Learning Model of Generator from Terminal Data

Nikolay Stulov*, Dejan J Sobajic†, Yury Maximov‡*, Deepjyoti Deka‡, and Michael Chertkov§*
* Skolkovo Institute of Science and Technology, Moscow, Russia
† Grid Consulting LLC, San Jose, CA, USA
‡ Theoretical Division and Center for Nonlinear Studies, Los Alamos National Laboratory, Los Alamos, NM, USA
§ Program in Applied Mathematics, University of Arizona, Tucson, AZ, USA

***Abstract*—Assuming that a generator is monitored by the system operator via a PMU device positioned at the generator's terminal bus, we pose and resolve the question of the real-time, data-driven and automatic monitoring of the generator's performance. We establish regimes of optimal performance for four complementary techniques ranging from the computationally light (a) Vector Auto-Regressive Model, suitable for normal, linear or almost linear regime, via (b) Long-Short-Term-Memory and (c) Neural ODE Deep Learning models, appropriate to monitor mildly nonlinear regimes, and finally to the (d) physics-informed model. For example, the physics-informed model is capable of fast identification of nonlinear transients and providing interpretable results, suitable, in particular, for corrective actions. The conclusions are reached in the result of validating the models on synthetic data generated in a realistic setting from an open-source, state-of-the-art modeling software. Advanced analysis is followed by a summary and conclusion suitable for the next step - validation of the hierarchy of the suggested data-driven schemes in the industry setting.**

***Index Terms*—Dynamics, Machine Learning, Neural Networks, Physics Informed Machine Learning, Power Generator Model, Regression, Transients**

I. Introduction

Power Generator (PG) is a complex device that delivers energy to the power system. System operator needs to maintain an up-to-date model of all generators in the system, for multiple real-time energy management tasks run thousands of times a day. The generator model is supposed to be known from the calibration tests (normally updated once a season). However, the subject to many, and often rather significant changes taken place in between the scheduled tests. Some of the changes are from wear and tear, some are due to changes in the environment, and some are caused by malfunctions. Looking into the future grid, which will be undoubtedly subject to many more dramatic changes than what is the status quo of today, we see a need in identifying and updating the generator model often, e.g., for multiple energy management tasks run by system operators frequently, many in real-time.

PG is connected to the rest of the power system through the terminal bus. It is becoming a standard in modern power systems to equip each significant generator with a Phasor Measurement Unit (PMU) device placed at the terminal bus of the generator. PMU provides a stream of data on active and reactive powers, P and Q, voltage and phase, V and φ - all measured locally at the terminal bus.

In this manuscript we pose and resolve the challenge of developing *an hierarchy of data driven reduced models* which allows fast, real-time adaption for computationally light and more accurate assessment of generator models for use by system operators. Parameters of the reduced models are learned from the $\{P, Q, V, \varphi\}$ data stream assumed available from the PMU positioned at the generator's terminal bus.

The challenge is resolved by building properly reduced Machine Learning (ML) models of the generator of two principally different flavors – Physics Informed and/vs Dynamical System based. The Physics-Informed models are built based on a parameterized system of non-linear Differential-Algebraic Equations (DAEs) of sufficiently high order (see e.g. [1]). These models contain plenty of physical information about how generators operate. On the contrary the Dynamic System (DS) approach based on Machine Learning, which may be of a regression or neural-network type, posses almost no prior information about how PG operates. We aim to devise, train, and compare the performance of both types of models, power-system informed type and general dynamic system type.

In both cases we consider supervised learning setting, where $(P(t), Q(t))$-inputs are mapped to $(V(t), \varphi(t))$-outputs. Training data will be taken from synthetic PG software simulator for a generator connected to an exemplary simple model of the rest of the system with embedded randomness, running the aforementioned PG DAE for a fixed set of generator parameters.

In the power-system informed case, training consists of reconstructing parameters of the DAEs from the data available. An efficient custom algorithm with an embedded adjoint method is designed to accomplish this DAE-learning task. We will also learn a Dynamic System (DS) model of the generator utilizing learning methodologies developed for this task in the literature.

Applications of machine learning to the study of dynamical systems has been in discussion at least since the 90s [2]. The subject has gotten a new boost recently. Different methods have been proposed for system identification, ranging from regression [3] to deep neural networks [4] and reinforcement learning [5]. The most recent result to date [6] proposes a general approach to system identification both in a black box and white box scenarios. In this manuscript, we develop and apply some of these DS-learning approaches to learning dynamics of a power generator embedded into a power system on the transmission level. Specifically, we test performances of Vector Auto-Regressive (VAR), Weight-Dropped Long Short-Term Memory (WD-LSTM) Neural Network, and Neural

Ordinary Differential Equation (N-ODE) learning schemes. The DS-learning schemes are listed here in the order of increasing complexity, with VAR being linear, thus of limited reconstruction power and accuracy, but also fastest in training and inference, while WD-LSTM and N-ODE being nonlinear, advanced ability to represent a complex dynamic system, but also heavier in training and inference.

