FREQUENCY-SUPPORTED NEURAL NETWORKS FOR NONLINEAR DYNAMICAL SYSTEM IDENTIFICATION

A PREPRINT

Krzysztof Zając
Wrocław University of Science and Technology
krzysztof.zajac@pwr.edu.pl

Paweł Wachel
Wrocław University of Science and Technology
pawel.wachel@pwr.edu.pl

April 30, 2023

ABSTRACT

Neural networks are a very general type of model capable of learning various relationships between multiple variables. One example of such relationships, particularly interesting in practice, is the input-output relation of nonlinear systems, which has a multitude of applications. Studying models capable of estimating such relation is a broad discipline with numerous theoretical and practical results. Neural networks are very general, but multiple special cases exist, including convolutional neural networks and recurrent neural networks, which are adjusted for specific applications, which are image and sequence processing respectively. We formulate a hypothesis that adjusting general network structure by incorporating frequency information into it should result in a network specifically well suited to nonlinear system identification. Moreover, we show that it is possible to add this frequency information without the loss of generality from a theoretical perspective. We call this new structure *Frequency-Supported Neural Network* (FSNN) and empirically investigate its properties.

Keywords Neural Networks · Nonlinear Dynamics · Fourier Transform · System Identification

1 Introduction

Among different tasks of modern system modelling and identification, one can consider the problem of estimating the system's output, conditioned on its past values and excitation. In this context, one can further focus on two classes of problems: *simulation modelling* and *predictive modelling* Schoukens and Ljung [2019]. *Simulation modelling* is a task in which outputs are predicted based only on the input signal, while *prediction modelling* requires measurements of the system trajectory to predict future states (hybrid approaches are also possible).

This paper considers simulation modelling, which can be framed as an optimization problem of n-step ahead prediction based on m samples of the input sequence. The training dataset contains measurements of input and output signals from the system, which are grouped into windows of fixed length.

The number of methods developed for the identification of nonlinear systems is very large. Some of them include domain knowledge Hjalmarsson and Schoukens [2004], Schoukens et al. [2014], Ljung et al. [2004], while others are very general and require only very mild assumptions about the nature of modelled systems Śliwiński et al. [2017], Tanaka et al. [2019], Geneva and Zabaras [2020]. One of the most successful classes of models applied to the identification of nonlinear dynamics are neural networks Ribeiro et al. [2020], Andersson et al. [2019], Geneva and Zabaras [2022]. Those models are very general structures, capable of modelling many types of input-output relationships without major changes to the architecture. Nevertheless, even given the generality of the network structure, many specialized networks have been developed specifically for nonlinear dynamics and usually achieve better results than the less specialized structures Forgione and Piga [2021], Beintema and Tóth [2021]. Often those specialized networks are built using general *a priori* knowledge about the nature of the system; sometimes, they have knowledge about physical equations baked into the architecture Karniadakis et al. [2021].

One of the causes of the good performance of specialized models in specific cases is that those networks have useful inductive *a priori* knowledge about the task added during the architecture development processes. Those biases can be very general, such as translation invariance for convolutional networks or causality and memory introduced in recurrent neural networks, particularly long-short term memory network Hochreiter and Schmidhuber [1997] and gated-recurrent unit Chung et al. [2014]. Dynamical system identification, as a sequence modelling task, has a number of similarities to natural language processing. However, the inductive biases are different for both problems. In language, positional information is much more relevant than in dynamical systems modelling. In turn, for the dynamical system, it is possible to derive useful biases from physical insight. Our hypothesis is that adding frequency information to the network's structure will be useful inductive knowledge for dynamical system identification in a simulation setting. Similar approach was used to derive Fourier neural operators, which achieve good performance in fluid dynamics using predictive modelling Li et al. [2021], while still being very general and applicable to various kinds of physical or engineering systems.

This hypothesis leads to a formulation of *frequency-supported neural networks* (FSNN), which are neural structures designed for simulation modelling task [1]. They processes system input signal in parallel in time and frequency domain. Both of those branches are shown to be universal approximators, so such a structure also retains this property originally proved for feedforward network. This type of network is capable of successful identification, which is verified on two widely used benchmarks and a toy problem. Moreover, experiments show that adding this frequency structure gives the model an edge over a similar model without this additional information. FSNNs are particularly useful for input signals, which can be decomposed into a finite number of frequencies. Intuitively, this can be attributed to the fact that such a model requires only a single parameter to represent a pure-tone wave.

