Pacific Northwest
NATIONAL LABORATORY

PNNL-34963

Increased Interpretability for Model-driven Deception

MARS LDRD Project

September 2023

William Hofer
Oceane Bel
Burhan Hyder
Matthew Bruggeman
Thomas Edgar

U.S. DEPARTMENT OF ENERGY

Prepared for the U.S. Department of Energy
under Contract DE-AC05-76RL01830

DISCLAIMER

PACIFIC NORTHWEST NATIONAL LABORATORY
operated by
BATTELLE
for the
UNITED STATES DEPARTMENT OF ENERGY
under Contract DE-AC05-76RL01830

Printed in the United States of America

Available to DOE and DOE contractors from
the Office of Scientific and Technical Information,
P.O. Box 62, Oak Ridge, TN 37831-0062
www.osti.gov
ph: (865) 576-8401
fox: (865) 576-5728
email: reports@osti.gov

Available to the public from the National Technical Information Service
5301 Shawnee Rd., Alexandria, VA 22312
ph: (800) 553-NTIS (6847)
or (703) 605-6000
email: info@ntis.gov
Online ordering: http://www.ntis.gov

PNNL-34963

Increased Interpretability for Model-driven Deception

MARS LDRD Project

September 2023

William Hofer
Oceane Bel
Burhan Hyder
Matthew Bruggeman
Thomas Edgar

Prepared for
the U.S. Department of Energy
under Contract DE-AC05-76RL01830

Pacific Northwest National Laboratory
Richland, Washington 99354

Summary

Machine learning has been proposed as a solution to several cybersecurity solutions and one of the most promising applications is for digital twins for intrusion detection and driving deceptive defense. However, machine learning techniques often result in a black-box function that is difficult for end users to interpret which for deception limits their ability to effectively define decoys. In this report, an approach to validate the equations learned are accurate is provided and demonstrated. Following, begins the process of addressing this issue for a model-driven deception technology that produces equations representing the physical process controlled by operation technology devices. This research was performed by applying subject matter expert context to machine learned models.

Acknowledgments

This research was supported by the **Mathematics for Artificial Reasoning in Science (MARS) Initiative**, under the Laboratory Directed Research and Development (LDRD) Program at Pacific Northwest National Laboratory (PNNL). PNNL is a multi-program national laboratory operated for the U.S. Department of Energy (DOE) by Battelle Memorial Institute under Contract No. DE-AC05-76RL01830.

This project team would like to acknowledge the Mathematics for Artificial Reasoning in Science (MARS) Initiative and the leadership team for providing the opportunity to conduct this research.

Contents

Figures

Tables

1.0 Introduction

Honeypots, Honeynets, and Cyber Deception are a cyber security defensive technique that tries to cause an attacker to accept as true or valid what is false or invalid to gain an advantage in cyberspace. Cyber deception can be defined as planned actions taken to mislead hackers with the goal of causing them to take, or not take, certain actions in favor of the defenders (Yuill, 2007). In general, deception is used to confuse an adversary and give an advantage to the party employing the deceptive technology. While there is a long history of cyber deception applied to information technology (IT) systems, only relatively recently has this technique been applied to operational technology (OT) systems.

Operational technology is a unique cross section of computing and process control engineering. The term is defined by NIST as "hardware and software that detects or causes a change through the direct monitoring and/or control of physical devices, processes and events in the enterprise." (Stouffer, K. et al., 2017). Structures involving this technology are referred to as industrial control systems (ICS).

A key differentiator for applying cyber deception to OT environments is that they include a physical process that underpins all the digital devices and not just data and networked services like IT. To make a decoy appear as part of the system, it is necessary for it to not only respond like an embedded field device but with data that appears to come from the physical process (Green, B. et al, 2017). Model-driven deception entails deploying decoy industrial control system devices in a cyber environment to entice and confuse potential cyber threats. Existing approaches use physical process models, simulated or machine learned, to act as a forecasting/predictive engine for the physical response to control inputs coming from decoy controllers. These decoy controllers are then exposed on the cyber network to entice and confuse potential attackers.

If the learned model is presented in a non-understandable way, it is difficult to use. Increased interpretability of the backend model will allow cyber analyst/incident response personnel to optimally deploy defensive deception campaigns. The current technology has a limitation in how it presents the model to the end user, in this case a cyber security personnel. The model is either represented as a black box or, in the current machine learning method, an equation. To provide a more interpretable and usable model, this project aims to map learned physical process models to Subject Matter Expert (SME) derived equations to categorize the modeled equations into common industrial control system devices, ultimately giving the end user a choice in the form of what control system device decoys they want to deploy. The remainder of this report will outline current work being done to validate the Machine Learning (ML) method and SME equation derivation and increase the interpretability and usability of model-driven deception defense.

2.0 Related Work

There are two fields of which this research pulls from. There is the field of deception which is focused on how to develop effective decoys and the field of modeling cyber physical systems using machine learning to accurately represent system behavior. This work is contributing knowledge for both fields.

Recent cyber deception capabilities for OT environments have been developed that implement control system protocols and device information (Conpot, n.d.; Rist, Lukas, 2015). There have also been efforts to develop high interaction deceptions (Antonioli, D. et. al, 2016) These implementations do not consider the physical process these devices are controlling. ML approaches have been used to develop models of system operation to assist in detecting security events (Yan, W, et. al, 2018) but have not been used for decoys. Further research has been done to explore strategies for model-driven deception for cyber-physical system environments (Nowak et al., 2021). Various ML strategies have been evaluated for this model-driven approach (Edgar et al., 2020). This body of work addresses the interpretability problem identified in this previous work.