Extensive comparative analysis of the DAE-learning and three DS-learning schemes' performance is provided in the regimes representing different domains of the original model parameters and their level of uncertainty (fluctuations). A practical consequence of this analysis is identification of a sequence of transitions from the regime where even the lightest and least physics-informed scheme, VAR, is sufficient for a quality modeling of the generator, through an intermediate regime where VAR scheme fails but WD-LSTM and N-ODE schemes are competitive, to the regime where only the DAE scheme is capable of a successful learning of the generator model.

The rest of the manuscript is organized as follows. In the Technical Introduction Section II we describe the learning process in Section II-A; physical model of the generator in Section II-B and our data generating model in Section II-C. Physics-informed DAE-learning and physics-free DS-learning are introduced in Section III. Our generation learning experiments are described and analyzed in Section IV.

II. Technical Introduction

A. Learning process

Learning is an optimization of the vector of parameters, $\boldsymbol{\theta}$, describing the underlying model, which may be one of the four models described below in the following four subsections. We choose the mean squared error (MSE)

$$h(\boldsymbol{\theta}) = \frac{1}{2T}\sum_{t=1}^{T} \|\boldsymbol{S}_t - \hat{\boldsymbol{S}}_t\|_2^2 \to \min_{\boldsymbol{\theta}}, \tag{1}$$

as the loss function (objective) in the optimization (training).

We solve the optimization (1) using Adam algorithm/method from [7], which can be viewed as an extension of the stochastic gradient descent (SGD) method. The resulting optimization is a two-stage iterative procedure of inference (forward propagation resulting in an estimation of output $\hat{\boldsymbol{S}}$ for current $\boldsymbol{\theta}$) and learning (back propagation by Adam resulting in the update of $\boldsymbol{\theta}$). For a black-box DS approach, estimation of $\hat{\boldsymbol{S}}$ is inference by forward propagation through the set of regression relations, neural network, or neural ODE given the input vector $\boldsymbol{V}$. For the physics-informed white-box approach, estimation of $\hat{\boldsymbol{S}}$ is direct propagation/solution of the DAEs.

B. Physical Model of the Generator

We chose to work with the most universal and commonly accepted in the power system literature, see e.g., [1], third-order (one-axis) model of a generator with a bus-fed thyristor excitation system containing PSS and AVR (both with derivative feedback) and mounted to an infinite bus. Such generator is described by a system of non-linear DAEs, presented explicitly in Appendix A for completeness, and which can also be summarized in the following compact form

$$\dot{\boldsymbol{x}} = \boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{V}), \tag{2}$$

$$\boldsymbol{S} = \boldsymbol{g}_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{V}), \tag{3}$$

where $\boldsymbol{x}$ denotes vector of the PG internal state; vector, $\boldsymbol{V}$, includes, as components, voltage V and phase φ at the terminal bus; $\boldsymbol{S}$ is the two-component vector of apparent power through the terminal bus (including active power, P, and reactive power, Q); and $\boldsymbol{\theta}$ is a vector of the generator parameters. Given that in our setting of the supervised learning $\boldsymbol{V}$ is known, we can exclude it from the consideration. Then the system of DAEs splits into two disconnected parts: an ODE (2) governing dynamics of $\boldsymbol{x}$ under known $\boldsymbol{V}$, and an algebraic relation (3) giving an explicit expression for the output vector $\boldsymbol{S}$. Therefore, estimation of $\hat{\boldsymbol{S}}$ in the physics-informed (white-box) approach is simply solution of the ODE (2) followed by the explicit non-linear transformation 3. We will thus further refer to DAE-learning as ODE approach.

C. Data Generating Model

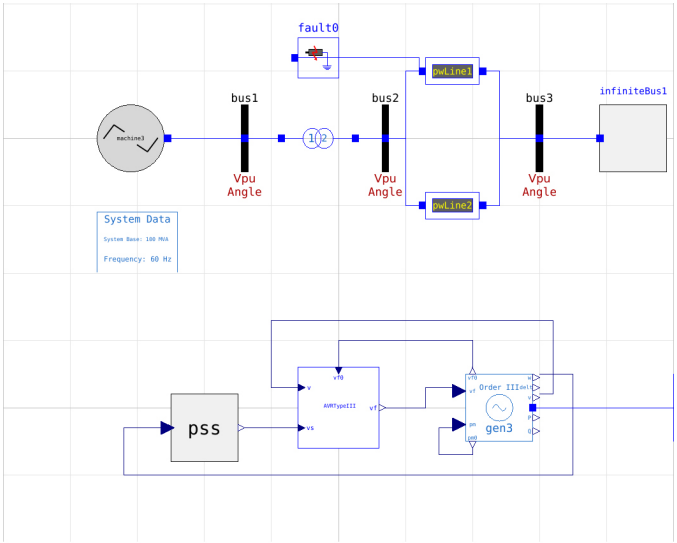


Fig. 1. Example grid (top) and machine (bottom) from [1] (example 13.2).

We work with synthetic data generated in OpenModelica environment using OpenIPSL [8] library providing an open-source implementation of a dynamic simulator of an exemplary grid, explained in example 13.2 of [1], also shown in Fig. 1, where the generator model is the one described in Section II-B. We start with a stationary initial condition corresponding to the solution of the static AC power flow found with PSAT [9]. Then, we introduce a stochastic perturbation at the external to the generator part of the grid by adding statistically steady random perturbations implemented in the form of stochastic faults to the ground imposed at the connection between the second bus and the first power line (see Fig. 1).

