Comparing theory based and higher-order reduced models for fusion simulation data

K. J. H. Law [*] , A. Litvinenko [†] , J. M. Park [‡] , M. R. Ciancosa [§] , D. L. Green [¶] & D. E. Bernholdt [‖]

October 27, 2018

Abstract

We consider fitting a theory-based log-linear ansatz, as well as higher order approximations for the thermal energy confinement of a Tokamak. We use general linear models based on total order polynomials, as well as deep neural networks. The results indicate that the theory-based model fits the data almost as well as the more sophisticated machines, within the support of the data set. The conclusion we arrive at is that only negligible improvements can be made to the theoretical model, for input data of this type.

1 Setup

Assume we have a set of labelled data points, or training data pairs $\mathcal{D} = \{x^{(i)}, y^{(i)}\}_{i=1}^N$, where $x^{(i)} \in \mathbb{R}^K$ and $y^{(i)} \in \mathbb{R}_+$.

We postulate amongst a family of ansatz $f(\cdot;\theta)$, parameterized by $\theta \in \mathbb{R}^P$, that if we can find a θ^* such that for all $i = 1, \ldots, N$,

$$y^{(i)} \approx f(x^{(i)};\theta^*)),$$

then for all x' "suitably similar" to the set $\{x^{(i)}\}_{i=1}^N$, for example at least within the convex hull and ideally in a high-probability region over training data distribution,

$$y' \approx f(x';\theta^*)\,,$$

where y' is the true output corresponding to input x'. In particular, the hope is that if we derive f based on physical theory, then in fact it will hold that even for x' different from the set $\{x^{(i)}\}_{i=1}^N$.

[*]School of Mathematics, University of Manchester, UK
[†]King Abdullah University of Science and Technology, Thuwal, Saudi Arabia
[‡]General Atomics, San Diego, CA, USA
[§]Oak Ridge National Laboratory, Oak Ridge, TN, USA
[¶]Oak Ridge National Laboratory, Oak Ridge, TN, USA
[‖]Oak Ridge National Laboratory, Oak Ridge, TN, USA

Here we consider parametric approximations, given by generalized linear models (GLM), in particular polynomial models, as well as nonlinear models (NM) given by neural networks (NN).

We will consider the problem of finding the θ^* which minimizes data misfit

$$\Phi(\theta) = \sum_{i=1}^{N} \ell(y^{(i)}, f(x^{(i)}; \theta)) \,,$$

for some distance function ℓ. Here we use the standard choice $\ell(\cdot) := |\cdot|^2 := \|\cdot\|_2^2$. It makes sense to interpret this as the log-likelihood of the data given the parameter, as a function of the parameter.

2 Physical setup and theory-based model

In recent work [8, 2], a theory-based scaling of thermal energy confinement time has been derived based on a comprehensive turbulent transport model TGLF [11] in core coupled to the EPED [10] edge pedestal model, especially in burning plasma conditions with dominant fusion alpha particle heating for future reactor design. The simulation dataset consists of a massive number of predictive IPS-FASTRAN [7] simulations, self-consistent with core transport, edge pedestal, fusion alpha particle heating, and MHD equilibrium, built upon a modern integrated modeling framework, Integrated Plasma Simulator (IPS).

For input parameters $x^{(i)} \in \mathbb{R}^K$, $i = 1, \ldots, N$, representing various physical quantities, the theoretical relationship with the output thermal energy confinement time $y^{(i)} \in \mathbb{R}_+$, is the following ansatz

$$f_{\text{th}}(x; \theta) = \theta_0 \prod_{k=1}^{K} x_k^{\theta_k} \,, \tag{2.1}$$

or on a logarithmic scale (redefining $\log \theta_0 \to \theta_0$)

$$\log f_{\text{th}}(x; \theta) = \theta_0 + \sum_{k=1}^{K} \theta_k \log(x_k) \,. \tag{2.2}$$

3 Parametric regression

Parametric models afford the luxury of discarding the data set after training, and reduce subsequent evaluations of the model to a cost relating to the number of parameters only. However, these models are much more rigid than the non-parametric ones, by virtue of restricting to a particular parametric family.