2 Frequency-Supported Neural Network

The novel structure of frequency-supported neural networks introduced in the paper is an extension of the multi-layered perceptron, designed to incorporate frequency information about the input signals using Fourier transform. The network is a layered structure consisting of blocks, which can be arranged in any way, as long as the correct length of the input and output is preserved. FSNN block is a building block of such a network and can be arranged into a deep network or combined with other blocks. The input of the FSNN block is a sequence, which can be a series of measurements of a dynamical system or some other time-dependent variable. Its output is also a sequence, though its length does not need to be preserved. In general, the input can be multi-dimensional (as well as the output) and transformations between dimensions are possible using this type of network.

2.1 FSNN Block

The proposed FSNN block consists of two parallel branches. One operates in the time domain and is simply a linear block processing the input sequence, whereas the other one is designed to focus on frequency space. Outputs of both branches are added together and passed through a non-linear activation function σ. Any of the commonly used functions could be used, including *Sigmoid*, *ReLU* or *Hyperbolic Tangent* Dubey et al. [2022]. Each FSNN block has four hyper-parameters: the length of the *input* sequence, T_i, the length of the *output* sequence it produces, T_o, and the input and output dimensionalities, D_i and D_o, respectively. This allows adjusting the model to different types of dynamical systems, even those with a high number of state dimensions. The output of each time-branch can be expressed by equation (1)

$$\hat{h}_l = xW_l^\mathsf{T} + b_l, \tag{1}$$

where $x \in \mathbb{R}^{T_i}$ is a row vector with length T_i, and W_l is a matrix of learnable parameters with shape $[T_o \times T_i]$, and b_l is a bias vector with length T_o. The output $\hat{h}_l \in \mathbb{R}^{T_o}$ is a row vector with length T_o. Elements of the time branch are denoted with subscript l to distinguish them from the elements of the frequency branch, denoted with subscript f. The frequency block uses Fourier transform to convert the signal to frequency space, applies learned linear transformation in this space and then converts back to the original domain. Real-valued Fourier transform is used Sorensen et al. [1987], so only the positive side of the spectrum is processed due to the assumption that such a model will process only real-valued signals. Due to the FFT algorithm used in implementation, it is most efficient on sequence lengths being a power of 2, *cf.* Brigham and Morrow [1967]. Clearly, the learned parameters are complex-valued in this branch. The equations defining this branch are given by eq. (2)

$$\hat{h}_f = \mathcal{F}^{-1}\left(\mathcal{F}(x)W_f^\mathsf{T} + b_f\right), \tag{2}$$

where $\mathcal{F}$ denotes the Fourier transform applied to the signal, and $\mathcal{F}^{-1}$ is its inverse. The matrix of parameters W_f is complex-valued with shape $[1/2\,T_o \times 1/2\,T_i]$. It is smaller than in the time branch since the length of x vector after

[1] Implementation is open-source and available at GitHub https://github.com/kzajac97/frequency-supported-neural-networks

applying the real-valued Fourier transform is half of its original length. Similarly, b_f is a complex-valued vector of parameters with length $1/2\,T_o$. The output of each FSNN block is computed as the sum of representations produced by both branches with a nonlinear activation function σ, which is expressed by eq: (3)

$$\hat{y} = \sigma(\hat{h}_f + \hat{h}_t). \tag{3}$$

The structure of the FSNN can be extended to the system with multiple-input multiple-output (MIMO) systems by extending the parameter matrices and using reshape operation. The input in the MIMO case is a matrix with shape $[T_i \times D_i]$, which is converted into a column vector of length $T_i D_i$, and the produced output is another matrix with shape $[T_o \times D_o]$. The matrices W_l and W_f are extended to shapes $[T_o D_o \times T_i D_i]$ and $[1/2\,T_o D_o \times 1/2\,T_i D_i]$, respectively. Bias vectors are also extended to contain the required parameters; $T_o D_o$ for the time branch and $1/2\,T_o D_o$ for the frequency branch. The output of the FSNN block in such a case is a vector with length $T_o D_o$, which can be rearranged into a matrix with the desired shape. The rearranging can be done before or after the aggregation and activation function because they are both element-wise operations.