In recent years there has also been a push to develop reproducible ML approaches that resemble closer to a white box approach (Linardatos, P., 2020). A white box approach includes approaches such as linear regression and decision trees. On the other hand, black box approach includes models like deep learning or federated learning models (LeCun, Y., 2015; Yang, Q. et al., 2019). These models usually don't have a heuristic representation and adapt to changes in data through training steps. By using an Equation Learning (EQL) engine to extract a heuristic representation from a black box approach such as a dense model, we can extend the explainability of the model by comparing the resulting equation to SME equations that are built from existing literature (Sahoo et al., 2018).

Validating a black box model is usually done via accuracy study to quantify the mean standard error that quantifies how closely related the prediction is to the real data (Taylor, B.J., 2006). Additionally, the error is also quantified through standard deviation differences between the predicted array and the target array. The second parameter measures how well the model captured the variations of the target data. Capturing the variation of target data becomes useful when applying a model to performance issues on systems. For example, in the Diolkos project, the framework looks for dips in performance to reroute data to other ports to reduce impact of network bottlenecks (Bel, O. et al., 2021). If the model can't correctly capture the variations in the data, the framework runs the risk of rerouting packets from a port that may be performing without issues to a port that may be starting to get overwhelmed.

3.0 Research Objectives

The goal of this research was to improve the understandability of the Machine Learning learned equations and model of a Cyber Physical System (CPS). Deception defense could be a very useful technique for defending CPS but it is predicated on the ability of defenders to define effective and relevant decoys to the system they are defending. In this report we address the challenge of increasing domain awareness for AI in both cyber deception strategy and control engineering/theory to improve deception campaign deployment. Current approaches to learning a model of a CPS uses equation learning to generate a series of formulas representing data collected from an operational system (Edgar, et al., 2020). These equations provide a model to drive a simulation of the system to enable the deployment of decoy controllers and sensors backed by realistic data. However, without context on what the learned equations represent it is hard to determine how to deploy realistic and effective deceptions. Comparing the learned equations with well-established domain relationships for control equations, we can better inform the AI about the physical laws and limitations of how the CPS operates. As an added benefit the cyber security personnel also gains insight into what the equations represent and can better adapt their defensive strategy.

The scope of this work was to increase the interpretability of a machine learned model for Industrial Control System (ICS) physical processes. Within this scope, interpretability can be defined as the confidence that a human would have in the accurateness and appropriateness of the machine learned model.

The objectives of this work were the following:

- Develop a set of subject matter expert (SME) equations that represent industrial control system devices, scoped to power distribution systems.

- Develop a module that uses the SME equations to ingest input values and produce sample output timeseries datasets

- Develop a methodology for comparing SME equation output to machine learned model output to categorize ML system models as specific control system devices

- Upon successful categorization, develop a recommendation system for deception/decoy deployments

4.0 Data Acquisition

To begin this research, it was necessary to have a system from which data could be generated for ML and validation. For this work a modified IEEE 123-node power distribution system model, implemented in Opal-RT's HyperSim real-time power system simulator was adapted for this research (Comprehensive Test Feeder – IEEE PES Test Feeder, n.d.; Jinsiwale, et al., 2023). The power system model was unmodified, but the relays and data-to-be-saved were updated for this project. Specifically, saved data was narrowed in scope to focus on the inverters in the model, as well as a relay set to protect the inverter with basic protection settings (e.g., over/under voltage, over current, etc.). Additionally, the option to trigger a customizable fault was added. This fault can be used to assist in training ML approaches to better match actual grid operations.

The chosen HyperSim model of an inverter includes example variables shown in Table 1 below. The last column indicates whether the variable is an input, output, or intermediate variable in relation to the inverter control function. The ML methodology will focus on learning the relationship between the output variables as a mathematical relationship with the input variables. Intermediate values, such as the Vref variables, can be described as internal to the control function. They are not input values that would be received via digital or analog signaling, nor are they sent via digital or analog signaling as outputs from the physical device. In this use case, the use of the Vref variable would be internal to a representative device model, meaning the variable and assigned values are only passed between internal components of the control device. The intermediate values were explored as features in the ML method, but ultimately should be ignored as a practical application of this technique using live cyber network data would not include those variables.

Table 1 Example Inverter variables

Variable Name	Signal Identifier	Relationship to Inverter Control Function
Inverter_51.DEV4.Iabc(Ia1)	I_a^g	INPUT
Inverter_51.DEV4.Iabc(Ib1)	I_b^g	INPUT
Inverter_51.DEV4.Iabc(Ic1)	I_c^g	INPUT
Inverter_51.DEV4.Pref_PU_set.y	P*	INPUT
Inverter_51.DEV4.Qref_PU_set.y	Q*	INPUT
Inverter_51.DEV4.Sbase_set.y	Constant	INPUT
Inverter_51.DEV4.Vabc(Va1)	V_a^g	INPUT
Inverter_51.DEV4.Vabc(Vb1)	V_b^g	INPUT
Inverter_51.DEV4.Vabc(Vc1)	V_c^g	INPUT
Inverter_51.DEV4.Vinv_ref(Va_ref)	V_a^{ref}	Intermediate
Inverter_51.DEV4.Vinv_ref(Vb_ref)	V_b^{ref}	Intermediate