3.1 General linear models

The simplest type of parametrization is in terms of coefficients of expansion in some basis $\{\psi_i\}_{i=0}^{P-1}$. Here we choose appropriately ordered monomials with total degree $\leq p$, then $P = (K+p)!/K!p!$. The polynomials $\{\psi_i\}$ are products of monomials in the individual variables, such

as $\psi_i(x) = \prod_{k=1}^K \psi^{\alpha_k(i)}(x_k)$, where $\psi_0(x) = 1$, each index $i \in \mathbb{Z}_+$ is associated to a unique multi-index $\alpha(i) \in \mathbb{Z}_+^K$, and the single variable monomials are defined as $\psi^{\alpha_k(i)}(z) = z^{\alpha_k(i)}$ for $z \in \mathbb{R}$. In particular, for $i = 1, \dots, K+1$, $\alpha(i) = \delta_i$ and $\psi_i(x) = x_i$. The polynomials are constructed with sglib (https://github.com/ezander/sglib) and the computations are done with Matlab.

Assume that $f(\cdot;\theta) = \sum_{i=0}^{P-1} \theta_i \psi_i(\cdot)$. Under the assumption that the data is corrupted by additive Gaussian white noise, the negative log-likelihood (i.e. data misfit) takes the form

$$\Phi(\theta) = \sum_{i=1}^N |f(x^{(i)};\theta) - y^{(i)}|^2 .$$

Let $X = [x^{(1)}, \dots, x^{(N)}]^T$ and $Y = [y^{(1)}, \dots, y^{(N)}]^T$, and let $\Psi(X)_{ij} = \psi_j(x^{(i)})$. Then

$$\Phi(\theta) = |\Psi(X)\theta - Y|^2 ,$$

and the maximum likelihood solution is given by

$$\theta^* = \left(\Psi(X)^T \Psi(X)\right)^{-1} \Psi(X)^T Y .$$

Predictions of outputs Y' for some X' are later given by

$$Y' = \Psi(X')\theta^* .$$

We will also consider log-polynomial regression, where the monomials are composed with log, i.e. the basis functions become $\psi_i \circ \log$, where log acts component-wise. We then define $\log f(\cdot;\theta) = \sum_{i=0}^{P-1} \theta_i \psi_i(\log(\cdot))$ and the misfit is

$$\Phi(\theta) = \sum_{i=1}^N |\log f(x^{(i)};\theta) - \log y^{(i)}|^2 . \tag{3.1}$$

To understand the motivation for this, observe that (2.2) is a log-linear model corresponding to the logarithm of (2.1). Therefore, this setup includes the theoretical model prediction, when $p = 1$, and provides a way to systematically improve upon the theoretical model, by increasing p.

As a final note, as p increases, there is a chance of including more irrelevant parameters, in addition to ones which may be relevant. Therefore, we must regularize somehow. For the GLM, it is natural to adopt a standard L^2 Tikhonov regularization term $\lambda|\theta|^2$, resulting in the following modified objective function

$$\bar{\Phi}(\theta) = \Phi(\theta) + \lambda|\theta|^2 , \tag{3.2}$$

which again can be exactly solved

$$\theta^* = \left(\Psi(X)^T \Psi(X) + \lambda I\right)^{-1} \Psi(X)^T Y .$$

One can readily observe that this provides an upper bound to the pseudo inverse operator mapping the observation set to the optimal parameter (in other words of $(\Psi(X)^T \Psi(X) + \lambda I$ has a spectrum lower bounded by λ).

3.2 Nonlinear models

In some very particular cases, it may make sense to invoke nonlinear models, at the significant additional expense of solving a nonlinear least squares problem (in the easiest instance) as opposed to the linear least squares solution presented above. The most notable example is deep neural networks. It has been shown that these models perform very well for tasks such as handwritten digit classification, voice and object recognition [3], and there has recently even been mathematical justification that substantiates this [5], however [6]. show that one can sometimes find very small perturbations on input values which lead to misclassification. The methods can also be used for regression problems, and will be considered below.

3.2.1 Deep neural networks

Let $\theta_i \in \mathbb{R}^{P_i}$ ($P = \sum_{i=1}^{L} P_i$) be weights and g_i be some possibly nonlinear function, corresponding to layer i, and let

$$f(\cdot;\theta) = g_L(\cdots g_2(g_1(\cdot;\theta_1);\theta_2);\cdots;\theta_L).$$

As an example, θ_i is typically split into parameters $\theta_{i,w} \in \mathbb{R}^{n_i,n_{i-1}}, \theta_{i,b} \in \mathbb{R}^{n_i}$, defining an affine transformation $\theta_{i,w}x+\theta_{i,b}$, and $g_i(x;\theta_i) = \sigma_i(\theta_{i,w}x+\theta_{i,b})$, where σ_i is a simple nonlinear "activation function" which is applied element-wise. In this case n_i are the number of auxiliary variables, or "neurons", on level i, $n_0 = K$ is the input dimension, $n_L = 1$ is the output dimension, and there are L levels. For regression one typically takes σ_L =Id the identity map, as will be done here.