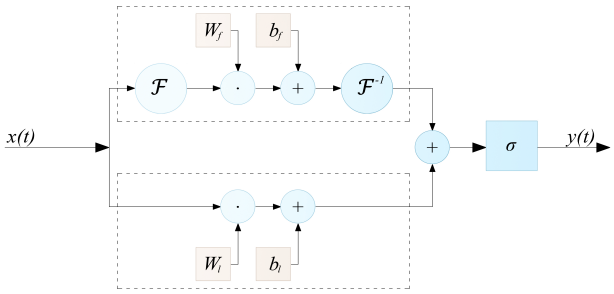

Figure 1: Schematic representation of a single FSNN block structure

2.2 N-Step Ahead Prediction

The requirement for using FSNN is the availability of a training dataset, which needs to be composed of input and output measurements of the system of interest aligned in time with constant sampling between measurements. Moreover, the model structure requires the input and output to be of the known and constant length, which can be different. To simulate a dynamical system using this approach, the input signal needs to be split into windows, for which outputs will be predicted, where each window of inputs corresponds to a single window of outputs. In practice, short output windows and long input windows are most efficient, which is intuitively clear since longer input gives the model more information, and shorter output makes error accumulation smaller. It is also possible to have different lengths in a sequential FSNN model, where the only requirement is that the output length of a given layer must match the input of the following layer. In the experiments, time windows were created using two parameters, m for input length and n for output length and time index t, which was moved along the sequence of sampling times T. During training, the overlap is possible and sometimes useful so that the same measurement can appear in different parts of the input or target sequence.

$$\{(U_{(t-m):t}, Y_{(t-n):t}) \quad | \quad t \in T\}. \tag{4}$$

Creating a training dataset as described in (4) allows formulating a training procedure as a regression task, computing the mean-squared-error between model predictions and measured targets and optimizing the parameters using stochastic gradient descent or one of its variants Ruder [2017]. U and Y in eq. (4) denote measurements of input signal and system output, respectively, which are indexed with the measurement time t, where the measurements of input signal $u(t)$ as the input to the first layer. For multi-dimensional systems, multiple vectors for excitation or output measurements can be included in the training dataset, all aligned using the same time index.

3 Theoretical Properties

Feedforward neural networks are known to have universal approximation property, which was originally proven in Cybenko [1989] and extended in particular in Hornik et al. [1989, 1990], Stinchcombe and White [1989]. Informally, the universal approximation property guarantees that for any n-dimensional function f from a given space, there exists

a feedforward neural network, $G(x)$, of the form given in (5), such that $|G(x) - f(x)| < \epsilon$ for arbitrarily small $\epsilon > 0$. In the case of simulation modelling of dynamical systems, the input to the network, $x \in \mathbb{R}^N$, is a input signal with finite time steps. To guarantee this property network the is required to have an infinite number of neurons (also called units) in the hidden layer. This representation allows showing that both time and frequency branches of FSNN block have universal approximation properties. For the time branch (*i.e.* FSNN with $\hat{h}_f \equiv 0$), this is straightforward, as the general form given in equation (5) expresses the same computation as the time branch

$$G(x) = \sigma(xW^{\mathsf{T}} + b)W_s + b_s. \tag{5}$$

The learned parameters are in the hidden layer of the network, while W_s and b_s are additional readout parameters, which are also present in the original formulation Cybenko [1989].

3.1 DFT Matrix

To show that not only the time branch of FSNN is a universal approximator but also the frequency branch (*i.e.* FSNN with $\hat{h}_l \equiv 0$), discrete Fourier transform in matrix representation needs to be used Winograd [1978], Serbes and Durak-Ata [2011]. In the derivation of FSNN-block, continuous-time Fourier transform was used, but in practical implementation the number of time steps is always finite. This allows using Fourier transform written in matrix notation, which is given in (6), where $\omega = e^{-2\pi i/N}$

$$F = \frac{1}{\sqrt{N}} \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \omega & \omega^2 & \cdots & \omega^{N-1} \\ 1 & \omega^2 & \omega^4 & \cdots & \omega^{2(N-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \omega^{N-1} & \omega^{2(N-1)} & \dots & \omega^{(N-1)(N-1)} \end{bmatrix}. \tag{6}$$

Multiplying a vector of length N by this matrix is equivalent to computing N-point discrete Fourier transform. The inverse of F corresponds to the inverse discrete Fourier transform, and its matrix form is guaranteed to exist since F is unitary Todd and Weisstein.