The reactance of the fault was set to zero, and the resistance of the fault was chosen to lie in such a range that the resulting

dynamics does not lead to a collapse, i.e., the generator does not disconnect and rides through the fault, in spite of experiencing a potentially highly nonlinear transient. Our experiments with the setting revealed that collapse does not occur if the resistance of the fault is chosen in the range from 10^{-2} to 10^{1} p.u. Then stochastic perturbations are injected into the model through a sequence of faults of the same resistance which are opened for a fixed time of 0.1 s sequentially at times determined by a stationary telegraph process [10], which is a memoryless continuous-time stochastic process that has two distinct states $a =$ fault is present and $b =$ fault is not present, described by the following master equations.

$$\dot{P}(a,t) = -\lambda P(a,t) + \mu P(b,t), \quad P(b,t) = 1 - P(a,t) \quad (4)$$

describing probability of observing the fault at the moment of time t. Note that solution of (4) initialized in the no-fault state, $P(a,0) = 1 - P(b,0) = 0$, becomes

$$P(a,t\,|\,\pi) = \frac{\mu}{\lambda+\mu} + \frac{\lambda\pi_a - \mu\pi_b}{\lambda+\mu} e^{-(\lambda+\mu)t} \quad (5)$$

where the first and second terms on the right-hand side correspond, respectively, to the stationary values, and to the exponentially decreasing with time correction due to finite memory. We set $\mu = 0.03$ and $\lambda = 0.02$ to bias towards an open fault and to randomize the time of the first fault occurrence. It is straightforward to check that at each time step, the result of this scheme is the generation of a Bernoulli distributed random variable with fault probability $P(a,t)$. Equations 5 are included (coded) explicitly in the computational scheme so that at each time step, the fault state is sampled from the resulting distribution. We are purposely fixing the time interval of the open fault to 0.1 s in order to allow a new fault to happen when the system had not yet recovered from previous fault, thus letting faults to overlap.

We prepare 9 data sets. Each data set will contain multiple samples. There are two parts in creating a temporal sample: (a) generating a sequence of faults, and (b) producing input-output sequence $\{P(t), Q(t)\} \to \{V(t), \theta(t)\}$ correspondent to the given sequence of faults by running the OpenModelica model described above for a fixed $\boldsymbol{\theta}$ describing the state of the generator. For convenience, we also provide data set details in Table I.

Data sets #1-3 are "simple" correspondent to a unique fixed value of the vector of the generator parameters, $\boldsymbol{\theta}_0$. Then for each data set, we select and fix fault resistance. Data set #1 has the largest resistance, and thus cause the least intense faults, and data set #3 has the smallest resistance, producing the largest perturbations. Values of the resistance are selected in such a way that the system does not collapse, and the generator rides through even under the largest of the three perturbations. Then, for each set, we generate 100 samples, each containing a sequence of random faults of fixed amplitude generated according to (4), as described above.

Data set #4 is "mixed" in the sense that when we generate random samples (200 for this set). We assign an amplitude to any new fault at random from the three values used to generate data sets #1-3. $\boldsymbol{\theta}$ is the same as in the data sets #1-3.

Data set #5 is the extreme version of the data sets #1-3. It is generated in the same way as data sets #1-3. However, using a smaller value for the fault resistance, such that any further decrease of the fault resistance would leave to the generator tripping.

Data sets #6-9 are equivalent to the data sets #1-4, however now for each sample we also select a new vector of the generator parameter generated i.i.d. at random from the Gaussian distribution $\boldsymbol{\theta} \sim \mathcal{N}(\boldsymbol{\theta}_0, 0.1 \cdot \mathrm{diag}(\boldsymbol{\theta}_0))$.

TABLE I
COMPARISON BETWEEN DATA SETS.

Number	1	2	3	4	5	6	7	8	9
Fault resistances	10	1	0.1	all	0.01	10	1	0.1	all
PG parameters	fixed					random			

III. One Physics-Informed and Three Physics-Free Learning Schemes

For all four of our learning schemes described in the following, we use the same strategy, borrowed from [11]. The strategy, originally developed for deep (i.e., heavily parametrized) neural networks, includes cyclical learning rate, cyclical momentum, and weight decay.

A. Learning Physics-Informed Model of Generator - ODE

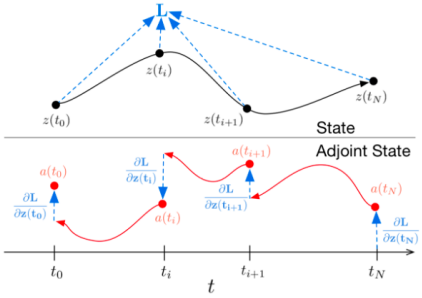


Fig. 2. Adjoint method illustated, figure from [12].