Inverter_51.DEV4.Vinv_ref(Vc_ref)	V_c^{ref}	Intermediate
Inverter_51.DEV5.P_MW	P	OUTPUT
Inverter_51.DEV5.Q_MW	Q	OUTPUT
Inverter_51.DEV5.Vabc_rms(Va_rms)	V_a^{rms}	OUTPUT
Inverter_51.DEV5.Vabc_rms(Vb_rms)	V_b^{rms}	OUTPUT
Inverter_51.DEV5.Vabc_rms(Vc_rms)	V_c^{rms}	OUTPUT

5.0 Subject Matter Expert Derived Equations

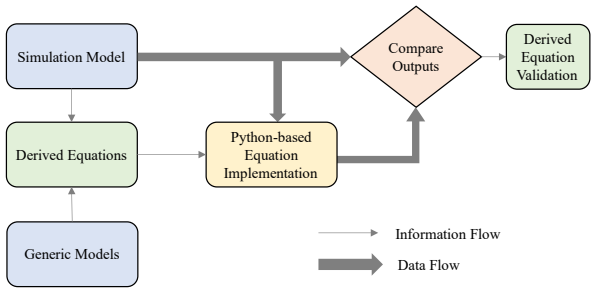

Figure 1 Methodology for SME Equation Derivation and Validation

The methodology to derive the SME equations for various devices in the grid model is shown in Figure 1. The SME analyzes the device models used in the simulation environment, defined in HyperSim in this case, and compares them with the generic models of those devices in the existing literature and based on prior knowledge of the SME. The information from these models is combined to derive equations for the devices under consideration. The derived equations are then implemented in a lightweight Python script, which emulates the behavior of the device based on the derived equations. For validating the accuracy of the derived equations, the device input data acquired from the simulation model, as described in Section 4, is fed into the Python-based device emulator and the outputs of this emulator are statistically compared, such as the Kolmogorov-Smirnov (KS) test (Frank J. M., 1951), with the acquired output data. This provides a measure of validation of the derived equations if the statistical distances between the outputs from the simulation model and the derived equations are within acceptable thresholds.

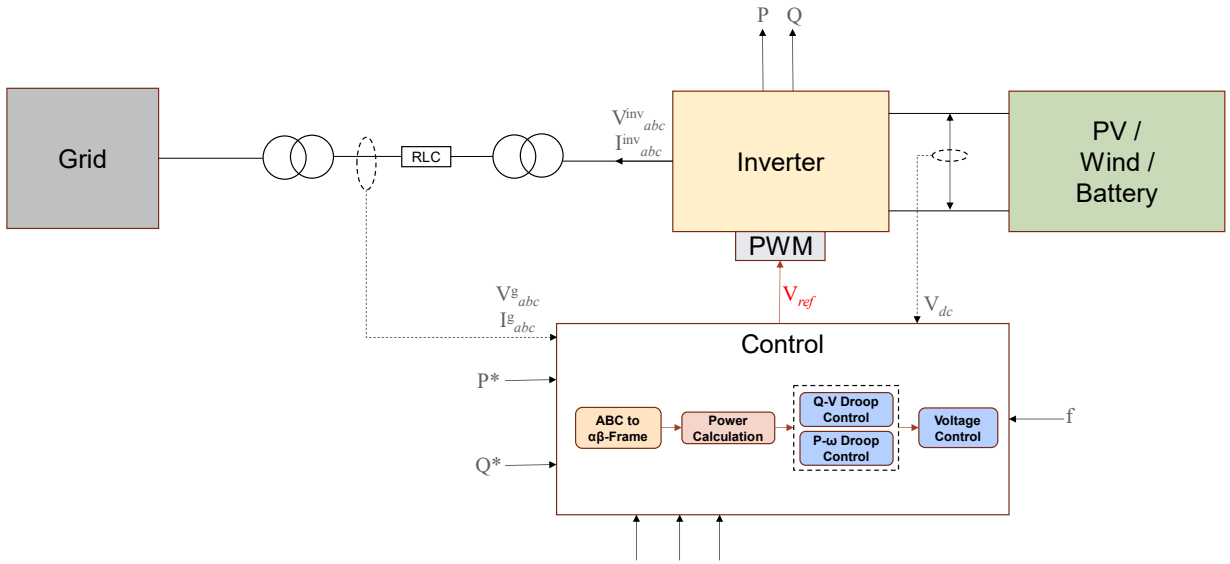

Figure 2 Single-loop control-based grid-forming voltage source inverter

As a case study, the rest of this section describes the derivation of equations that represent a single-loop control grid-forming inverter. Specifically, Figure 2 shows the system for which the SME equations are derived in this case study. This model is developed by considering the HyperSim model and grid-forming inverter models that exist in literature (Du, W. et al., 2020; Du, W. et al., 2021; Chen, M. et al., 2022)

