CYBER-PHYSICAL DEFENSE IN SMART DISTRIBUTION NETWORKS

An Undergraduate Research Scholars Thesis

by

LEEN AL HOMOUD, SAFIN BAYES, AND RINITH REGHUNATH

Submitted to the LAUNCH: Undergraduate Research office at
Texas A&M University
in partial fulfillment of requirements for the designation as an

UNDERGRADUATE RESEARCH SCHOLAR

Approved by
Faculty Research Advisor: Robert S. Balog

May 2021

Major: Electrical Engineering

RESEARCH COMPLIANCE CERTIFICATION

Research activities involving the use of human subjects, vertebrate animals, and/or biohazards must be reviewed and approved by the appropriate Texas A&M University regulatory research committee (i.e., IRB, IACUC, IBC) before the activity can commence. This requirement applies to activities conducted at Texas A&M and to activities conducted at non-Texas A&M facilities or institutions. In both cases, students are responsible for working with the relevant Texas A&M research compliance program to ensure and document that all Texas A&M compliance obligations are met before the study begins.

We, Leen Al Homoud, Safin Bayes, and Rinith Reghunath, certify that all research compliance requirements related to this Undergraduate Research Scholars thesis have been addressed with my Research Faculty Advisor prior to the collection of any data used in this final thesis submission.

This project did not require approval from the Texas A&M University Research Compliance & Biosafety office.

TABLE OF CONTENTS

ABSTRACT

Cyber-Physical Defense in Smart Distribution Networks

Leen Al Homoud, Safin Bayes, and Rinith Reghunath
Department of Electrical Engineering
Texas A&M University

Research Faculty Advisor: Robert S. Balog
Department of Electrical Engineering
Texas A&M University

The existing electric grid is transitioning to a smart grid with increased penetration of distributed energy resources (DERs), such as photovoltaic (PV) units, battery storage units, electric vehicles (EV), and EV chargers. DERs facilitate the increase in renewable energy generation, which leads to a more sustainable, efficient, and reliable grid paradigm. However, with the rise of communication exchanges and data flow due to DERs, cybersecurity vulnerabilities arise. Much of the literature has focused strictly on mitigating data attacks resulting in non-technical losses, false state estimation, and inaccurate load forecasting. However, the grid paradigm's cyber-physical security also needs to be taken into account to ensure that no grid operations take place that impact the physics of the system. Our project achieved that by developing a Machine Learning (ML) algorithm that will detect anomalies in the commands issued to the distribution network's assets. The algorithm was trained using data from a base case obtained from the simulation of the IEEE 34 distribution network. It was tested and improved by adding modifications to the base case. We successfully developed a local anomaly detection algorithm for a photovoltaic system and two voltage regulators, achieving F1-

scores of 0.5141, 0.8173, and 0.8982, respectively. All three algorithms achieved low values of false negatives, which is promising as false negatives have a much higher cost since missing one anomaly can result in disastrous effects on the entire grid.

ACKNOWLEDGEMENTS

Contributors

We would like to thank our faculty advisor, Dr. Robert S. Balog, for his continued guidance and support throughout the course of this research. We would also like to thank Engr. Sawsan Shukri and Engr. Aaqib Peerzada for their supervision and help in advancing this research project.

The irradiance and temperature profiles analyzed/used for Cyber-Physical Defense in Smart Distribution Networks were provided by the Qatar Environment and Energy Research Institute (QEERI).

All other work conducted for the thesis was completed by the students independently.

Funding Sources

No funding for this thesis was received.

1. INTRODUCTION

The current electric grid has been designed for unidirectional power and data flow from large synchronous electric generators to consumers [1]. With the increased implementation of distributed energy sources (DERs), such as photovoltaic (PV) units, battery storage units, electric vehicles (EV), and EV chargers [2], the inevitable need for bidirectional power and data flow has risen. As a result, the current electric grid has been transitioning to Smart Grid (SG) to accommodate the continuously increasing penetration of DERs, which include renewable energy generation in the distribution network of the electric grid [1].