In order to compute estimation $\hat{\boldsymbol{S}}$ in the physics-informed setting, we leverage the adjoint method explored in [12]. The essence of the method is the use of adjoint quantity $\boldsymbol{a}(t) = \partial(\boldsymbol{\theta})/\partial\boldsymbol{x}(t)$, where $h(\boldsymbol{\theta})$ is the optimized functional (1) and $\boldsymbol{x}(t)$ is the state variable of (2). Then, the gradient of $h(\boldsymbol{\theta})$ with respect to the parameters $\boldsymbol{\theta}$ can be computed efficiently with one call to ODE solver via

$$\frac{dh}{d\boldsymbol{\theta}} = \int_{t_0}^{t_1} \boldsymbol{a}(t)^\top \frac{\partial \boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{x}(t), t)}{\partial \boldsymbol{\theta}} dt \quad (6)$$

The advantage of the adjoint method is in the fact that it allows utilizing a numerical integration scheme alongside with the Adam (gradient-based) optimization algorithm. We find the simplest Euler integration scheme satisfactory for our needs;

however, the approach is not limited to it. Our implementation of the adjoint method is also adapted to the model details, described in Section II-B, specifically to support exogenous inputs and constraints.

Gradient-based optimization works best when optimized parameters are of equivalent magnitude. Thus, we use re-scaled parameters for training and consider the optimization over d-dimensional cube $[0,1]^d$, where $d = 7$ is the cardinality of $\boldsymbol{\theta}$. At every iteration of Adam, we perform ℓ_∞ projection onto this cube.

We observe that the issue of parameters' initialization is of paramount importance. One would guess that, given that the parameter range is restricted, initializing uniformly at random from the cube's interior may work. However, this approach incidentally leads to physically infeasible configurations, preventing us from reaching the desired optimum. To avoid this problematic situation, we choose to initialize the parameters in such a way, that the residual error of (2) at the initial (steady state) time step is minimized, i.e. initial condition $\mathbf{0} = \boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{x}_0, \boldsymbol{V}_0)$ is satisfied approximately.

Notice that the ODE approach shows a strong sensitivity to how the initial value, $\boldsymbol{x}_0$, is set. We calculate this value for stationary power flow and use it to initiate all of our experiments. This works in all of our experiments, while it may not hold for some other cases. We leave this issue as an important topic for future analysis (see also Section V).

B. Learning Vector Auto-Regressive Model

Auto-regressive (AR) model is a proven tool in many stationary time series applications, e.g., in the field of econometrics and network-wide grid parameters such as line impedances [13]. In our case, it reads

$$\boldsymbol{y}_t = \boldsymbol{\mu} + \sum_{i=1}^{p} \boldsymbol{A}_i \boldsymbol{y}_{t-i} + \sum_{i=1}^{p} \boldsymbol{B}_i \boldsymbol{x}_{t-i} + \boldsymbol{\varepsilon}_t, \tag{7}$$

where $\boldsymbol{y}_t$ is the endogenous time series, $\mathbf{x}_t$ is the exogenous time series, μ, $\boldsymbol{A}_i$ and $\boldsymbol{B}_i$ are the parameters to be learned, $\boldsymbol{\varepsilon}_t$ is a stochastic white noise process and p is the order of the VAR model.

To determine the optimal order of the VAR model, we carry out several experiments, computing correlations between input and output variables. Given two time-series, we construct a matrix of correlations, C, between the two variables shifted by different time lags (columns for lags in first variable and rows for lags in the second). Next, we invert C and investigate non-zero elements. For Gaussian noise, non-zero elements of the upper triangle of C^{-1} signifies conditional independence (lack of correlations) between first variable's present and second variable's past, while non-zero elements of the lower triangle denote the converse [14]. We immediately obtain a causal relation between the variables, indicating that inputs and outputs are set up correctly. We also derive from this calculation, that for all pairs of variables, notable dependence occurs with time lags of up to 32. Based on this observation, we set the order of the VAR model to be equal to 32.

This decision is further supported by information criteria-based order selection methods built in the software we use to test the method. We have experimented with multiple criteria, including Akaike information criterion (AIC) from [15], where $\mathrm{AIC} = 2k - 2\ln\mathcal{L}$, k is the number of parameters and $\mathcal{L}$ is the maximum likelihood achieved by the model, final prediction error (FPE) [15], Hannan-Quinn information criteria (HQIC) and Bayesian information criterion (BIC) [16]. We observe that AIC yields the most stable results for our case, yielding the same, as discussed above, lag of 32 time steps.

C. Weight-Dropped LSTM Learning

In this physics-blind setting, we follow state of the art in sequence modeling, see e.g., [17], and utilize the WD-LSTM. Two other specifics in the implemented scheme are related to the use of the multiple LSTM approach and its applications to the supervised setting (input-output relation). Experiments with various LSTM architectures, utilizing a number of different training techniques, e.g., dropout, hidden-to-hidden dropout, weight decay, and varied sequence length, have revealed superiority of the LSTM scheme to other ML options linked to its ability to capture wide and varying ranges of data dependencies. We chose to work with the adapted version of WD-LSTM from [18], also sticking to the WD-LSTM hyper parameters described in the paper.

D. Neural ODE

Proposed in [12], Neural ODE is the novel "continuous time" deep architecture approach taking advantage of the adjoint methodology for the back propagation. The method consists in seeking for a vector ODE with nonlinearity (right-hand side) represented by a deep neural network. Specifically, we use a nonlinearity given by a two-layer fully-connected neural network with 100 hidden units and a rectified linear unit activation function between the layers. We also leverage dropout [19] to improve performance.