For the neural networks used here, $\sigma_i = \sigma : \mathbb{R} \to \mathbb{R}_+$, for $i < L$, is the the rectified linear unit (ReLU) defined by

$$\sigma(z_i) = \max\{0, z_i\}. \tag{3.3}$$

This activation function contains no trainable parameters, and we iterate that it is applied element-wise to yield a function on $\mathbb{R}^{n_i}$.

The misfit again takes the form

$$\Phi(\theta) = \sum_{i=1}^{N} |f(x^{(i)};\theta) - y^{(i)}|^2, \tag{3.4}$$

except now we have a nonlinear and typically non-convex optimization problem.

There are many methods that can be employed to try to find the minimum of this objective function, and they will generally be successful in finding some minimum (though perhaps not global). Due to the mathematically simple functions that make up the neural network, derivatives with respect to the unknown parameters can be determined analytically. A popular method to solve non-convex optimization problems is stochastic gradient descent, or Robbins-Monro stochastic approximation algorithm [9, 4], which is defined as follows. Let $\widehat{\nabla\Phi}$ be an unbiased estimate of $\nabla\Phi$, i.e. a random variable such that $\mathbb{E}\widehat{\nabla\Phi}(\theta) = \nabla\Phi(\theta)$. Then let

$$\theta_{i+1} = \theta_i - \gamma_i \widehat{\nabla\Phi}(\theta_i). \tag{3.5}$$

If $\{\gamma_i\}$ are chosen such that $\sum_i \gamma_i = \infty$ and $\sum_i \gamma_i^2 < \infty$, then the algorithm is guaranteed to converge to a minimizer [9, 4], which is global if the function is convex [1]. Noting that we can compute

$$\nabla\Phi(\theta) = \sum_{i=1}^{N} \nabla_\theta \ell(f(x^{(i)};\theta), y^{(i)})$$

easily in this case, one could use for example $\widehat{\nabla\Phi}(\theta) = \nabla\Phi(\theta) + \xi$, where ξ is some random variable, for example a Gaussian, with $\mathbb{E}\xi = 0$. Something more clever and natural can be done in the particular case of (3.4). Recall that N can be very large. Observe that (3.4) is, up to a multiple of N, an equally weighted average of the summands $\ell(f(x^{(i)};\theta), y^{(i)})$. Let $\{n_i\}_{i=1}^{N_{\text{sub}}}$ be a random (or deterministic) subset of $N_{\text{sub}} < N$ indices, where N_{sub} could be 1. Then the following provides an unbiased estimator which is also much cheaper than the original gradient

$$\widehat{\nabla\Phi}(\theta) := \frac{N}{N_{\text{sub}}} \sum_{i=1}^{N_{\text{sub}}} \nabla_\theta \ell(f(x^{(n_i)};\theta), y^{(n_i)}) \,.$$

Neural Network models were implemented and trained using the Mathematica Neural Network functions built on top of the MXNet framework.

Increasing the number of free parameters, by increasing the depth or width of the network, allows more flexibility and can provide a better approximation to complex input output relationships. However, as with GLM, increasing the number of degrees of freedom in the optimization problem also increases the susceptibility of the algorithm to instability from fitting noise in the data too accurately. A Tikhonov regularization could be employed again, but here we appeal to another strategy which is applicable only to DNN. In particular, the training data is randomly divided into N training samples and N_{val} validation samples. The algorithm 3.5 is run with the training data, however the loss of the validation data is computed along the way

$$L(\theta) = \sum_{i=N+1}^{N+N_{\text{val}}} |f(x^{(i)};\theta) - y^{(i)}|^2 \,,$$

and the network with the smallest loss on the validation data is retained at the end. This serves as a regularization, i.e. avoids fitting noise, since the training and validation sets are corrupted by different realizations of noise.

4 Numerical results

We consider a data set of 2800 data points with $K = 9$ input parameters and a single output. A subset of $N = 2100$ data points will be used for training, with the rest of the N_{out} data points aside for testing. General linear and log linear models with order up to 6 are considered, as well as deep neural networks.