3.2 Frequency Branch as Universal Approximator

The frequency-branch (*i.e.* FSNN with $\hat{h}_l \equiv 0$), is a universal approximator, which can be shown based on Cybenko's proof since it is possible to write it in a way equivalent to $G(x)$ network structure, utilizing the properties of DFT matrix, given above. The general form of frequency branch in matrix notation is given by equation (7). Given that parameters can take any value, it is possible to find matrix W_f and complementing bias vector b_f, such that general form (7) is equivalent to universal feedforward network (5)

$$G_F(x) = \sigma\left(F^{-1}(FxW^{\mathsf{T}} + b)\right) W_s^{\mathsf{T}} + b_s. \tag{7}$$

Those values can be computed analytically and they are presented in equations (8). This property only holds for square matrices, since F is always square by definition. After plugging in those values, the frequency branch has all the properties of a feed-forward network and from this point of view, using Fourier transform and inverse Fourier transform is effectively a form of initialization.

$$W_f = FW^{\mathsf{T}}F^{-1} \tag{8}$$

$$b_f = bF^{-1} \tag{9}$$

4 Numerical Experiments

Three numerical experiments were run to verify the hypothesis of the FSNN model. One consists of a toy problem with a static system, while the other two were benchmarks selected from system identification literature.

Three core models were developed, for which a large grid search over the parameters was conducted. Those models were: FSNN, which is the model described in the previous section, consisting of a number of FSNN blocks stacked

together. FMLP, stands for feedforward network consisting of frequency blocks, which is a subset of the FSNN architecture, where $\hat{h}_l \equiv 0$. The final model was a regular feedforward network processing the signal using delayed input measurements, which also is a subset of FSNN with $\hat{h}_f \equiv 0$. Additionally, selected state-of-art models were re-implemented and run on the same benchmark problems, or when available, the results were transferred from original papers.

4.1 Hyperparameter Search

For all benchmarks and the three core architectures (FSNN, FMLP and MLP) random search over a defined set of hyper-parameters was run, and later full grid search was run over the subset of hyperparameters, which were important for the model. Searched parameters were the following: number of input samples and number of predicted samples, number of hidden layers, and number of units in all layers. Some hyper-parameters were frozen and used for all models, such as the optimization algorithm *Adam,* Diederik and Ba [2017], and GeLU activation function, *cf.* Hendrycks and Gimpel [2020].

The most important parameter is the number of input samples, especially for the models utilizing frequency information, since for shorter windows the amount of information about signal frequencies is lower, which makes it less useful. A smaller number of output samples effectively means the model needs to predict fewer time steps. A one-step-ahead prediction makes it usually more accurate, so the optimal value for all benchmarks was equal to one. However, using longer output windows could also be used effectively, especially when fast predictions are required by the application.

For the DynoNet model Forgione and Piga [2021], reported values are a reproduction of the model using original code on different benchmarks, with the exception of the Wiener-Hammerstein system where the value from the original paper is reported. For the State Space Encoder reported results are also values from the original work Beintema and Tóth [2021], since reproducing the model was not possible due to very long training runtimes.

4.2 Evaluation

All the above algorithms were evaluated using root mean squared error (RMSE), which is a standard method of evaluation in regression problems. Physical units are added were possible. RMSE was computed as

$$RMSE(y, \hat{y}) = \sqrt{MSE(y, \hat{y})} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y - \hat{y})^2}. \tag{10}$$

