LEON: LEARNING EXPLICIT OPERATOR NETWORKS FOR NONLINEAR DYNAMICAL SYSTEMS

HUA-LIANG WEI

School of Electrical and Electronic Engineering, University of Sheffield, Mappin Street, Sheffield, S1 3JD, UK
E-MAIL: w.hualiang@sheffield.ac.uk

Abstract:

The identification of complex nonlinear systems often requires a trade-off between the accuracy, complexity, transparency, and interpretability. This paper introduces LEON (Learning Explicit Operator Networks), a new sparse and transparent framework for multivariate dynamical systems modelling. Anchored in the foundational control and system theory, LEON assumes that hidden state dynamics can be reconstructed from a finite window of past inputs and outputs, provided observability and reachability conditions are met. Unlike traditional deep learning architectures that utilise opaque activation functions which show opaque output behavour, LEON employs a specified library of polynomial basis functions which are transparent. This approach allows to identify explicit, physically meaningful operators, such as nonlinear gains and cross-product interactions directly from data. Through a sparse identification process, LEON prunes non-essential terms to yield a mathematically grounded, glass-box representation. Experimental results demonstrate that LEON achieves predictive accuracy that is competitive with state-of-the-art LSTMs and RNNs while utilising several orders of magnitude fewer parameters. By providing an explicit mathematical operator, LEON enables formal verification, offering a robust alternative for safety-critical applications.

Keywords:

Interpretable Machine learning; Model explainability; Nonlinear system identification; NARMAX; Neural networks; Sparse learning; Deep learning

1. Introduction

The identification and modelling of nonlinear dynamical systems are central to modern engineering and sciences. As systems grow in complexity, ranging from autonomous vehicle dynamics and high-precision industrial processes to complex neuroscience and space science, the ability to accurately predict future states from historical data becomes a critical requirement for monitoring, diagnosis, control, optimisation, and safety verification. Traditionally, the field of data-driven dynamic modelling has been dominated by two distinct approaches: classical nonlinear system identification (NSI) based on rigorous mathematical structures and modern machine learning based on high-dimensional neural architectures.

In the mid-1980s, the landmark work of Leontaritis and Billings [1],[2] provided a unified framework for discrete-time nonlinear systems through the Nonlinear AutoRegressive Moving Average with eXogenous inputs (NARMAX) model. By establishing that a wide class of nonlinear systems could be represented by a mapping of past inputs and outputs, provided that mild conditions of local observability and reachability are met; they provided a theoretical bridge from hidden state-space dynamics to observable input-output relationships. However, the practical challenge of selecting the correct functional form (the *structure selection* problem) suggests that there is significant potential to further enhance the scaling of these models for high-dimensional or highly non-smooth systems.

With the advent of Deep Learning, architectures such as Long Short-Term Memory (LSTM) networks and Gated Recurrent Units (GRUs) have achieved remarkable success in modeling temporal dependencies [3]-[7]. By using massive datasets and millions of trainable parameters, these models act as universal approximators for complex sequences. However, this performance comes at a significant cost: opacity. Standard neural networks rely on non-physical activation functions like ReLU or Sigmoid, resulting in a *black-box* model that offers no structural insight into the underlying physical laws. In safety-critical applications, such as aerospace or medicine, the inability to explain or formally verify a model's internal logic remains a major barrier to adoption.

To address these challenges, this paper proposes LEON (Learning Explicit Operator Networks). LEON is an adaptive, sparse identification framework that merges the theoretical rigor of NARMAX with the learning efficiency of modern neural architectures. Rather than utilising generic hidden layers, LEON employs a transparent network of polynomial basis functions to construct an explicit operator of the system.

The primary contributions of this work are three-fold:

- *Theoretical Alignment*. We demonstrate how the LEON architecture satisfies the observability and reachability conditions required for robust system identification.
- *Structural Transparency*. We show that by using polynomial operators, the identified model remains physically interpretable, allowing for a direct mapping between learned weights and system behaviors.
- *Parameter Efficiency*. We provide evidence that LEON can match the predictive accuracy of state-of-the-art deep learning models while requiring a fraction of the parameters, significantly reducing the computational footprint for real-time deployment.