This method is a natural fit for the problem of our interest, as it combines the ODE structure, motivated by the physical model of the generator, with the efficiency of the neural networks. As noticed in [12], it is often difficult to guess correctly the structure of the underlying differential equation representing a complex dynamical system. Moreover, picking a wrong type of ODE may introduce biases in parameter estimates skewing predictions. Therefore, Neural ODE may be advantageous to other physics-blind approaches as introducing some (even though not too much) of the underlying physics into a black-box approach.

IV. Experiments and Results

We use the same experimental setup to train and examine all the four models we built to represent a power generator. We train the models to read (V, φ) pairs sequentially and output (P, Q) in a (varied-length) window. The objective function is the mean squared error (MSE), introduced in (1), which juxtaposes model outputs at each step and the ground truth

data. We estimate the quality of the prediction using the following normalized root mean square error (NRMSE) metric

$$\mathrm{NRMSE}(\mathrm{x},\mathrm{y}) = \sqrt{\frac{1}{\mathrm{T}}\sum_{\mathrm{t}=0}^{\mathrm{T}}\|\mathrm{x_t}-\mathrm{y_t}\|_2}\Big/\sqrt{\frac{1}{\mathrm{T}}\sum_{\mathrm{t}=0}^{\mathrm{T}}\|\mathrm{x_t}\|_2} \quad (8)$$

We train each black-box DS model on the data sets #1-4, i.e., data sets with a specific unique value of the generator parameters, $\boldsymbol{\theta}$. Then, we examine the quality of the training on all other data sets (not used in training) and report results in the form of the colormaps described below. We also continue to train the models resulted from the combined data set #4 a bit further (fine-tuning the models) on the data set #9, where $\boldsymbol{\theta}$ changes randomly from sample to sample, to improve the generalizability of the models. We then evaluate these models on data sets #5-8 and report the results in Table II.

In the case of physics-informed approach, the training data naturally plays a less important role. We even find that learning from static power flow values produces a decent approximation of parameters. Thus, we only train our physics-informed approach on data set #4. We evaluate the obtained solution on data sets #6-9 and report the results in Table II. In the case of the physics-free models, we use the realistic temporal resolution of 10 measurements per second. In the case of the physics-informed model, we use the enhanced resolution of 10^4 measurements per second.

TABLE II
COMPARISON BETWEEN MODELS, MEAN NRMSE IN %.

Model	Test fault resistance			
	10	1	0.1	mix
VAR	-	-	-	4.72
WD-LSTM	0.04	0.39	3.77	2.09
ODE	0.24	2.26	12.78	8.01
N-ODE	0.98	1.11	5.18	3.30

A. WD-LSTM

As shown in Fig. 3, given more complex data for training, the WD-LSTM approach performs better on less complex data. Also, learning on more complex data allows for slight improvement in performance on high-order noise data.

Most importantly, a color map in Fig. 3 highlights the lack of domain generalization, as models trained on one fault parameters poorly generalize to others. We approach this issue via learning from randomized generator parameters. Even though the model manages to produce reasonable results for noisy generator parameters, the model does not improve on highly perturbed data reliably, as Table II shows.

Finally, in Fig. 4 and Fig. 5 we present examples of the output signal generated by LSTM model from input sequences (not seen during training) in different nonlinear regimes.

The reconstruction is not perfect, in compliance with NRMSE values in Table II. However, the difference is very subtle and is localized mostly in the areas of extreme deviation from static power flow values. Note that in the mildly nonlinear regime the reconstruction is close to perfect.

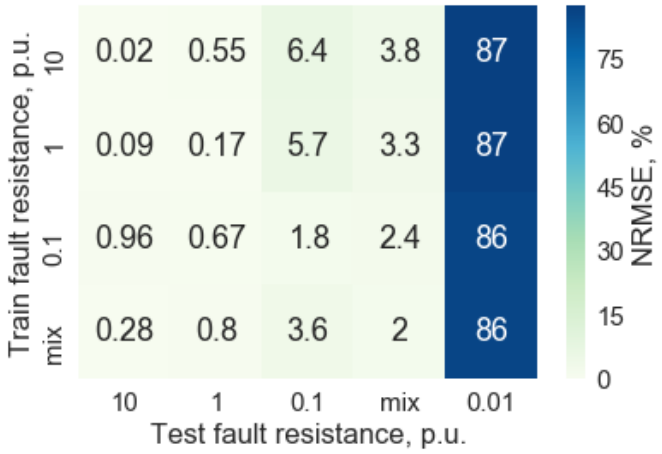


Fig. 3. Colormap of the LSTM performance, brighter is better.

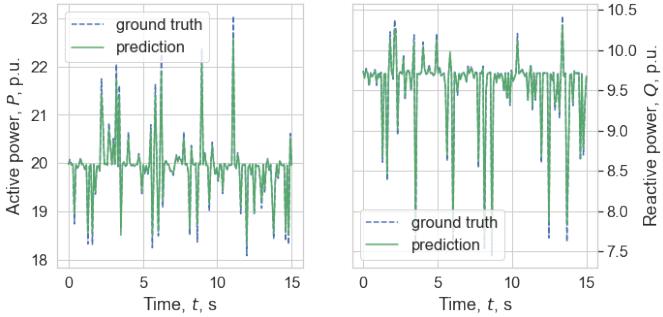


Fig. 4. LSTM performance on (unseen) example from data set #9.