Additionally, a normalized mean squared error was also evaluated, *i.e.* the ratio of RMSE and standard deviation of the predicted value (reported as a percentage)

$$NRMSE(y, \hat{y}) = \frac{RMSE(y, \hat{y})}{\sigma_y}. \tag{11}$$

4.3 Static System with Frequency Input

A simple static affine system with a input signal consisting of pure-tone sine waves was created to test FSNN structure, under conditions most suitable for it. The input signal was generated using a sum of frequencies drawn from uniform distribution: $\alpha \sim \mathcal{U}(-5, 5)$, which is given by eq. (12)

$$u(t) = \sum_{i=1}^{5} sin(\alpha_i t). \tag{12}$$

The output of the system was generated using two additional parameters, randomly drawn from the uniform distribution $\mathcal{U}(-5, 5)$, similar to the excitation frequencies. Those are denoted as β_1, β_2 in

$$f(u) = \beta_1 u(t) + \pi\beta_2. \tag{13}$$

Results for this benchmark show the advantage of a two-branch structure over single-branch MLP and FNN models, where FSNN is able to achieve much better results, which are summarized in table 1. During the experiments, the

Model	$\#P$	RMSE	NRMSE
FNN	1610	$10.3 \cdot 10^{-3}$	0.30%
MLP	2157	$5.4 \cdot 10^{-3}$	0.16%
FSNN	887	$1.4 \cdot 10^{-3}$	0.04%
DynoNet	49	$0.3 \cdot 10^{-3}$	0.02%

$\#P$ number of parameters

Table 1: Evaluation results for selected models on test dataset for a static affine system with frequency input

best-performing models were those with a low number of parameters, but larger models were also capable of achieving satisfactory results.

DynoNet model achieving best results on this benchmark had only one static layer, without the learnable dynamical operator, which also effectively made it a feed-forward network. However, the architecture is different than MLP, and the model could be easily obtained by performing hyperparameters search on the DynoNet model. Moreover, the DynoNet model is constructed in a way allowing to easily infer the structure of the architecture given knowledge about the system since it can be decomposed into linear dynamical blocks and static nonlinearities Forgione and Piga [2021]. This allows for selecting good candidate architecture, however, such knowledge is not guaranteed in real-world situations.

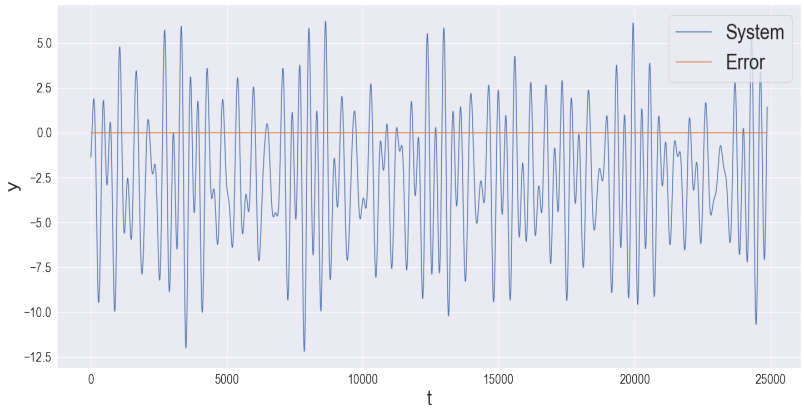

Figure 2: Simulation error computed for best FSNN model on the test dataset for a static affine system with frequency input

4.4 Wiener-Hammerstein Benchmark

Wiener-Hammerstein benchmark is a well-known benchmark problem for the identification of nonlinear dynamics. It consists of two linear blocks and a static non-linearity, which were implemented using an electronic RLC circuit with a diode, *cf.* Schoukens et al. [2009]. The measurements of this system are used to create training and test datasets for the model.

The results achieved by FSNN are not state-of-art, however, the structure performs significantly better than plain MLP network on real-world data. For FSNN this performance was improved with longer input sequences, which allowed the model to access more frequency information. Evaluation results are reported in table 2 and for selected model on figure 3. For the MLP model, good results can be achieved with a wide range of hyper-parameters. The two reported results are on the two extremes of model size with a number of parameters different by three orders of magnitude, but have very similar performances.