The remainder of this paper is organised as follows. Section II reviews the mathematical foundations of nonlinear operators; Section III details the LEON methodology and the sparse identification process; Section IV presents experimental results and comparisons with existing architectures; and Section IV concludes with a discussion on the impact for explainable AI.

2. Related Work

The evolution of nonlinear system identification (NSI) has transitioned from rigid parametric models to flexible, yet opaque, machine learning architectures. This section reviews the trajectory of these developments and identifies the structural gaps that the LEON framework aims to bridge.

2.1. Classical foundations and the NARMAX legacy

The formalisation of discrete-time nonlinear identification was significantly advanced by Leontaritis and Billings [1], [2], who introduced the Nonlinear AutoRegressive Moving Average with eXogenous inputs (NARMAX) model. Their work established that under mild conditions of local observability and reachability, a wide class of nonlinear systems could be represented by a functional expansion of lagged inputs and outputs. Subsequent research focused on structure selection algorithms, such as the Orthogonal Least Squares (OLS) method [8]-[10], to identify significant polynomial terms from a candidate dictionary. While theoretically robust, these classical methods often struggled with the curse of dimensionality when faced with high-order nonlinearities or large-scale multivariable datasets. To mitigate these challenges, other types of basis functions including radial basis functions and wavelet functions have been introduced to the NARMAX models [11]-[14].

2.2. The black-box era: RNN and LSTMs

With the explosion of computational power, researchers pivoted toward connectionist models. Recurrent Neural Networks (RNNs) and their derivatives, specifically Long Short-Term Memory (LSTM) networks [3] and Gated Recurrent Units (GRUs), became the standard for modelling temporal dependencies. These models utilise internal *gates* to maintain long-term memory, allowing them to outperform classical NARMAX models in capturing long-range non-stationary dynamics. However, as noted by Ljung et al. [15], the lack of physical interpretability in these *black-box* models remains a significant drawback. Because their internal state transitions are governed by non-physical activation functions (e.g., Tanh or ReLU), they offer no explicit mathematical operator that can be easily analysed by model users.

2.3. Sparse identification and physics-informed learning

Recently, a new paradigm has emerged that seeks to combine the flexibility of machine learning with the interpretability of classical physics. Brunton et al. [16] introduced the SINDy (Sparse Identification of Nonlinear Dynamics) algorithm, which uses LASSO-based regression [17],[18] to identify governing differential equations from data. Simultaneously, Physics-Informed Neural Networks (PINNs) have gained traction by embedding physical constraints directly into the loss function [19].

While these methods are powerful, they often require prior knowledge of the system's governing equations or rely on continuous-time derivatives that are sensitive to measurement noise. There remains a critical need for a model that operates in the discrete-time operator domain, utilises the learning efficiency of neural networks, but maintains a sparse, polynomial-based transparency. The LEON framework addresses this gap by implementing an explicit operator network that satisfies NARMAX conditions while achieving the parameter efficiency required for modern complex system identification.

3. Methodology

The LEON (Learning Explicit Operator Networks) framework is designed to map a high-dimensional, hidden state-space into a transparent, discrete-time input-output operator. This section details the mathematical architecture of the LEON network and the sparse identification strategy used to achieve model interpretability.

3.1 Problem formulation and data lagging

The starting point for LEON is the assumption that the target system can be described by a nonlinear mapping of its historical data. Following the conditions proposed in [1],[2], we consider a general multi-input, single-output case and construct an information vector $\boldsymbol{\varphi}(t)$ (t =1, …, N) that serves as the input to the network. This vector contains the lagged values of both the system outputs and inputs as follows:

$$\boldsymbol{\varphi}(t) = [y(t-1), \quad \dots, \quad y(t-p), \; u_1(t-d), \dots, \; u_1(t-q), \dots, u_r(t-d), \dots, u_r(t-q)] \tag{1}$$

where y and $u_j\,(j = 1,..,r)$ are system output and inputs, p and q are the maximum lags determined by the system's observability and reachability requirements, and d is the minimum time delay between input inputs and the inputs and output (usually $d = 1$ or 0). By explicitly using these lags, LEON ensures that the internal dynamics of the complex are captured within the observable data window.