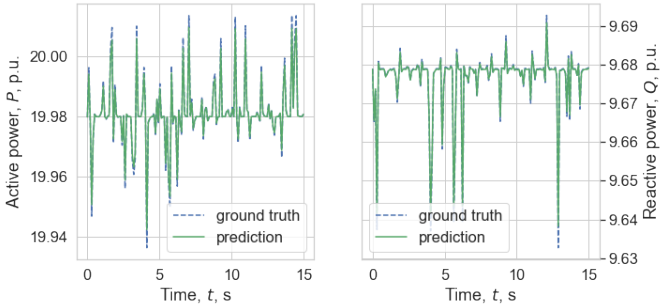


Fig. 5. LSTM performance on (unseen) example from data set #1.

B. ODE

From Table II, we conclude that ODE constitutes a strong competitor to all physics-blind approaches. With larger perturbations, however, non-optimality of learned parameters plays a more significant role, preventing the model from performing on par with others.

We also experiment to examine the evolution of parameters throughout the training. We note that the considered parameters fall into three groups: 4 parameters describing reactances of the three axis of the generator and infinite bus, and per parameter for gains in exciter, voltage regulator and stabilizer. For each of these groups, we observe different convergence. For the first group, we obtain nearly optimal variables within

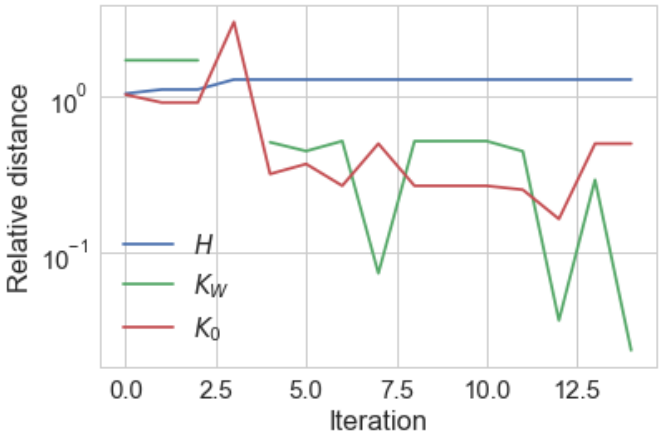


Fig. 6. Convergence of physical parameters.

one iteration on static AC power flow values. We conclude that these parameters can be estimated with sufficient precision and fix them for later training procedures. For the remaining parameters, we obtain behavior illustrated in Fig. 6.

C. Neural ODE

Interestingly enough, we can derive from color maps in Fig. 7 that Neural ODE does not exhibit behavior similar to WD-LSTM, i.e., providing more complex data during training does not introduce a notable improvement to quality on less complex data. On the other hand, Neural ODE generally performs better on data with a higher-order noise.

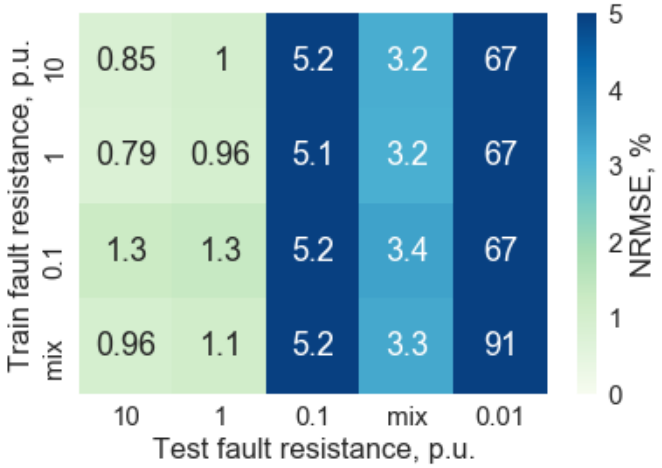


Fig. 7. Color map of the Neural ODE performance, brighter is better.

V. Discussion and Path Forward

A. Pros and cons of the proposed schemes

a) VAR: This approach is the lightest of all considered. It should be considered as an implementation light, working horse for online adjustments suitable for an automatic embedding into real-time simulators. This method is only limited to linear or almost linear regimes, and it lacks interpretability. Thus it is not suitable for generating in-depth corrective suggestions (on what to modify at the generator).

b) WD-LSTM: An important feature of this approach is more efficient training and inference, as the required computations are matrix multiplications. The WD-LSTM allows generalizations, that is it can be used when the model is not specified or even when it is specified incorrectly. The most notable drawbacks of the method are lack of interpretability and fixed temporal resolution. We conclude that the scheme has a niche in a mildly non-linear regime requiring no corrective control (latter would not be possible without interepretation). In this regime the prediction matches ground truth almost perfectly.