Model	$\#P$	RMSE	NRMSE
FNN	7856	1.9 mV	0.78%
DynoNet	63	1.2 mV	0.50%
MLP	1193	1.1 mV	0.46%
Large MLP	1379841	0.9 mV	0.38%
FSNN	1591	0.5 mV	0.22%
State-Space Encoder	21410	0.2 mV	0.10%

$\#P$ number of parameters

Table 2: Evaluation results for selected models on test dataset for Wiener-Hammerstein benchmark compared to selected results reported in literature

Both FSNN and MLP models are capable of achieving lower simulation error than DynoNet model while having substantially fewer assumptions about the nature of modelled data. Additionally, it is worth noting that all listed models have normalized simulation error lower than 1%, which would be sufficient for most practical situations.

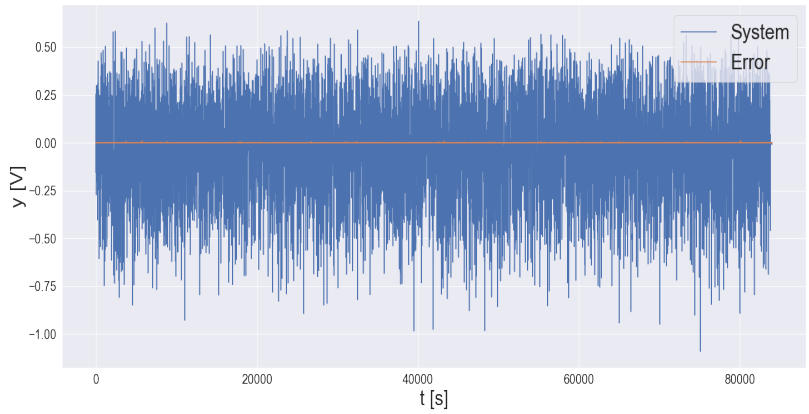

Figure 3: Simulation error computed for best FSNN model on the test dataset for Wiener-Hammerstein benchmark

4.5 Silverbox Benchmark

Silverbox benchmark is an electronic implementation of the Duffing oscillator, which can be modelled using a second-order LTI system with a polynomial nonlinearity in the feedback Wigren and Schoukens [2013]. This type of system is challenging due to its nonlinearity. Experiments performed using this benchmark were conducted in the same way as all the others.

The structure of models applied to this benchmark cannot reflect the polynomial nonlinearity, which causes larger simulation errors when compared to the Wiener-Hammerstein benchmark. Moreover, models performing well on this benchmark tend to be larger then in previous cases, which also could be attributed to this nonlinearity. Results are reported in table 4.5.

5 Discussion

Concluding, the presented architecture is capable of successfully modelling static, linear and nonlinear dynamics with almost no assumptions about the nature of the data it is trained on. From the theoretical point of view, it can be also interpreted as an initialization scheme for feedforward neural networks. The following conclusions can be drawn from our work:

Model	$\#P$	RMSE	NRMSE
FNN	14192	4.1 mV	7.69%
MLP	37313	3.9 mV	7.32%
DynoNet	81	2.9 mV	5.39%
FSNN	69719	2.3 mV	4.31%
State-Space Encoder	19930	1.4 mV	2.60%

$\#P$ number of parameters

Table 3: Evaluation results for selected models on test dataset for Silverbox benchmark compared to selected results reported in literature

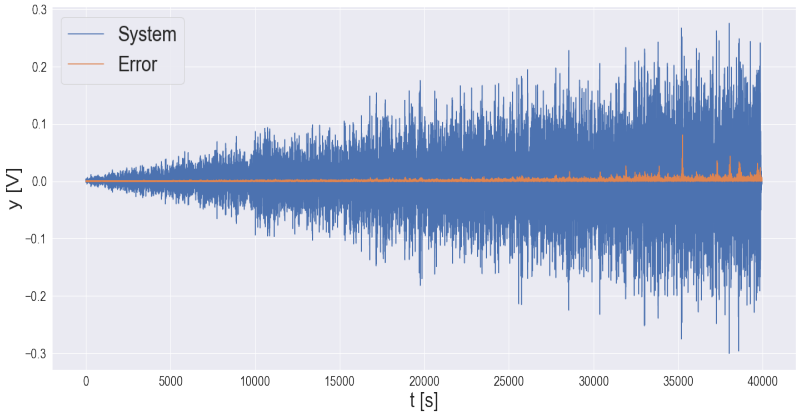

Figure 4: Simulation error computed for best FSNN model on the test dataset for Siverbox benchmark