For convenience of description, in what follows we set $d = 1$ and use $x_1(t)$, …, $x_n(t)$, with $n = p + rq$, to represent the lagged variables involved in (1).

3.2 The polynomial operator

Unlike standard neural networks that pass inputs through a hidden layer of sigmoidal neurons, LEON utilizes a polynomial expansion scheme. This layer transforms the linear input vector $\boldsymbol{\varphi}(k)$ into a high-dimensional library of basis functions, $\mathcal{D}$, which includes:

- *All 1st degree (linear) terms,* such as x_1, x_2,…, x_n. Denote by $\mathcal{D}_1$ the set consisting of all these linear terms.
- *All 2nd degree self-product and cross-product terms,* such as x_2^2, x_1x_2. Denote by $\mathcal{D}_2$ the set consisting of all these interaction terms.
- *All 3rd degree self-product and cross-product terms,* such as x_1^3, $x_2x_3x_5$. Denote by $\mathcal{D}_3$ the set consisting of all these terms.
- *All ℓth degree self-product and cross-product terms.*

We denote by $\mathcal{D}_j$ ($1 \le j \le \ell$) the set consisting of all the ℓth degree terms. It is recommended that polynomial basis functions of up to *4th* (*quartic*) degree be used in practice as higher degree basis functions become sensitive to noise and coefficient precision, resulting in less robust models.

The LEON operator $\boldsymbol{\mathcal{L}}$ for a nonlinear difference equation is defined as:

$$y(t) = \boldsymbol{\mathcal{L}}[\varphi(t)] + e(t) \tag{2}$$

To satisfy the *glass-box* requirement, we decompose $\boldsymbol{\mathcal{L}}$ into a linear combination of explicit polynomial basis functions. This allows users to see the exact degree of nonlinearity and the interaction between lags. The expectation of the actual system output is then approximated as the weighted sum of these polynomial basis functions as:

$$E[y(t)] = \boldsymbol{\mathcal{L}}[\varphi(t)] = \textstyle\sum_{i=1}^{M} \theta_i z_i(\varphi(t)) \tag{3}$$

where each z_i is an *explicit basis function* included in the *full library* $\mathcal{D}$, θ_i is the *structure coefficient*, and M is the total number of candidate terms in the library $\mathcal{D}$.

3.3 The parsimonious structure selectors

Assume that we use polynomial basis functions of up to 3rd degree to build model (3), then the total number of candidate terms is $M = (n+1)(n+2)(n+3)/6$. Note that for most real problems, the full model (3) is a redundant representation which usually over fits data. To highlight this, we introduce a switch *selection indicator* $\gamma_i \in \{0,1\}$, to indicate whether a basis function is important ($\gamma_i = 1$) or should be removed from the full model ($\gamma_i = 0$):

$$E[y(t)] = \textstyle\sum_{i=1}^{M} \gamma_i[\theta_i z_i(\varphi(t))] \tag{4}$$

This is a typical best model selection problem which is usually represented as an L_0-norm regularised optimisation problem:

$$\min_{||\boldsymbol{\theta}||_0 \le m} \frac{1}{2} ||\boldsymbol{y} - Z\boldsymbol{\theta}||_2^2 \tag{5}$$

where Z is an $N \times M$ matrix whose columns are formed by the M basis function $z_1(\varphi(t))$,…, $z_M(\varphi(t))$, with t =1,…, N, and $\boldsymbol{\theta}$ is the regression coefficient vector. The constraint $||\boldsymbol{\theta}||_0 \le m$ means that we want to find the s best basis functions from the library $\mathcal{D}$ that best explain the target signal $\boldsymbol{y}$ in the least squares sense.