The objective here is to derive a canonical relationship between the inputs and the outputs of the inverter, namely, $\boldsymbol{V_{abc}^{g}}$ (and $\boldsymbol{I_{abc}^{g}}$) and $\boldsymbol{V_{abc}^{inv}}$ (and $\boldsymbol{I_{abc}^{inv}}$). We have broken down the derivation of these relationships into various blocks of the "Control" block for simplicity. The equations that describe each of these blocks are delineated below:

1. ABC to αβ-Frame:

$$\begin{bmatrix} V_\alpha \\ V_\beta \\ I_\alpha \\ I_\beta \end{bmatrix} = \frac{2}{3} \begin{bmatrix} 1 & \frac{-1}{2} & \frac{-1}{2} & 0 & 0 & 0 \\ 0 & \frac{\sqrt{3}}{2} & \frac{-\sqrt{3}}{2} & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & \frac{-1}{2} & \frac{-1}{2} \\ 0 & 0 & 0 & 0 & \frac{\sqrt{3}}{2} & \frac{-\sqrt{3}}{2} \end{bmatrix} \begin{bmatrix} V_a^g \\ V_b^g \\ V_c^g \\ I_a^g \\ I_b^g \\ I_c^g \end{bmatrix}$$

2. Power (measured) Calculation:

$$P_m = \frac{3}{2}\left(V_\alpha I_\alpha + V_\beta I_\beta\right)$$
$$Q_m = \frac{3}{2}\left(V_\alpha I_\beta - V_\beta I_\alpha\right)$$
$$V = \sqrt{V_\alpha^2 + V_\beta^2}$$

3. Q-V and P-ω Droop Control:

$$V^* - V_0 = m_Q\left(Q_{mf} - Q^*\right)$$
$$\omega^* - \omega_0 = m_P\left(P_{mf} - P^*\right)$$

where $\boldsymbol{P_{mf}}$, $\boldsymbol{Q_{mf}}$, and $\boldsymbol{V_f}$ can be represented in the frequency domain as follows:

$$P_{mf} = \frac{1}{1 + \tau s} P_m$$

$$Q_{mf} = \frac{1}{1 + \tau s} Q_m$$

$$V_f = \frac{1}{1 + \tau s} V$$

$\boldsymbol{m_Q}, \boldsymbol{m_P}$, $\boldsymbol{\omega_0}$, $\boldsymbol{V_0}$, $\boldsymbol{Q^*}$, $\boldsymbol{P^*}$, and $\boldsymbol{\tau}$ are inputs (constants) to the controller.

4. Voltage Control:

$$\theta^* = \int \omega^* dt$$
$$V_{PI}^* = K_P\left(V^* - V_f\right) + K_I \int \left(V^* - V_f\right) dt$$

$\boldsymbol{K_P}$ and $\boldsymbol{K_I}$ are the PI controller proportional and integral gains, respectively (constant inputs to the controller).

5. Output of Control block:

$$V_{ref} = \begin{bmatrix} V_{PI}^{*} sin(\theta^{*}) \\ V_{PI}^{*} sin(\theta^{*} + \frac{2\pi}{3}) \\ V_{PI}^{*} sin(\theta^{*} - \frac{2\pi}{3}) \end{bmatrix}$$

The final stage is to derive the relationship between V_{abc}^{inv} and V_{ref} based on the average inverter model and is given by:

$$V_{abc}^{inv} = \frac{V_{dc}}{2} + V_{ref} * V_{base}$$

where V_{dc} is the DC link voltage, and V_{base} is the base voltage used to calculate the actual value of voltage from per unit (pu) values.

For an accurate representation of the data that an inverter exchanges over the communication networks, the root mean square (rms) value of the voltages is calculated to signify the outputs of the inverter as given below:

$$V_{abc}^{rms} = \sqrt{\frac{1}{T} \int_{t-T}^{t} (V_{abc}^{inv})^2}$$

where $t - T$ represents the time window over which the rms value is calculated.

The equation above represents the calculation of the rms voltage in the continuous time domain. In the discrete time domain, the rms voltage is calculated as follows:

$$V_{abc}^{rms} = \sqrt{\frac{1}{N} \sum_{n-N}^{n} (V_{abc}^{inv})^2}$$

where $n - N$ is the number of samples over which the rms value is calculated.

The Kolmogorov-Smirnov (KS) distance test is one of the measures we have used for the validation of the SME derived equations when referenced with the simulation models. A comparison of the inverter controller output reference voltage, V_{ref}, from the dataset and the SME derived equations is shown in Figure 3. The plot shows the cumulative distribution functions (CDF) of the two outputs along with the KS-distance, which is a measure of the maximum distance between the two CDFs. The small magnitude of the KS-distance (=0.024) is evident from the plot as the distributions of the two outputs are very similar.

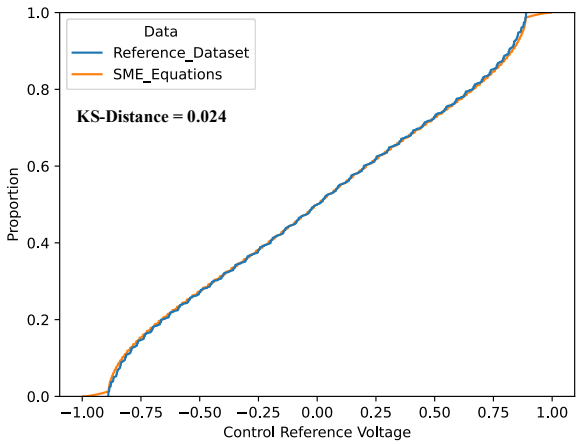

Figure 3 Cumulative Distribution Functions of V^{ref} from the Reference Dataset vs the SME Derived Equations

6.0 Machine Learning Methodology

The methods chosen in this research are an initial attempt to compare outputs in as simple a manner as possible. Other, more complex, methods may be explored in the future. Currently, most of the effort has focused on validating that both the SME equations and ML equations perform adequately so that future comparisons can be made between their respective outputs to categorize the equation being learned by the ML method.