- Frequency information is a useful feature for feedforward neural networks modelling of nonlinear dynamical systems. It is, however, most successful when used together with a plain feedforward network with a delay line in a branched structure.
- There is no easy-to-see dependency between the number of trained parameters in feedforward models and their performance on benchmarks consisting of engineering systems. In our experiments, smaller models tend to perform marginally better, which could be attributed to a greater number of updates they can make to their parameters in a limited training time.
- FSNN structure is capable of achieving good results on a number of benchmarks, and it shows relatively low sensitivity to changes in hyperparameters, which makes it potentially a good candidate for practical applications in which running large hyperparameters searches for the model is often impossible.
- Adding orthogonal transforms to neural networks is a potentially interesting area of research, where those transforms could be used as initialisation or regularization methods, using *a priori* knowledge about the functional basis particularly useful for a given problem. For example, in applications involving audio processing, the frequency domain is potentially useful for trainable models.

References

J. Schoukens and L. Ljung. Nonlinear system identification: A user-oriented road map. *IEEE Control Systems Magazine*, 39(6):28–99, 2019.

H. Hjalmarsson and J. Schoukens. On direct identification of physical parameters in non-linear models. *IFAC Proceedings Volumes*, 37(13):375–380, 2004. ISSN 1474-6670. doi:https://doi.org/10.1016/S1474-6670(17)31252-1. URL `https://www.sciencedirect.com/science/article/pii/S1474667017312521`.

M. Schoukens, R. Pintelon, and Y. Rolain. Identification of wiener–hammerstein systems by a nonparametric separation of the best linear approximation. *Automatica*, 50(2):628–634, 2014. ISSN 0005-

1098. doi:https://doi.org/10.1016/j.automatica.2013.12.027. URL https://www.sciencedirect.com/science/article/pii/S0005109813005864.

L. Ljung, Q. Zhang, P. Lindskog, and A. Juditski. Estimation of grey box and black box models for non-linear circuit data. *IFAC Proceedings Volumes*, 37(13):399–404, 2004. ISSN 1474-6670. doi:https://doi.org/10.1016/S1474-6670(17)31256-9. URL https://www.sciencedirect.com/science/article/pii/S1474667017312569.

P. Śliwiński, A. Marconato, P. Wachel, and G. Birpoutsoukis. Non-linear system modelling based on constrained volterra series estimates. *IET Control Theory & Applications*, 11(15):2623–2629, 2017. doi:https://doi.org/10.1049/iet-cta.2016.1360. URL https://ietresearch.onlinelibrary.wiley.com/doi/abs/10.1049/iet-cta.2016.1360.

G. Tanaka, T. Yamane, J. Benoit Héroux, R. Nakane, N Kanazawa, S. Takeda, H. Numata, D. Nakano, and A. Hirose. Recent advances in physical reservoir computing: A review. *Neural Networks*, 115:100–123, 2019. ISSN 0893-6080. doi:https://doi.org/10.1016/j.neunet.2019.03.005. URL https://www.sciencedirect.com/science/article/pii/S0893608019300784.

N. Geneva and N. Zabaras. Modeling the dynamics of pde systems with physics-constrained deep auto-regressive networks. *Journal of Computational Physics*, 403:109056, 2020. ISSN 0021-9991. doi:https://doi.org/10.1016/j.jcp.2019.109056. URL https://www.sciencedirect.com/science/article/pii/S0021999119307612.

A.H Ribeiro, K. Tiels, L.A. Aguirre, and T.B Schön. Beyond exploding and vanishing gradients: analysing rnn training using attractors and smoothness. *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, 108:2370–2380, 2020.

C. Andersson, A.H. Ribeiro, K. Tiels, N. Wahlström, and T.B. Schön. Deep convolutional networks in system identification. In *2019 IEEE 58th Conference on Decision and Control (CDC)*, pages 3670–3676, 2019. doi:10.1109/CDC40024.2019.9030219.

N. Geneva and N. Zabaras. Transformers for modeling physical systems. *Neural Networks*, 146:272–289, 2022. doi:10.1016/j.neunet.2021.11.022. URL https://doi.org/10.1016%2Fj.neunet.2021.11.022.