4.1 GLM

An empirically chosen regularization of $\lambda = 1$ for $p = 2$, $\lambda = 50$ for $p = 3$, and $\lambda = 100$ for $p > 3$ is used in (3.2). Figure 1 presents the results for the theoretical log linear model, as well as the best fit based on a expansion in log polynomials of total order 6, for the out-of-sample set of testing data. It is clear already that the enriched model set brings negligible improvement in the reconstruction fidelity. Figure 2 shows some slices over the various covariates, where the other covariates are fixed at a value from the center of the data set. This gives some idea of how the reconstruction behaves, although one should bear in mind that the data set forms a manifold and not a hypercube, in $\mathbb{R}^9$, and so these slices are bound to include unobserved subsets of the parameter space. Furthermore, there may be sparse coverage even where there is data. This can be observed in Figure 3 which illustrates the single component marginal histograms over the data.

Figures 4 and 5 show the relative L^2 fidelity and the coefficient of determination R^2, defined below, for log polynomial and polynomial models, respectively, of increasing order.

The relative error $L^2_{\mathrm{rel}}(f)$ of the surrogate f is given by

$$L^2_{\mathrm{rel}}(f) := \sqrt{\frac{\sum_{i=N+1}^{N+N_{\mathrm{out}}} |f(x^{(i)};\theta^*) - y^{(i)}|^2}{\sum_{i=N+1}^{N+N_{\mathrm{out}}} |y^{(i)}|^2}}\,.$$

The coefficient of determination $\mathrm{R}^2(f)$ is given by

$$\mathrm{R}^2(f) := 1 - \frac{\sum_{i=N+1}^{N+N_{\mathrm{out}}} |f(x^{(i)};\theta^*) - y^{(i)}|^2}{\sum_{i=N+1}^{N+N_{\mathrm{out}}} |y^{(i)} - \overline{y}|^2}\,,$$

where $\overline{y} = \frac{1}{N_{\mathrm{out}}} \sum_{i=N+1}^{N+N_{\mathrm{out}}} y^{(i)}$.

Both models begin to saturate at the same level of fidelity, indicating that this is a fundamental limitation of the information in the data. Furthermore, one observes from Figure 4 that the saturation level of the fidelity occurs with only a few percentage points of improvement over the theoretically derived log-linear model.

4.2 Neural Networks

The neural network models were setup and trained using the built in neural network functions of Mathematica, which is built around the MXNet machine learning framework.

By connecting various combinations of layers, the neural network model gains the flexibility to model the behavior of the training data. As the size of the network increases so do the number of unknown parameters in the model. If there is too much flexibility then the network model begins to conform to the noise in the data. In addition, due to the nature of gradient decent methods, the minima reached is a local minima dependent on the starting position. As the number of unknown parameters increases, in particular if it exceeds the amount of training data, one expects the

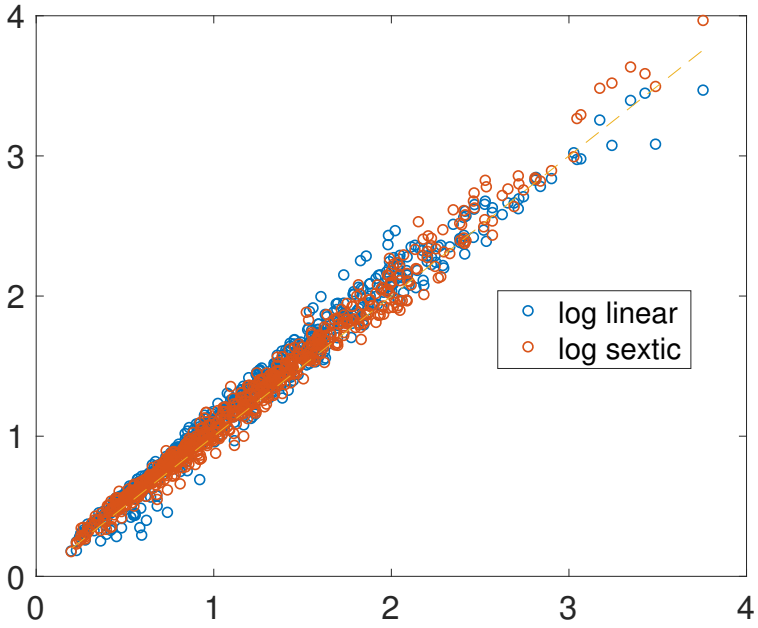

Figure 1: Comparison of true values with GLM prediction, for log linear, and log sextic. Out-of-sample, test data.