The chosen ML method for this research involved Equation Learner code (EQL) researched previously and evaluated in a different control system context (Sahoo, S. et al., 2018; Edgar et al., 2020) The EQL code relates TensorFlow equations to regular equations. Their customized EQL layer outputs a tensor containing chunks of data that have been passed through the corresponding activation functions, and these contain all the functions that can be modeled using the EQL engine. These chunks are used as construction pieces for the final equation that captures the target values, such as the output voltage of an inverter in this works use case. We compare such equations to equations constructed from literature and to raw data to measure how well they performed.

The EQL engine has challenges, including the issue of scales in training models. The data used from the HyperSim model is on different scales that vary up to 1018 nanoseconds for timesteps and up to 10-6 volts for reference voltages. This means that the data needs some preprocessing to set all the features on similar scales so that the model can correctly train on the incoming data. To give the EQL engine a chance to create equations that could model the different components we grouped the individual features by value. The reason the features are grouped is because it allows the values to be closely related in range, which reduces noise introduced in the model.

When investigating how to compare the ML learned equations with the SME derived equations, it was discovered that directly comparing the characteristics of the equations themselves was likely infeasible. For example, the potential set of equations produced using ML may not represent the dataset sufficiently to compare to the derived equations accurately. Sequential runs of the ML algorithm over the same data produces a unique equation despite tuning. The following subsections describe the results of the initial comparison attempts.

6.1 Comparing EQL to SME

When comparing the output equations of the EQL engine to subject matter expert's equations, it becomes difficult to identify if an equation is similar since there are only so many actions that can be modeled by the EQL. For example, the EQL engine cannot model integrals. Therefore, to compare the equations we had to compare the output data of the EQL equations to the output data when using the SME's equations. Another factor that limits the equation comparison is the model structure. For more complex data, like power data, we may not be able to capture all the intricacies of the data, which makes the resulting equation perform sub-par compared to that of a subject matter expert.

Grouping features by range produced equations that produced output data with a similar standard deviation compared to the data that was outputted from the SME equations. When testing with similar range data, we found that grouping worked well for the Vref values, which have a smaller range. When attempting a similar approach for larger range data, such as the raw voltage of the inverter, we found that the equation generated by the EQL engine fails to capture the mean and

standard deviation of the data. This can be an effect of using a normalizer on the data when sending it through the EQL model. To mitigate this, we normalize the data going through the EQL equation and then inverse transform the resulting data.

This approach's results for the Vainv are shown in Table 2. The values for mean and standard deviation of the data are shown, comparing EQL learned equations to the equation derived by an SME. We can see that the mean is two times higher compared to the result from using the SME's equation and a higher standard deviation. This means that the EQL agent captured more variations which impacted the resulting mean.

Table 2 Mean and Standard Deviation of V_a^{inv}

Modeling Method	Mean	Standard Deviation
EQL	1089.23	44.13
SME Equation	549.93	21.97

The resulting equations for Varef, Vbref, and Vcref were generated by the EQL method for the reference voltage intermediate values. Figure 4 shows the graph representation of the equation for Varef. The equations use the reference voltage values as inputs to the equations. This contrasts with the SME's equation which uses other features present in the dataset like the reference power values (Pref and Qref) as the way to calculate the reference voltage. The variable X_3 corresponds to Varef, X_4 to Vbref, and X_5 to Vcref.

