlo sampling methods using markov chains and their applications. *Biometrika*, **57**, no. 1, 97–109.

[13] Hatarik, R., D. B. Sayre, J. A. Caggiano, T. Phillips, M. J. Eckart, E. J. Bond, C. Cerjan, G. P. Grim, E. P. Hartouni, J. P. Knauer, J. M. Mcnaney, and D. H. Munro, 2015: Analysis of the neutron time-of-flight spectra from inertial confinement fusion experiments. *Journal of Applied Physics*, **118**, no. 18, 184502, doi:10.1063/1.4935455.

[14] Humbird, K., L. Peterson, R. McClarren, J. Field, J. Gaffney, M. Kruse, R. Nora, and B. Spears, 2017: Using deep neural networks to augment NIF post-shot analysis. *APS Meeting Abstracts*, YO7.013.

[15] Humbird, K. D., J. L. Peterson, and R. G. McClarren, 2018: Deep neural network initialization with decision trees. *IEEE transactions on neural networks and learning systems*.

[16] L., I. R., 2014: *Latin Hypercube Sampling*, American Cancer Society.

[17] Lindl, J., 1998: *Inertial Confinement Fusion*. Springer-Verlag.

[18] Liu, D. C. and J. Nocedal, 1989: On the limited memory bfgs method for large scale optimization. *Mathematical programming*, **45**, no. 1-3, 503–528.

[19] MacPhee, A. G. and et al., 2017: X-ray shadow imprint of hydrodynamic instabilities on the surface of inertial confinement fusion capsules by the fuel fill tube. *Phys. Rev. E*, **95**, 031204, doi:10.1103/PhysRevE.95.031204.

[20] Marinak, M., G. Kerbel, N. Gentile, O. Jones, D. Munro, S. Pollaine, T. Dittrich, and S. Haan, 2001: Three-dimensional HYDRA simulations of national ignition facility targets. *Physics of Plasmas*, **8**, no. 5, 2275–2280, doi:http://dx.doi.org/10.1063/1.1356740.

[21] Munro, D. H., 2016: Interpreting inertial fusion neutron spectra. *Nuclear Fusion*, **56**, no. 3, 036001.

[22] Nagel, S., J. Rygg, L. Benedetti, T. Ma, M. Barrios, S. Haan, B. Hammel, T. Doeppner, A. Pak, R. Tommasini, et al., 2013: The impact of capsule“tent”thickness on interpreting low mode shape. *APS Meeting Abstracts*.

[23] Nora, R., J. Field, B. Spears, and C. Thomas, 2016: Using ensembles of simulations to find high-fidelity post-shot models of inertial confinement implosions at the National Ignition Facility. *APS Meeting Abstracts*.

[24] Pearce, T., M. Zaki, A. Brintrup, and A. Neel, 2018: Uncertainty in neural networks: Bayesian ensembling. *arXiv preprint arXiv:1810.05546*.

[25] Rasmussen, C. E., 2006: Gaussian processes for machine learning. MIT Press.

[26] Srivastava, N., G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, 2014: Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, **15**, no. 1, 1929–1958.

[27] Thomas, C., 2016: BigFoot, a program to reduce risk for indirect drive laser fusion. *58th Annual Meeting of the APS Division of Plasma Physics*, **61**, 18.