To prevent overfitting and determine the most compact and high-fidelity models, LEON employs an ensemble strategy that allows to fully leverage the advantages of several sparse learning algorithms including the iterative Orthogonal Forward Regression (iOFR) [20] and Sparse Bayesian Learning (SBL) [21]. The iOFR algorithm starts with an empty model subset, with all the M selection indicators being zero, that is, $\boldsymbol{\Gamma}_0 = \boldsymbol{0}$. It iteratively searches and determines best or most important basis functions step by step, and update the selection indicator set accordingly, resulting in $|\boldsymbol{\Gamma}_i| = i$, that is, the number of non-zero elements in the original full model is i ($1 \le i \le m \ll M$). Unlike iOFR, SBL starts with a full over-parameterised model by including all potential basis functions, and iteratively prunes irrelevant ones to achieve sparsity. SBL utilises a hierarchical prior to automatically determine which weights should be non-zero.

3.4 A graphical illustration

The implementation procedure of LEON comprises two stages (as shown in Figure 1): firstly, it uses the determined feature subset consisting of m ($\ll M$) basis functions to build a *glass-box* model (or a set of models), which is (are) transparent, interpretable, parsimonious and simulatable (TIPS) [22]; secondly, the selected s features may be used as inputs to train new networks to enhance the predictions obtained in the first stage. It is worth mentioning that previous experiences (see e.g. [23]) showed that this second stage may not always be necessary because for many practical modelling tasks models obtained in stage one can usually provide satisfactory performance. However, the second stage can be used as an auxiliary validator for the first stage-model to help verify the first-stage prediction performance in case that the first stage-model does not sufficiently capture the underlying dynamics or nonlinearities of interest. In this way, the model still maintains the useful interpretability achieved in stage one and meanwhile achieves higher prediction through stage two.

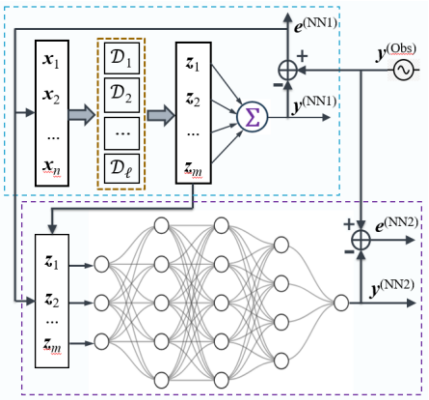

FIGURE 1. A diagram of the proposed network model.

4. Application in space weather

This section provides an illustration example showing how the proposed LEON approach is applied to space weather modelling and forecasting.

4.1 Background and overview

Reliably modeling Earth's radiation belt electron fluxes is of escalating importance, driven by our growing reliance on space-based technology. These regions are highly dynamic, full of energetic particles, primarily relativistic electrons in the outer belt, whose fluxes can fluctuate by orders of magnitude within hours in response to solar wind and geomagnetic storms. These volatile, high-energy electron fluxes pose a severe threat to satellites in Medium Earth (MEO) and Geostationary (GEO) orbits. Accurate, timely predictions are vital for space weather forecasting to protect critical communications, navigation, and security assets. However, while data-driven methods offer high accuracy, their *black-box* nature often lacks physical interpretability. Prioritizing transparent modeling ensures users can understand how predictions are made, allowing for better uncertainty quantification and the integration of data-driven insights with physics-based models. This synergy is essential for achieving the high-fidelity forecasting required to safeguard our space infrastructure.

4.2 System input variables and data

Four input variables, namely solar wind velocity, proton density, pressure and a southward interplanetary magnetic field are used as inputs/drivers to build models for the Earth's radiation belt electron flux prediction, measured by GOES-16 MPS-HI. Table 1 presents brief descriptions of these drivers.

TABLE 1. Variables used in the Earth's radiation belt electron flux modelling and forecasting

Variable	Description	
Output	$\log_{10}$(Flux)[a]	Electron fluxes unit: $(cm^2 \cdot s \cdot sr \cdot eV)^{-1}$
Inputs (Drivers)	V	Solar wind velocity (km/s)
	n	Solar wind proton density (cm^{-3})
	p	Solar wind pressure (nPa)
	B_s	Southward interplanetary magnetic field (nT)

a. Due to the massively high magnitudes of electron flux values, a log-like transform, such as $\log_{10}$(flux) is traditionally used. In this study, we follow the practice to use log10(flux) as the model output.