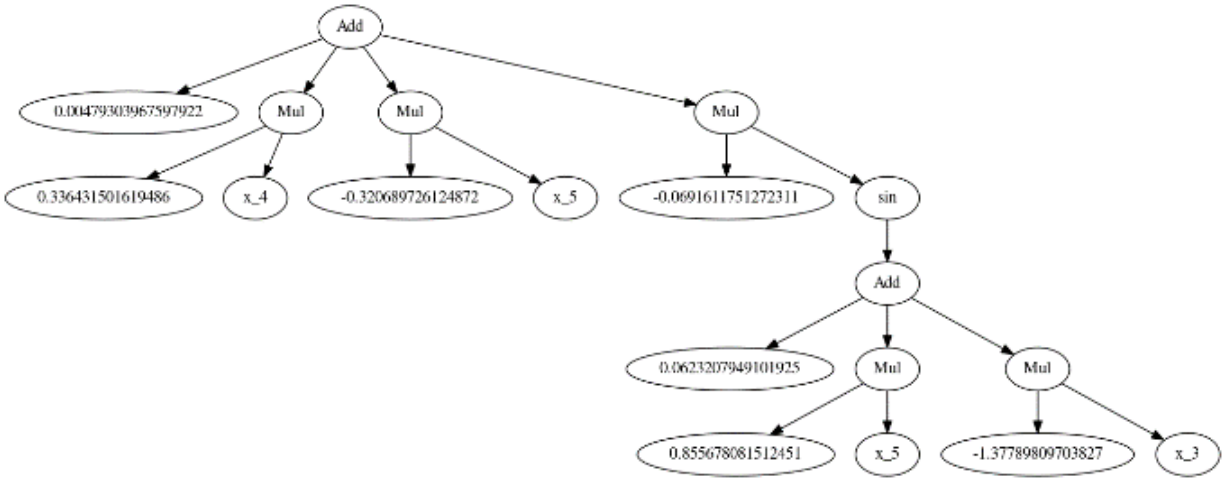

Figure 4: V_a^{ref} EQL Equation Graph

6.2 Comparing EQL and SME to Raw Data

This section describes the initial steps taken to validate the performance of the EQL and the SME derived equations. This step is necessary before results can be compared to categorize the EQL learned equations. The mean and standard deviation of the data produced by the EQL model, SME equations, and raw data from the HyperSim model were compared. They are shown in Table 3 below. When looking at the data produced by the EQL equations we can see that they matched the SME's equation in terms of standard deviation for the Varef feature. But in terms of the mean

for that feature, the EQL equations were closer to matching the mean of the raw data compared to the data produced by the SME's equations.

Table 3 Mean and Standard Deviation of V_a^{ref}

Modeling Method	Mean	Standard Deviation
EQL	0.0004	0.05
SME Equation	-0.0002	0.06
HyperSim	0.0003	0.11

To get a clearer idea on how well the EQL performs we explore the co-similarity between the generated data and the expected data from the testing dataset. This gives us a measure of the cosine angle between both arrays. This angle is used to determine if both arrays are roughly pointing in the same direction. However, the cosine similarities for features with a large range, such as the voltage, remain around 99% for all potential combinations of approaches between using the SME equations or using the EQL equations and the raw data from the dataset. This shows that even though the SME equation did not do a good job capturing the mean or standard deviation it still managed to capture the overall trend of the data.

When exploring other features, such as the Vainv feature, that have a higher range of value compared to the reference voltage we can see that the EQL agent produced an equation that produced data that is closer to the raw data compared to the subject matter expert. This is shown in Table 4 below.

Table 4 Mean and Standard Deviation of V_a^{inv}

Modeling Method	Mean	Standard Deviation
EQL	1089.23	44.13
SME Equation	549.93	21.97
HyperSim	1045.51	65.81

7.0 Discussion

The results of initial comparison of EQL generated equations to SME equations indicated the need for improvement in both SME equation derivation and EQL tuning. As indicated in Section 6.1, the mean and standard deviation for the equations for the output voltage (Vainv) values vary between SME and EQL. However, when looking at the mean and standard deviation for the output reference voltage (Varef), the values more closely match between EQL and SME and additionally match the reference dataset.

7.1 EQL and SME Validation

Improvements on the EQL method could be made to further increase the accuracy of the learned equations. The current method is only capable of representing addition, subtraction, multiplication, division, sin, and cosine. Theoretically, with enough dense layers, a model could approximate integrals and more complex operations but will need a lot of data and time to run such a test. Additionally, in the future, experimentation with other types of network nodes such as relays will be conducted. Finally, it will be necessary to fine tune the EQL model so that it produces more consistent results across runs. The current issue with this version is that the model is not seeded; therefore, the weights are randomly initialized at each run which causes slight variations on the results. Also, the model needs to be fined tuned in terms of network parameters such as epochs, training or testing set sizes. Currently for all features we used 15000 timesteps for training and 7500 timesteps for testing. This setup worked well for Varef and Vainv but may struggle for other features that need smaller or bigger sets to correctly capture variations in data. An example of features where this setup doesn't work as well is Vbref and Vcref, for which the SME's equation produced a closer standard deviation and mean, respectively, compared to the raw data.

7.2 Interpretability

The hypothesized method by which interpretability will be applied is shown in Figure 5. The variable set from a control system model, in this case HyperSim, is broken into the set of inputs and outputs for different device functions. Then those values are fed to the EQL, and a learned equation model is produced. Next, the original input and output data is fed to the learned EQL model and a validated SME derived equation. Finally, the resultant datasets are comparatively analyzed to evaluate the similarity between them.