As with every new rising technology, benefits are accompanied by challenges that need to be identified, detected, and mitigated to ensure a trustworthy final product. With the increased need for communication flow and data exchange due to the implementation of DERs, new cybersecurity vulnerabilities arise. For example, false data attacks are a significant threat that cause severe consequences [3]. One instance of false data attacks could be false data injection, where the attacker would hijack communication channels and lead to false state estimation and non-technical losses [3], such as energy theft, unmetered supplies errors, and conveyance losses [4]. One thing of utmost importance needs to be highlighted, which is the fact that cyberattacks can hinder grid operations that are governed by the laws of physics relating to the grid power flow, and not just information exchanges [6]. For instance, a data attack on the DERs, generators, or loads can interrupt power flow on the grid and result in severe physical damage [5]. As a result, a cyber-resilient power system that can detect and mitigate cyber-physical attacks is needed.

Previous literature has focused on developing algorithms to monitor grid behavior and detect discrepancies in grid performance due to cyberattacks. To detect non-technical losses, many papers have utilized supervised machine learning algorithms such as support vector machine (SVM), k-nearest neighbor (k-NN), decision tree (DT), regression-based, and artificial neural networks (ANN) [3]. Some papers used unsupervised learning algorithms, such as the gradient boosting classifier (GBC) and clustering-based algorithms [3]. Additionally, some papers have used deep learning techniques such as convolutional neural network (CNN) and a novel detection model called MFEFD [3]. Other authors have taken advantage of phasor measurement units (PMUs), which are measurement devices that can provide real-time data about the electric grid status [6]. Chamie et. al [6], who has used Micro-PMU measurements, focused on creating an anomaly detection algorithm that can detect data attacks in grid-edge devices. A pseudo-supervised learning (PSL) algorithm is used, where isolation forests is used as an unsupervised learner first, and then, nonlinear regressors is used as a supervised learner. Pandey et. al [7] created a detection algorithm that detects and classifies anomalies into three events; active power, reactive power, and fault events. Pandey et. al [7] has used statistics, clustering, maximum likelihood criterion (MLE), and density-based spatial clustering (DBSCAN) for event detection, and physics-based rule and decision tree for event classification. Deng et. al [8] concluded that one way to detect false data injection attacks in Smart Grids is to protect meter measurements from being manipulated by using PMUs.

As seen from the extensive literature survey, much of the literature has focused on detecting data attacks and mitigating consequences instead of detecting calculated, intelligent, and sophisticated cyber-attacks aimed at harming and disrupting the grid's cyber-physical security. Little work has been done to detect cyber-attacks while considering the grid's physical nature [3], [6], [8]. As a result, it is imperative to address the cybersecurity concerns holistically [6]. That can only be achieved if work is specifically done and focused on the cyber-physical consequences of attacks.

Therefore, we are proposing an algorithm to enhance the cyber-physical resiliency of the grid. Our project aims to detect cyber-attacks on a smart distribution network that consists of various controllable assets, such as, capacitor banks, DERs, bus voltage regulators etc. We are focusing on sustained, intelligent, and coordinated attacks which can cause accelerated harm to the network's assets. Our detection scheme is based on a Machine Learning (ML) algorithm which will detect anomalies in the commands issued to the assets' controller.

2. METHODS

2.1 Modeling

2.1.1 PV System

To account for the effect of distributed energy resources (DERs) in the IEEE 34 System, a photovoltaic (PV) system was modeled. A PV system converts solar energy to electrical energy using one or more solar panels that collect the solar power, an inverter, and other hardware. The PV system modeled in OpenDSS comprises the PV array, an inverter, and a Norton Equivalent circuit. When a PV array collects solar energy, it transforms that solar energy into an electrical current that enters the inverter, entering the Norton Equivalent circuit to convert the current source into the required voltage source.

1. Essentially, the properties needed to define this system can be divided into three major groups:
    - 1.1. Array properties:
        - 1.1.1. Pmpp: power at the maximum power point.
        - 1.1.2. PV Correction Factor vs. Temperature curve.
    - 1.2. Inverter properties:
        - 1.2.1. kVA: Inverter's kVA rating.
        - 1.2.2. kV: Nominal rated voltage, which is the line-to-line voltage.
        - 1.2.3. Phases: Number of phases of the PV system.
        - 1.2.4. Bus1: Bus to which the PV system is connected to.
        - 1.2.5. Conn: Connection of the PV system (can be wye or delta).
        - 1.2.6. PF: Power factor of the output AC power.