Layer Type	Inputs	Width	Weights	Biases	Total Parameters
Linear	9	20	180	20	200
ReLU	20	20	0	0	0
Linear	20	20	400	20	420
ReLU	20	20	0	0	0
Linear	20	1	20	1	21
				Total	641

Table 1: Description of the DNN

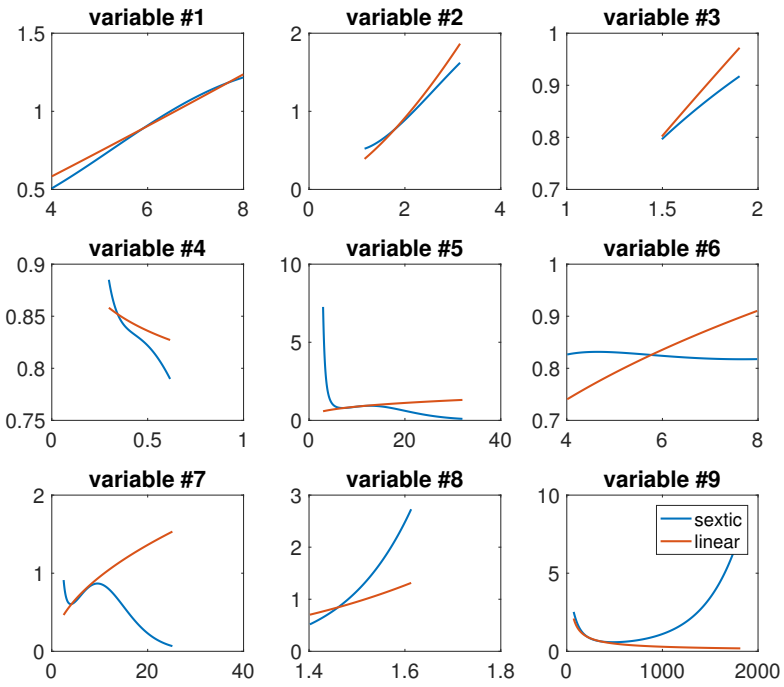

Figure 2: Comparison of GLM predictions, for log linear, and log sextic, at some random slices (the given parameter is varied over its support, while the others are fixed at a random value from the data set). Notice the correlation between big differences and small data density, by comparison to Fig. 3.

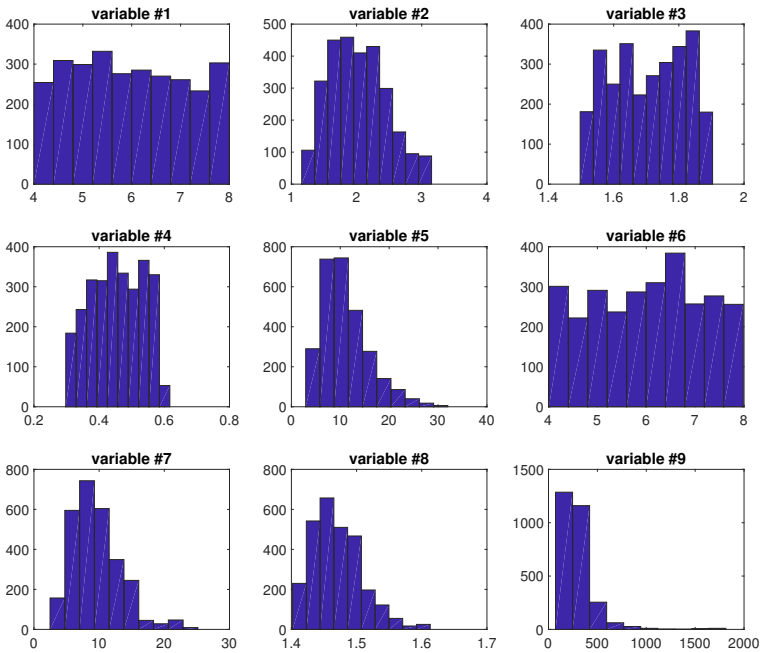

Figure 3: Input data histograms.