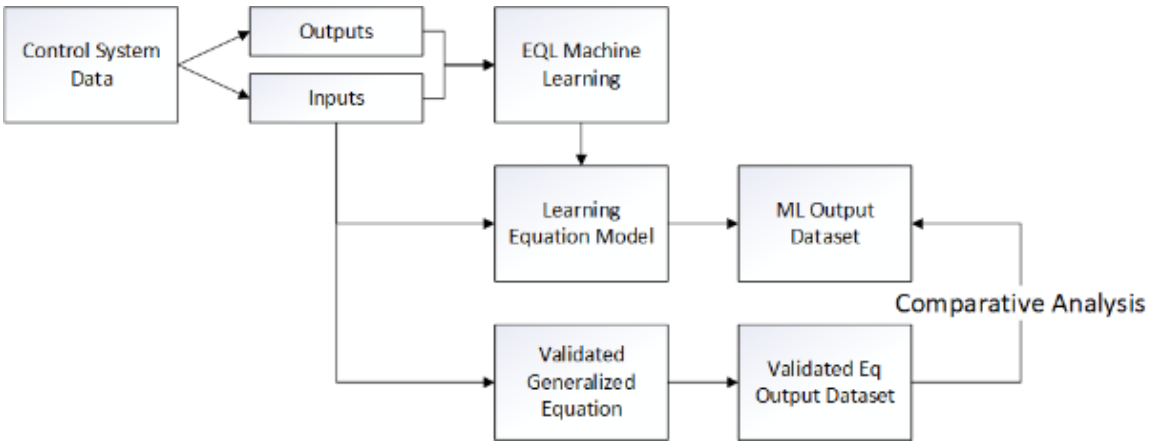

Figure 5 Interpretability Method

Increased interpretability for this model-driven deception method will require further validation before categorization can be explored. The objective at that point should be to compare SME and EQL results and match the EQL equations to the best fit within the set of SME equations. The current progress has shown that the SME equations and EQL perform relatively well when comparing to the shared reference dataset but would more closely match with further tuning. Further, it will be necessary to include other devices in this tuning process as well. Future work will include the derivation of SME equations for more power distribution system devices. This will allow for the comparison of EQL equations to multiple SME equations with the intent to find the best match and evaluate if the best match is the appropriate choice of device based on the data points being used in the equation. For example, by allowing the EQL to generate equations for all the variables within the dataset, including those related to inverter controller, relays, generator controller and others, it may be possible to identify what type of device the EQL equation represents. One simple approach to this would be to calculate the mean and standard deviation for each equation, EQL and SME, and evaluate the closest match for accuracy. From mapping learned equations to device types enables a user to quickly understand what is represented in their model and how decoys can be added to it.

8.0 Additional Human Factors Considerations

Throughout the breadth of this work, a few key research questions and considerations where identified which warrant further efforts to study the method by which model driven deception is implemented using machine learning techniques. Most of these questions and considerations relate to the human factor element in pairing ML/AI solutions with human beings. The following subsections outline the major outstanding research questions.

8.1 Questions on Feedback on Recommended Decoys

The largest body of this research focused on SME equation derivation and ML/AI produced equation categorization, however, questions arose around whether recommended decoys and deceptions would make sense when a human was introduced into the loop. For example, when and if the ML/AI algorithm categorizes a specific device type and recommends that a decoy of this device be created, a human may or may not accept that recommendation given their knowledge of the network or control system being used to generate a model. This question is further confounded when you consider the varying expertise of individuals who would interface with this type of recommender system. For example, a cyber engineer and a control system engineer may have very different responses to the recommended decoys. The research team for this body of work identified that once a recommender system is devised, it will be necessary to solicit feedback on the recommendations made from both cyber experts and control system experts to assess the human factor in a system such as this.

8.2 Questions on Generation of SME Equations

Currently, the research team solicit SME derived equations by having an SME build a normalized equation from existing control system models. It has not been determined how well these SME equations perform compared to other methods for generating SME derived equations. For example, other modeling technologies could be considered such as Modelica, GridLAB-D, or MATLAB because their representation of the control algorithms for various devices may differ from HyperSim. It is also relevant to perform a deeper investigation into literature on control algorithms to validate the current SME equations. Further, soliciting feedback from other SMEs would provide additional validation of accuracy and appropriateness.

8.3 Questions on Live System Data Input Manipulation

Early experimentation with EQL revealed that targeting equations generated for output variables that used other variables in the dataset as inputs was more successful than allowing the EQL to generate equations for all variables regardless of their use in the known control algorithm. However, in a more commercial ready implementation of this technology, live packet capture data of a control system would be used in the ML/AI modeling. This live packet capture data would not have the context of data points being inputs versus outputs.

The research team hypothesized that it may be possible to parse packet capture data and apply context to determine an initial categorization of device and in turn, determine whether the variables are inputs or outputs. It was theorized that based on request type messages versus response type messages and the general structure of the data, a determination could be made on what type of device was running at a given IP address. Further assessment of capturing live network data should be performed in the future.

8.4 Questions on Attack Objective Mapping

Future work should also be conducted to map adversarial interactions with decoys to specific Tactics, Techniques, and Procedures (TTPs). Specifically analyzing interactions with control system decoys may reveal attacker objectives such as their intended effect on the physical system being controlled.

9.0 Conclusion

In this report, the first steps towards comparing equation-based ML models to SME representations of control functions for CPS was discussed. This initial research utilized existing modeling capabilities to generate power distribution system data to train the EQL method and develop SME informed equations for the included devices. An initial equation for a grid forming inverter controller was developed.

Initial validation of the SME derived equation and the EQL method was performed as the first step before comparisons between the respective representations of the control device can be made. Additional tuning is necessary to validate the generation of SME equations and EQL models. In general, the method for comparing equations seems plausible, but until validation of both SME and EQL equations is done, and additional SME equations are generated, the results remain to be seen.

The research also began revealing additional research questions around the usability of \this model-driven deception technology. These questions include details for feedback on a recommendation system for decoy deployments, questions on generating SME equations, questions on using live system data versus modeled data, and questions on attack objective mapping.