c) Physics-based ODE: Contrary to above discussed approaches, it offers interpetable solution stated in terms of much fewer number of physically-sensible parameters. An additional benefit of the physics-informed learning can be performed even on the static (not varying in time) data. Notice, however, that the latter feature of the ODE approach also has a flip side - one needs to work with the correct model and initialize training from the condition which is quite close from the actual operational values. Significant advantages of this physics-based ODE approach, completely unavailable via WD-LSTM, are in the methods ability to handle sample irregularity (both in the training/learning and inference stages) and its remarkable ability to produce sensible results in the highly nonlinear case. This important latter feature of the method suggests that its niche is monitoring emergencies, i.e. highly nonlinear regimes, which will most probably result (if the emergency is confirmed) in a fast adjustment (control) of the generator. In practice Physics-based ODE can be used as an alternative efficient parameter learning procedure for traditional reduced-order models.

d) Neural ODE: is positioned right in the middle between the physics-based ODE and the physics-free WD-LSTM, both in terms of its pros and cons, and thus may offer a good compromise. As other NN-based schemes, it allows efficient training, handles miss-specification well, and also supports irregularity in measurements. Overall the Neural ODE is well-positioned to take the intermediate niche of the methods – most appropriate for the mildly nonlinear regime when no follow-up control actions are required. However, and on the flip side, the method offers no physical explanations. In this regards, the method can be used as a pre-conditioner for computationally heavier, but quality-wise superior Physics-based ODE methods in the case when the level of nonlinearity exceeds a pre-defined threshold.

B. Practical value of the results

While the developed methods are designed to solve the same problem, their practical value varies, as discussed above, depending on the regime.

In this sense, the Physics-based ODE method can be viewed as an efficient alternative to traditional methods of parameter estimation for well-established reduced-order models.

The physics-blind approaches including WD-LSTM should be viewed as a new kind of reduced-order models. Although

we did not explore efficiency of the improvement offered by the models thoroughly, we note that WD-LSTM is faster than the Physics-based ODE in both training and inference, which implies it is more efficient than well-established reduced-order models.

C. Possible extensions of the physics-informed approach

a) Parameters domain: In this work, we restrict the values of parameters from arbitrary values to a pre-defined domain in the form of ℓ_∞ ball. This choice of the parameter domain was not optimized. A different choice of parameter domain might lead to a better optimization results, provided it allows an efficient implementation of the gradient-descent projections.

b) Initialization issues: Apart from the proposed initialization of the parameters, other choices can be considered, including initializing from the solution of linearized DAEs or utilizing some physically motivated guesses.

c) Device-specificity of parameters: Notice that considering physical model of the generator, we treated all the parameters of the model on equal footing, even though better results may be achieved if the parameters are split into groups associated with different compartments/devices of the generator.

VI. Conclusion

We have introduced and tested light and accurate methods learning models of a power generator. We have also designed our custom synthetic data generation process to provide the data needed to train the models. We show that the LSTM method is truly non-linear and robust to high-order perturbations and randomization of the generator parameters. We prove, through extensive experiments, that power generator parameters can be estimated from terminal bus data through the physics-informed approaches built on low-dimensional parametrizaion of the generator's governing equations. In conclusion, preliminary analysis of the hierarchy of reduced models of a generator, designed and tested in the paper, shows promise for future use as reduced order, thus light and efficient, models in different regimes and under different circumstances. However, further tests, extending the proof-of-principles studies to setting representing power industry operations more realistically, in particular in terms of tests based on actual (not synthetic) measurements, will be needed to prove practical utility of our results.

References

[1] P. Kundur, N. Balu, and M. Lauby, *Power system stability and control*, ser. EPRI power system engineering series. McGraw-Hill, 1994.

[2] K. S. Narendra and K. Parthasarathy, "Identification and control of dynamical systems using neural networks," *IEEE Transactions on neural networks*, vol. 1, no. 1, pp. 4–27, 1990.

[3] S. L. Brunton and J. N. Kutz, *Data-driven science and engineering: Machine learning, dynamical systems, and control*. Cambridge University Press, 2019.

[4] S. Genc, "Parametric system identification using deep convolutional neural networks," in *2017 International Joint Conference on Neural Networks (IJCNN)*, May 2017, pp. 2112–2119.

[5] F. Bagge Carlson, "Machine learning and system identification for estimation in physical systems," Ph.D. dissertation, Department of Automatic Control, 12 2018.

[6] T. Meeds, G. Roeder, P. Grant, A. Phillips, and N. Dalchau, "Efficient amortised bayesian inference for hierarchical and nonlinear dynamical systems," in *International Conference on Machine Learning*, 2019, pp. 4445–4455.

[7] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[8] L. Vanfretti, T. Rabuzin, M. Baudette, and M. Murad, "iTesla power systems library (iPSL): A modelica library for phasor time-domain simulations," *SoftwareX*, vol. 5, pp. 84 – 88, 2016.

[9] F. Milano, "An open source power system analysis toolbox," *IEEE Transactions on Power Systems*, vol. 20, no. 3, pp. 1199–1206, Aug 2005.

[10] A. D. Kolesnik and N. Ratanov, *Telegraph processes and option pricing*. Springer, 2013.