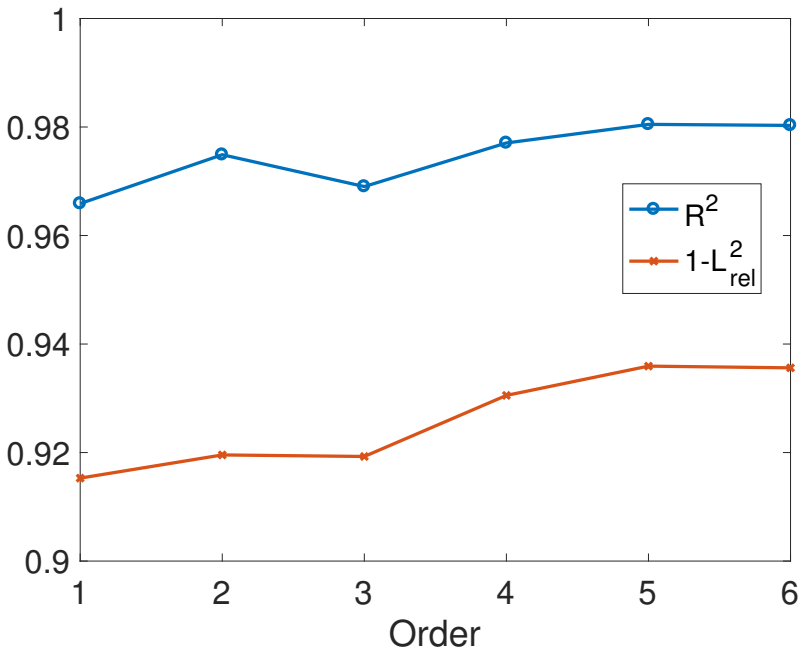

Figure 4: Relative L^2 error and coefficient of determination R^2 for out-of-sample test data as a function of model complexity, for log polynomial models.

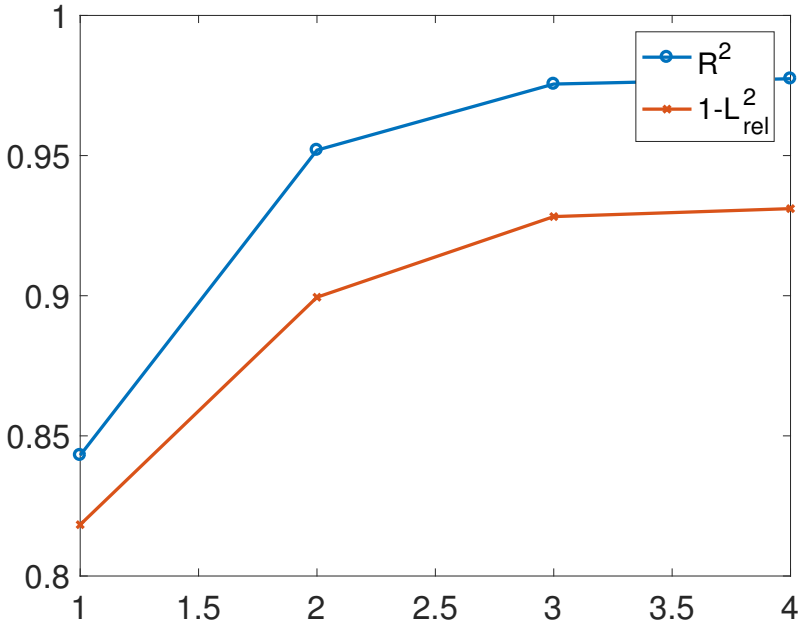

Figure 5: Same as Fig. 4 but for direct polynomial models.

Network	Total loss
3 Layer 1	0.0128
3 Layer 2	0.0138
3 Layer 3	0.0140
Linlog	0.0154

Table 2: Total loss of all available data

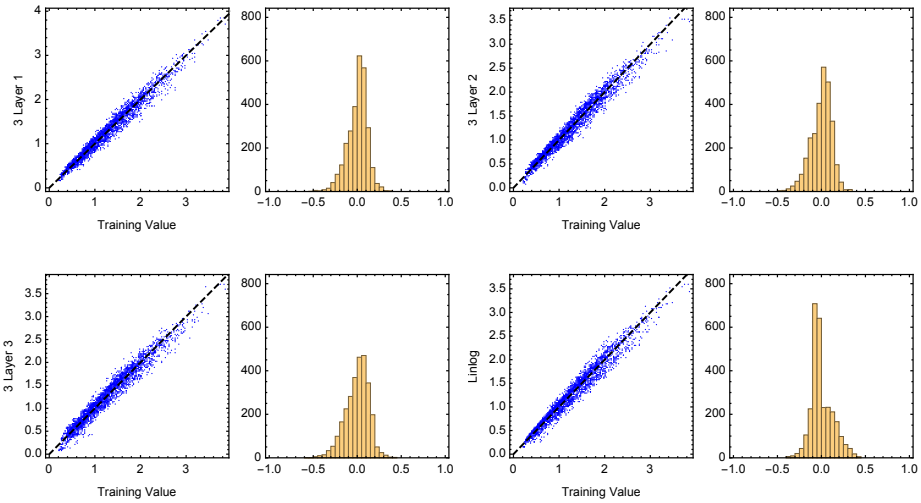

Figure 6: Histograms of the various NN models.

inversion to become unstable, with an increased risk of converging to an undesirable local minimum.