Three datasets were used as follows: The first dataset, consisting of 640 daily samples (01/04/2021–31/12/2022), was used for model training; the second dataset, consisting of 181 samples (01/01/2023–30/06/2023) was used for validation; and the third, consisting of 184 daily data of the period 01/08/2023–31/01/2024, was used to test the model performance. The electron flux (output), together with the four solar wind drivers, are shown in Figure 2.

4.3 Experimental settings

Studies show that the electron fluxes reach largest magnitudes about 1 to 2 days after the appearance of high solar wind streams (see e.g. [24],[25]). We therefore chose the maximum lag for all the five input and output variables as 3, resulting a total of 15 lagged variables for model construction.

For LEON network construction, the library $\mathcal{D}_3$ was as the model building blocks, i.e., we used polynomial basis functions of up to 3rd degree to build models. This resulted in a total of 816 candidate basis functions.

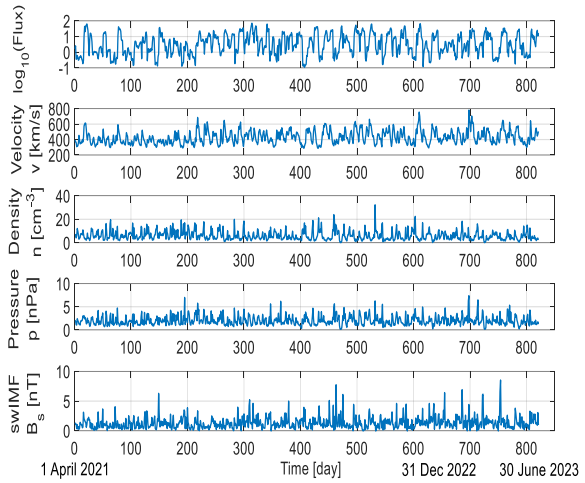

FIGURE 2. The plots of variables used for model training in dataset 1 consisting of 640 samples (01/04/2021–31/12/2022) and validation in dataset 2 consisting of 181 samples (01/01/2023–30/06/2023).

4.4 Results

4.4.1 Results produced by LEON

The sparse structure detection algorithms determined six model terms out of the 816 candidate building blocks, which are shown in Table 2, where the model should be read as:

$$\log_{10}\text{Flux}(t) = 0.6796 \times \log_{10}\text{Flux}(t-1) + 9.933 \times 10^{-5} \times V(t-1) \times p(t-2) - \cdots \quad (6)$$

In Table 2, EER (error energy reduction) is an index indicting how much percent of the energy of the target response signal is explained by the corresponding model term - here signal energy is defined as the sum of squared elements of a sample of interest. From the table it can be observed that solar wind velocity (V) is one of the most critical drivers for accelerating electrons to high energies within Earth's outer radiation belt. High-speed solar wind streams, often originating from solar coronal holes, are well-documented to correlate with significantly enhanced relativistic electron fluxes.

An illustration of the predicted log-flux on the test dataset (01/08/2023-30/06/2023) is shown in Figure 3. It is worth mentioning that data normalisation or standardisation is not an essential step for training LEON models. With and without these pre-processing operations will not change the resulting model structure; the only difference is that an extra constant term may occur if data normalisation or standardization is performed.

TABLE 2. The identified important basis functions by LEON

Model Term	Weight	p-value	EER (%)
$\log_{10}$Flux(t-1)	0.6796	7.4719e-137	74.1539
V(t-1)p(t-2)	9.933e-05	0.00022106	3.9911
n(t-2)	−0.0466	4.8421e-53	3.6362
V(t-2)	0.0018	1.9162e-12	1.4291
V(t-1)	−0.0012	5.9235e-07	0.6763
Bs(t-1)	0.0554	9.8643e-07	0.4670

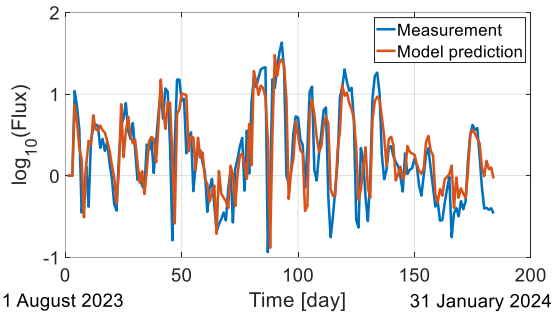

FIGURE 3. A comparison between the predicted and measured log-flux on the test dataset (01/08/2023-31/01/2024).