[11] L. N. Smith, "A disciplined approach to neural network hyper-parameters: Part 1–learning rate, batch size, momentum, and weight decay," *arXiv preprint arXiv:1803.09820*, 2018.

[12] T. Q. Chen, Y. Rubanova, J. Bettencourt, and D. K. Duvenaud, "Neural ordinary differential equations," in *Advances in Neural Information Processing Systems*, 2018, pp. 6571–6583.

[13] A. Y. Lokhov, M. Vuffray, D. Shemetov, D. Deka, and M. Chertkov, "Online learning of power transmission dynamics," in *2018 Power Systems Computation Conference (PSCC)*. IEEE, 2018, pp. 1–7.

[14] M. J. Wainwright and M. I. Jordan, "Graphical models, exponential families, and variational inference," *Foundations and Trends® in Machine Learning*, vol. 1, no. 1–2, pp. 1–305, 2008.

[15] H. Akaike, "Information theory and an extension of the maximum likelihood principle," in *Selected papers of Hirotugu Akaike*. Springer, 1998.

[16] G. Claeskens and N. L. Hjort, "Model selection and model averaging," *Cambridge Books*, 2008.

[17] S. Merity, N. S. Keskar, and R. Socher, "Regularizing and optimizing LSTM language models," in *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*, 2018.

[18] J. Howard and S. Ruder, "Fine-tuned language models for text classification," *CoRR*, vol. abs/1801.06146, 2018.

[19] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *CoRR*, vol. abs/1207.0580, 2012. [Online]. Available: http://arxiv.org/abs/1207.0580

Appendix A
Model of Generator

A third-order (one-axis) model of an undamped generator with a bus-fed thyristor excitation system containing PSS and AVR (both with derivative feedback) and mounted to an infinite bus [1] is described by the following system of equations (9)-(20) and constraints (25)-(26). All quantities are in machine base per unit, times are in seconds, angles are in electrical radians per second.

$$\dot{\delta} = \omega_b(\omega - 1) \tag{9}$$

$$\dot{\omega} = \frac{1}{2H}\left(T_M - (V_q + R_a I_q)I_q - (V_d + R_a I_d)I_d\right) \tag{10}$$

$$\dot{v}_2 = K_W \dot{\omega} - \frac{1}{T_W} v_2 \tag{11}$$

$$\dot{v}_s = \frac{1}{T_{pss2}}\left(T_{pss1}\dot{v_2} + v_2 - v_s\right) \tag{12}$$

$$\dot{V_M} = \frac{1}{T_R}\left(V_t - V_M\right) \tag{13}$$

$$\dot{V_R} = \frac{1}{T_{avr2}}\left(K_0\left(1 - \frac{T_{avr1}}{T_{avr2}}\right)(V_{ref} + v_s - V_M) - \right. \\ \left. - V_R\right) \tag{14}$$

$$\dot{E_{fd}} = \frac{1}{T_e}\left(\left(V_R + K_0\frac{T_{avr1}}{T_{avr2}}\left(V_{ref} + v_s - V_M\right)\right) \cdot \right. \\ \left. \cdot \left(1 + s_0\left(\frac{V_t}{V_M} - 1\right)\right) - E_{fd}\right) \tag{15}$$

$$\dot{E}'_q = \frac{1}{T'_{do}}\left(-E'_q - (X_d - X'_d)I_d + E_{fd}\right) \tag{16}$$

$$I_d = \frac{1}{X'_d X_q + R_a^2}\left(X_q E'_q - X_q V_q - R_a V_d\right) \tag{17}$$

$$I_q = \frac{1}{X'_d X_q + R_a^2}\left(R_a E'_q - R_a V_q + X'_d V_d\right) \tag{18}$$

$$P = V_d I_d + V_q I_q \tag{19}$$

$$Q = V_q I_d - V_d I_q \tag{20}$$

$$V_R = V_d \sin\delta + V_q \cos\delta \tag{21}$$

$$V_I = -V_d \cos\delta + V_q \sin\delta \tag{22}$$

$$V_t = \sqrt{V_R^2 + V_I^2} \tag{23}$$

$$\varphi = \arctan\frac{V_I}{V_R} \tag{24}$$

$$v_{s\,\min} \le v_s \le v_{s\,\max} \tag{25}$$

$$E_{F\,\min} \le E_{fd} \le E_{F\,\max} \tag{26}$$

We consider the input vector, $\boldsymbol{S} = (P, Q)$, characterizing injection/consumption of active and reactive powers at the terminal node of the device (generator), network terminal variables vector, $\boldsymbol{V} = (V_d, V_q, I_d, I_q)$, characterizing voltage and phase measured at the terminal bus, state vector $\boldsymbol{x} = (\delta, \omega, v_2, v_s, V_M, V_R, E_{fd}, E'_q)$, describing phase, frequency and a set of other variables representing voltages/potentials at different internal nodes/elements of the device, and $\boldsymbol{\theta} = (R_a, X'_d, X_d, X_q, H, K_W, K_0)$. All other entities entering the equations are considered constants.

There is ambiguity in definition of $\boldsymbol{V}$, which is illustrated with equations (21)-(24); we choose to represent $\boldsymbol{V} = (V_t, \varphi)$, motivated by physical availability of these values.