To reduce the degeneracy of the objective function, the total number tunable weights and biases in the network will be kept below N, although this can be relaxed if a Tikhonov or other type of regularization is employed. Table 1, shows the make up of the neural network model used here. Initial weights and biases and chosen at random. The networks will be trained multiple times with random initial parameters to explore various local modes. Three examples are shown in Figures 6 and 7. Figure 6 shows four pairs of images, with the left being the output in comparison to the true value, and the right being a histogram of the difference between the 2. The left two pairs and the top right are all corresponding to different instances of the same network optimization, but with different random selections of training data and different initial guesses. The bottom right is the best fit log linear model 2.2. The L^2_{rel} and R^2 fidelity measures are shown in Table 2. Figure 7 shows 9 pairs of panels, one for each parameter. The left panels are analogous to those of Figure 2, while the right panels are analogous to those of Figure 3.

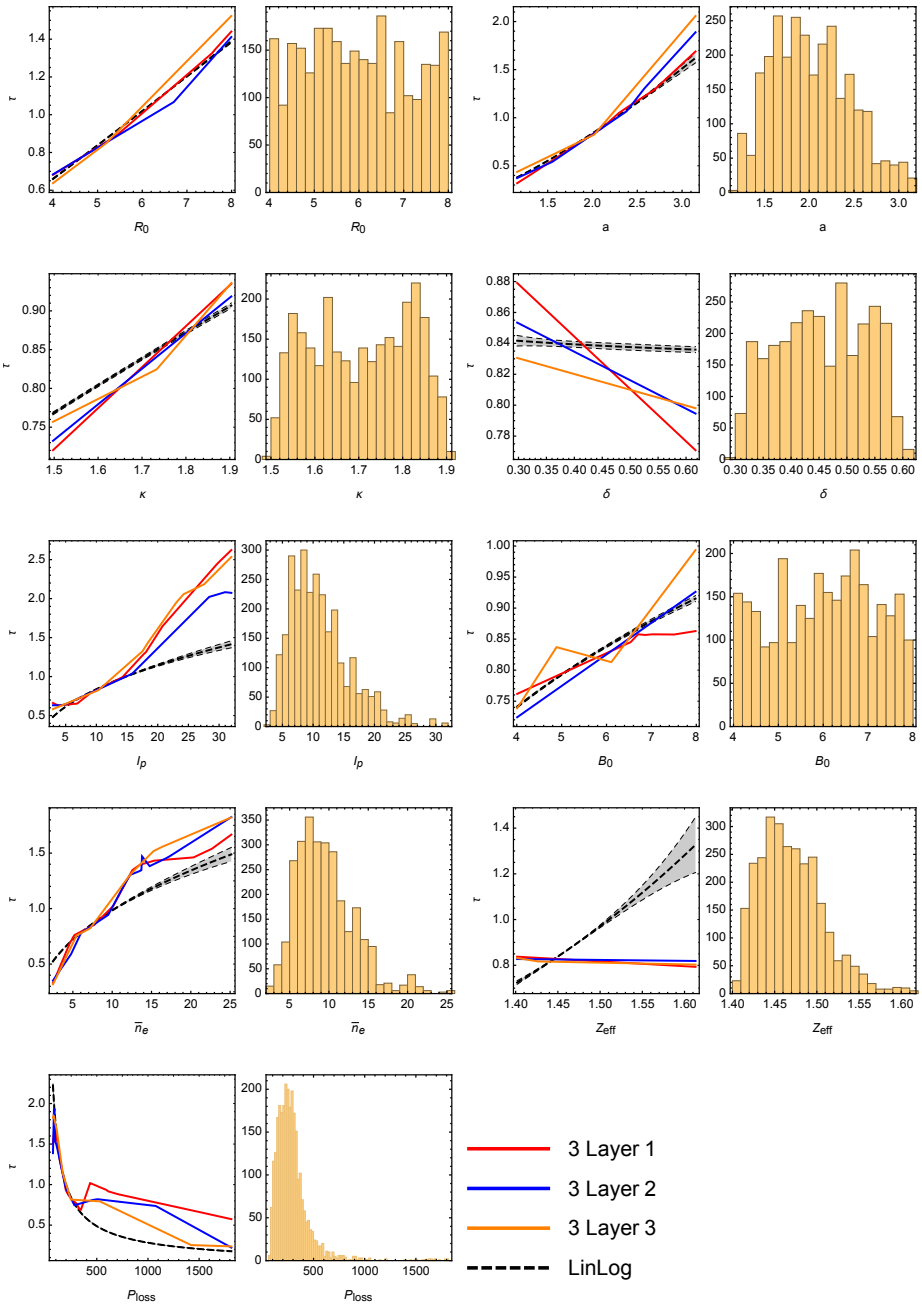

Figure 7: Scaling and data distribution of NN models.

5 Summary

In this work we considered fitting a theory-based log-linear ansatz, as well as higher order approximations for the thermal energy confinement of a Tokamak. General linear models based on total order polynomials of degree 6, as well as deep neural networks with up to 641 parameters were considered. The results indicate that the theory-based model fits the data almost as well as the more sophisticated machines, within the support of the data set. The conclusion we arrive at is that only negligible improvements can be made to the theoretical model, for input data of this type. Further work includes (i) Bayesian formulation of the inversion for uncertainty quantification (UQ), or perhaps derivative-based UQ (sensitivity), (ii) inclusion of further hyper-parameters in the GLM for improved regularization within an empirical Bayesian framework, (iii) using other types of regularization for the deep neural network, in order to allow for more parameters, and (iv) exploring alternative machine learning methods, such as Gaussian processes, which naturally include UQ.

Acknowledgements Work supported by the U.S. Department of Energy, Office of Science, Office of Fusion Energy Sciences, using the DIII-D National Fusion Facility, a DOE Office of Science user facility under awards, DE-FG02-04ER54761 and DE-FC02-04ER54698.