4.4.2 Comparison with other methods

To evaluate the superiority of the LEON framework, we compare its performance with two state-of-the-art deep learning methods: LST (Long Short-Term Memory) and GRU (Gated Recurrent Unit) network models.

The same training, validation and test data used in LEON model building were used to LSTM and GRU models. For both methods, values of the 15 lagged model input were standardised as follows: $x_{new} = (x - m_{train}) \,/\, \sigma_{train}$, where x is a model input variable, m_{train_data} and σ_{train} are the mean and standard deviation of the input variable calculated based on the training data.

The following three metrics are used to evaluate model performance: RMSE (root mean squared error), MAE (mean absolute error), R^2 (coefficient of determination) and Corr (Correlation coefficient between predictions and measurements). The values of the three metrics for LSTM, GRU and LEON, calculated on the test dataset (184 daily data of the period 01/08/2023 – 31/01/2024), are shown in Table 3.

5. Conclusions

In this paper, we introduced LEON (Learning Explicit Operator Networks), a novel framework for the identification of complex dynamical systems. By anchoring the architecture

in the fundamental NARMAX principles established by Billings and co-workers, we have demonstrated that it is possible to bridge the gap between high-performance neural learning and classical system theory.

TABLE 3. RMSE, R2 and Corr for LSTM, GRU and LEON methods

	RMSE	**MAE**	**R^2**	**Corr**
LSTM	0.3576	0.2721	0.6341	0.8005
GRU	0.5962	0.4812	0.2541	0.5157
LEON	0.3449	0.2568	0.6447	0.8031

The LEON model successfully replaces traditional *black-box* activation functions with a structured, sparse library of polynomial basis functions. This transition allows for the discovery of explicit governing equations that are mathematically transparent and physically interpretable. Our results confirm that LEON can identify complex nonlinear behaviors, with a level of clarity that is unattainable by standard deep learning models like LSTMs or GRUs. Ultimately, LEON offers a robust path forward for researchers who require the predictive power of machine learning without sacrificing the rigors of explainability and interpretability.

Acknowledgements

This work was supported in part by STFC (Ref. ST-Y001524-1), NERC (Ref. NE/W005875/1, Ref. NE/V001787/1, Ref. NE/Y503290/1 and Ref. UKRI1339) and EPSRC (Ref. EP/H00453X/1 and Ref. EP/I011056/1).

References

[1] I.J. Leontaritis and S. A. Billings, “Input-output parametric models for non-linear systems part I: deterministic non-linear systems,” Int. J. Control, vol. 41, no. 2, pp. 303-328, Feb 1985.

[2] I.J. Leontaritis and S.A. Billings, “Input-output parametric models for non-linear systems part II: stochastic non-linear systems,” Int. J. Control, vol. 41, no. 2, pp. 329-344, Feb 1985.

[3] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” Neural Computation, vol. 9, no. 8, pp. 1735-1780, Nov 1997.

[4] X. Jin, J. Shao, X. Zhang, W. An, and R. Malekian, “Modeling of nonlinear system based on deep learning framework,” Nonlinear Dyn., vol. 84, no. 3, pp. 1327-1340, May 2016.

[5] J. Gonzalez and W. Yu, “ Non-linear system modeling using LSTM neural networks,” IFAC-PapersOnLine, vol. 51, no. 13, pp. 485-489, Jan 2018.

[6] T. Simpson et al. “Machine learning approach to model order reduction of nonlinear systems via autoencoder and LSTM networks,” Journal of Engineering Mechanics, vol. 147, no. 10, art no. 04021061, Oct 2021.