10.0 References

Antonioli, D., Agrawal, A., and Tippenhauer, N. O. 2016. Towards High-Interaction Virtual ICS Honeypots-in-a-Box. In Proceedings of the 2nd ACM Workshop on Cyber-Physical Systems Security and Privacy (CPS-SPC '16). Association for Computing Machinery, New York, NY, USA, 13–22. https://doi.org/10.1145/2994487.2994493

Bel, O., Pata, J., Vlimant, J. R., Tallent, N., Balcas, J., & Spiropulu, M. (2021, May). Diolkos: improving ethernet throughput through dynamic port selection. In Proceedings of the 18th ACM International Conference on Computing Frontiers (pp. 83-92).

Chen, M., Zhou, D., Tayyebi, A., Prieto-Araujo, E., Dörfler, F., & Blaabjerg, F. (2022). Generalized Multivariable Grid-Forming Control Design for Power Converters. IEEE Transactions on Smart Grid, 13(4), 2873–2885. doi:10.1109/TSG.2022.3161608

Comprehensive Test Feeder – IEEE PES Test Feeder. (n.d.). Comprehensive Test Feeder – IEEE PES Test Feeder. https://cmte.ieee.org/pes-testfeeders/comprehensive-test-feeder/

Conpot. (n.d.). Conpot. http://conpot.org/

Du, W., Chen, Z., Schneider, K. P., Lasseter, R. H., Pushpak Nandanoori, S., Tuffner, F. K., & Kundu, S. (2020). A Comparative Study of Two Widely Used Grid-Forming Droop Controls on Microgrid Small-Signal Stability. IEEE Journal of Emerging and Selected Topics in Power Electronics, 8(2), 963–975. doi:10.1109/JESTPE.2019.2942491

Du, W., Tuffner, F. K., Schneider, K. P., Lasseter, R. H., Xie, J., Chen, Z., & Bhattarai, B. (2021). Modeling of Grid-Forming and Grid-Following Inverters for Dynamic Simulation of Large-Scale Distribution Systems. IEEE Transactions on Power Delivery, 36(4), 2035–2045. doi:10.1109/TPWRD.2020.3018647

Frank J. Massey Jr. (1951) The Kolmogorov-Smirnov Test for Goodness of Fit, Journal of the American Statistical Association, 46:253, 68-78

Edgar, T. W., Hofer, W. J., & Feghali, M. (2020, September 30). Model Driven Deception for Defense of Operational Technology Environments - CRADA 432 (Final Report) (Technical Report) | OSTI.GOV. Model Driven Deception for Defense of Operational Technology Environments - CRADA 432 (Final Report) (Technical Report) | OSTI.GOV. https://doi.org/10.2172/1959828

Green, B., Krotofil, M., & Abbasi, A. (2017). On the Significance of Process Comprehension for Conducting Targeted ICS Attacks. In Proceedings of the 2017 Workshop on Cyber-Physical Systems Security and PrivaCy (CPS '17). Association for Computing Machinery, New York, NY, USA, 57–67. https://doi.org/10.1145/3140241.3140254

Jinsiwale, R., Maharjan, M., Becejac, T., & Ashok, A. (2023). Evaluating a real-time model decoupling compensation approach for developing scalable, high-fidelity microgrid models. 2023 IEEE Texas Power and Energy Conference (TPEC), 1–6. doi:10.1109/TPEC56611.2023.10078472

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. nature, 521(7553), 436-444.

Linardatos, P., Papastefanopoulos, V., & Kotsiantis, S. (2020). Explainable ai: A review of machine learning interpretability methods. Entropy, 23(1), 18.

Rist, Lukas. (2015, September 9). The honeynet project. The Honeynet Project. https://www.honeynet.org/2015/09/09/gas-tank-monitoring-system-honeypot/

Nowak, Kathleen E., Brandi, Juan M., Hofer, William J., Edgar, Thomas W., & Vrabie, Draguna L. (2021). Data-driven model generation for deception defence of cyber-physical environments. United States.

Pacific Northwest National Laboratory | PNNL. (n.d.). Pacific Northwest National Laboratory | PNNL. https://pnnl.gov/shadow-figment

Sahoo, S., Lampert, C., & Martius, G. (10--15 Jul 2018). Learning Equations for Extrapolation and Control. In J. Dy & A. Krause (Eds.), Proceedings of the 35th International Conference on Machine Learning (pp. 4442–4450). Retrieved from https://proceedings.mlr.press/v80/sahoo18a.html

Stouffer, K., Zimmerman, T., Tang, C., Lubell, J., Cichonski, J., & McCarthy, J. (2017). Cybersecurity Framework Manufacturing Profile. National Institute of Standards and Technology.

Taylor, B. J. (Ed.). (2006). Methods and procedures for the verification and validation of artificial neural networks. Springer Science & Business Media.

Yan, W., Mestha, L., John, J., Holzhauer, D., Abbaszadeh, M., & McKinley, M. (2018). Cyberattack Detection for Cyber Physical Systems Security – A Preliminary Study. Annual Conference of the PHM Society, 10(1). https://doi.org/10.36001/phmconf.2018.v10i1.508

Yang, Q., Liu, Y., Chen, T., & Tong, Y. (2019). Federated machine learning: Concept and applications. ACM Transactions on Intelligent Systems and Technology (TIST), 10(2), 1-19.

Yuill, J. J. (2006). Defensive Computer-Security Deception Operations: Processes, Principles and Techniques. North Carolina State University.

Pacific Northwest National Laboratory

902 Battelle Boulevard
P.O. Box 999
Richland, WA 99354

1-888-375-PNNL (7665)

www.pnnl.gov