This manuscript has been authored by UT-Battelle, LLC under Contract No. DE-AC05-00OR22725 with the U.S. Department of Energy. The United States Government retains and the publisher, by accepting the article for publication, acknowledges that the United States Government retains a non-exclusive, paid-up, irrevocable, world-wide license to publish or reproduce the published form of this manuscript, or allow others to do so, for United States Government purposes. The Department of Energy will provide public access to these results of federally sponsored research in accordance with the DOE Public Access Plan (http://energy.gov/downloads/doe-public-access-plan).

References

[1] Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[2] JM Park etal. Theory-based scaling of energy confinement time for future reactor design. *To appear in Nuclear Fusion*, 2018.

[3] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016.

[4] Harold Kushner and G George Yin. *Stochastic approximation and recursive algorithms and applications*, volume 35. Springer Science & Business Media, 2003.

[5] Stéphane Mallat. Understanding deep convolutional networks. *Phil. Trans. R. Soc. A*, 374(2065):20150203, 2016.

[6] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. Deepfool: a simple and accurate method to fool deep neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2574–2582, 2016.

[7] Jin Myung Park, JR Ferron, Christopher T Holcomb, Richard J Buttery, Wayne M Solomon, DB Batchelor, W Elwasif, DL Green, K Kim, Orso Meneghini, et al. Integrated modeling of high βn steady state scenario on diii-d. *Physics of Plasmas*, 25(1):012506, 2018.

[8] J.M. Park, G. Staebler, P.B. Snyder, C.C. Petty, D.L. Green, and K.J. Law. Theory-based scaling of energy confinement time for future reactor design. `http://ocs.ciemat.es/EPS2018ABS/pdf/P5.1096.pdf`, 2018.

[9] Herbert Robbins and Sutton Monro. A stochastic approximation method. *The annals of mathematical statistics*, pages 400–407, 1951.

[10] PB Snyder, KH Burrell, HR Wilson, MS Chu, ME Fenstermacher, AW Leonard, RA Moyer, TH Osborne, M Umansky, WP West, et al. Stability and dynamics of the edge pedestal in the low collisionality regime: physics mechanisms for steady-state elm-free operation. *Nuclear Fusion*, 47(8):961, 2007.

[11] GM Staebler, JE Kinsey, and RE Waltz. A theory-based transport model with comprehensive physics. *Physics of Plasmas*, 14(5):055909, 2007.