[7] A. Rehmer and A. Kroll, “On using gated recurrent units for nonlinear system identification,” in Proc. 18th European Control Conference (ECC), Naples, Italy, Jun 2019, pp. 2504-2509.

[8] S. Chen, C.F. Cowan, and P.M., “Orthogonal least squares learning algorithm for radial basis function networks,” IEEE Trans. Neural Netw., vol. 2, no. 2, pp. 302-309, Mar 1991.

[9] S. A. Billings, Nonlinear System Identification: NARMAX Methods in the Time, Frequency, and Spatio-Temporal Domains. Hoboken, NJ, USA: John Wiley & Sons, 2013.

[10] H.-L. Wei et al., “Term and variable selection for non-linear system identification,” Int. J. Control, vol. 77, no. 1, pp. 86-110, Jan 2004.

[11] S. Chen et al., “Practical identification of NARMAX models using radial basis functions,” Int. J. Control, vol. 52, no. 6, pp. 1327-50, Dec1990.

[12] S. A. Billings et al., “Generalized multiscale radial basis function networks,” Neural Networks, vol. 20, no. 10, pp. 1081-94, Dec 2007.

[13] S. A. Billings and H.-L. Wei, “A new class of wavelet networks for nonlinear system identification,” IEEE Trans. Neural. Netw., vol. 16, no. 4, pp. 862-874, Jul 2005.

[14] S. A. Billings and H.-L., “The wavelet-NARMAX representation: A hybrid model structure combining polynomial models with multiresolution wavelet decompositions,” Int. J. Syst. Sci., vol. vol. 36, no. 3, pp. 137-152, Feb 2005.

[15] L. Ljung, “Perspectives on system identification,” Annual Reviews in Control, vol. 34, no. 1, pp. 1-2, April 2010.

[16] S. L. Brunton et al. “Discovering governing equations from data by sparse identification of nonlinear dynamical systems,” Proc. Natl. Acad. Sci. U.S.A., vol. 113, no. 15, pp. 3932-3937, April 2016.

[17] R. Tibshirani, “Regression shrinkage and selection via the LASSO,” J. Roy. Stat. Soc., Ser. B (Methodol.), vol. 58, no. 1, pp. 267–288, Jan 1996.

[18] T. Hastie, R. Tibshirani, and J. Friedman, The Elements of Statistical Learning, 2nd ed. New York: Springer, 2009.

[19] J. Stiasny et al., “Physics-informed neural networks for non-linear system identification for power system dynamics,” in IEEE Madrid PowerTech, Madrid, Spain, 28 Jun – 2 July 2021, pp. 1-6.

[20] Y. Guo et al. "An iterative orthogonal forward regression algorithm,"Int. J. Syst. Sci., vol. 46, no. 5, pp. 776-789, April 2015.

[21] M. E. Tipping, “Sparse Bayesian learning and the relevance vector machine,” J. Mach. Learn. Res., vol. 1, pp. 211–244, June 2001.

[22] H.-L. Wei and S. A. Billinsg, "Modelling COVID-19 pandemic dynamics using transparent, interpretable, parsimonious and simulatable (TIPS) machine learning models: A case study from systems thinking and system identification perspectives," in Recent Advances in AI-enabled Automated Medical Diagnosis, CRC Press, 2022, pp. 13-28.

[23] H.-L. Wei, “SAFE-IML: Sparsity-Aware Feature Extraction for Interpretable Machine Learning with two-stage neural network modelling,” in Proc. 10th International Conference on Machine Learning Technologies (ICMLT), Helsinki, Finland, May 2025, pp. 188-194.

[24] V. B. Belakhovsky et al., “Relativistic electron flux growth during storm and non-storm periods as observed by ARASE and GOES satellites,” Earth, Planets and Space, vol. 75, no. 1, art. 189, pp. 1-17, Dec 2023.

[25] H.-L. Wei et al., “Forecasting relativistic electron flux using dynamic multiple regression models,” Annales Geophysicae, vol. 29, no. 2, pp. 415-420, Feb 2011.