M. Forgione and D. Piga. dynoNet: a neural network architecture for learning dynamical systems. *International Journal of Adaptive Control and Signal Processing*, 35(4):612–626, 2021.

G. Beintema and M. Tóth, R. Schoukens. Nonlinear state-space identification using deep encoder networks. *Proceedings of Learning for Dynamics and Control*, 144:241–250, 2021.

G.E. Karniadakis, I. G. Kevrekidis, L Lu, P. Perdikaris, S. Wang, and L. Yang. Physics-informed machine learning. *Nature Reviews Physics*, 3:422–440, 2021.

S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

J. Chung, C. Gulcehre, K. Cho, and Y. Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. In *NIPS 2014 Workshop on Deep Learning, December 2014*, 2014.

Z. Li, N. Kovachki, K. Azizzadenesheli, B. Liu, K. Bhattacharya, A. Stuart, and A. Anandkumar. Fourier neural operator for parametric partial differential equations. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=c8P9NQVtmnO.

S.R. Dubey, S.K. Singh, and B.B Chaudhuri. Activation functions in deep learning: A comprehensive survey and benchmark. *Neurocomputing*, 503:92–108, 2022. ISSN 0925-2312. doi:https://doi.org/10.1016/j.neucom.2022.06.111. URL https://www.sciencedirect.com/science/article/pii/S0925231222008426.

H. Sorensen, D. Jones, M. Heideman, and C. Burrus. Real-valued fast fourier transform algorithms. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 35(6):849–863, 1987. doi:10.1109/TASSP.1987.1165220.

E. O. Brigham and R. E. Morrow. The fast fourier transform. *IEEE Spectrum*, 4(12):63–70, 1967. doi:10.1109/MSPEC.1967.5217220.

S. Ruder. An overview of gradient descent optimization algorithms, 2017.

G. Cybenko. Approximation by superpositions of a sigmoidal function. *Math. Control Signal Systems*, 2:303–313, 1989. doi:https://doi.org/10.1007/BF02551274.

K. Hornik, M. Stinchcombe, and H. White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366, 1989. ISSN 0893-6080. doi:https://doi.org/10.1016/0893-6080(89)90020-8. URL https://www.sciencedirect.com/science/article/pii/0893608089900208.

K. Hornik, M. Stinchcombe, and H. White. Universal approximation of an unknown mapping and its derivatives using multilayer feedforward networks. *Neural Networks*, 3(5):551–560, 1990. ISSN 0893-6080. doi:https://doi.org/10.1016/0893-6080(90)90005-6. URL https://www.sciencedirect.com/science/article/pii/0893608090900056.

M. Stinchcombe and H. White. Universal approximation using feedforward networks with non-sigmoid hidden layer activation functions. In *International 1989 Joint Conference on Neural Networks*, pages 613–617 vol.1, 1989. doi:10.1109/IJCNN.1989.118640.

S. Winograd. On computing the discrete fourier transform. *Mathematics of Computation*, 32(141):175–199, 1978. ISSN 00255718, 10886842. URL http://www.jstor.org/stable/2006266.

A. Serbes and L. Durak-Ata. The discrete fractional fourier transform based on the dft matrix. *Signal Processing*, 91(3):571–581, 2011. ISSN 0165-1684. doi:https://doi.org/10.1016/j.sigpro.2010.05.007. URL https://www.sciencedirect.com/science/article/pii/S0165168410002094.

R. Todd and E. Weisstein. Unitary matrix. URL https://mathworld.wolfram.com/UnitaryMatrix.html.

P. K. Diederik and J. Ba. Adam: A method for stochastic optimization, 2017.

D. Hendrycks and K. Gimpel. Gaussian error linear units (gelus), 2020.

J. Schoukens, J. Suykens, and L. Ljung. Wiener-hammerstein benchmark. *15th IFAC Symposium on System Identification (SYSID 2009)*, 2009. URL https://www.nonlinearbenchmark.org/benchmarks/wiener-hammerstein#h.2wdiw8u9jr39.

T. Wigren and J. Schoukens. Three free data sets for development and benchmarking in nonlinear system identification. *European Control Conference (ECC)*, pages 2933–2938, 2013. doi:https://doi.org/10.1007/BF02551274. URL https://www.nonlinearbenchmark.org/benchmarks/silverbox.