1.2.7. %cutin: Percentage of the inverter's kVA rating. When the inverter is off, the power collected from the PV array has to be greater than this value to turn on the inverter.

1.2.8. %cutout: Percentage of the inverter's kVA rating. When the inverter is turned on, the power collected from the PV array has to drop below this value to turn the inverter off.

1.2.9. Inverter efficiency curve.

1.3. Operating Conditions properties:

1.3.1. Base irradiance in kW/m2.

1.3.2. Irradiance curve.

1.3.3. Base temperature in °C.

1.3.4. Temperature curve.

Table 2.1 below shows the values for the parameters for the base case simulation.

Table 2.1: Simulation Parameters (STC – standard test conditions).

Parameter	Value
Pmpp	200 kW
kVA	500 kVA
kV	24.9 kV
Phases	3
Bus1	840
Conn	Delta
PF	1
%cutin	0.1

%cutout	0.1
Base Irradiance	1 kW/m^2 (STC)
Base Temperature	25°C (STC)

Regarding the curves defined for the PV system model, each curve is determined differently. For the PV correction factor vs. temperature curve, a correction factor of 1 was set for the base temperature, which is 25°C. To calculate the correction factor, the following formula is needed:

$$Correction\ Factor = \frac{P(t)}{P_{STC}} \tag{2.1}$$

where P(t) is the power at a specific temperature and the base irradiance, and P_{STC} is the power at STC.

The power at each temperature (0, 75, and 100°C) needed to be calculated to use the formula above. Therefore, the correction factor calculations for these temperatures were based on a PV module datasheet [9], where the STC rated output Pmpp and the temperature coefficient of the PV module were needed. This is because these parameters are required to calculate the power at each temperature, using the formula below:

$$P(t) = P_{STC} \times (1 - C \times (T - T_{STC})) \tag{2.2}$$

where P(t) is the power at a specific temperature and the base irradiance, P_{STC} is the power at STC which is 390 W, C is the temperature coefficient which is -0.28%/C, T_{STC} is 25°C, and T is the temperature defined.

Using formulas 1 and 2, the values for the correction factor vs. temperature curve were determined, which are defined in **Table 2.2** below.

Table 2.2: Values for the Correction Factor vs. Temperature Curve.

Temperature	P(t)	Correction Factor
0°C	417.3 W	1.07
25°C	390 W	1
50°C	335.4 W	0.86
75°C	308.1 W	0.79

The inverter efficiency curve is defined as the inverter efficiency vs. the power in per unit. The inverter efficiency values were collected from the California Energy Commission [10]. These inverter efficiency values were for the ABB PVI-3.0-OUTD-S-US-A inverter, a 3 kW, 208 Vac grid support utility-interactive inverter with an arc detector. The values for this curve are shown in **Table 2.3** below.

Table 2.3: Values for the Correction Factor vs. Temperature Curve.

Power (per unit)	Inverter Efficiency
0.1	93.3%
0.2	95.9%
0.3	96.4%
0.5	96.4%
0.75	96.2%
1	95.8%

Lastly, the temperature and irradiance profiles were obtained from the Qatar Environment and Energy Research Institute (QEERI). Both are yearly profiles based on data collected from

2017. The data obtained from QEERI was based on data collection during the day, which meant that the irradiance and temperature values at night were not in the dataset. As such, data preprocessing was required to fill in for the missing values. This is because OpenDSS takes in the profiles with values for every minute of the year. A MATLAB script was developed to fill in 0 for the missing irradiance values, and for the temperature, the missing values were filled in by the last value of temperature that was collected before the missing data points.

2.1.2 Storage Unit

We also included a storage element as one of the Distributed Energy Resources (DERs) in our distribution network model. The storage element is a battery element that performs energy storage in the power grid. When the energy level drops in the grid, the storage element is activated and provides the needed power.

1. The different components of the storage element model in OpenDSS are the following,
2. Ideal Storage: This component signifies ideal, lossless energy storage.
3. Charging and Discharging Losses: This component corresponds to the losses that occur due to conversion from the storage medium to electrical energy and the converse.
4. Inverter: This component is a piece of built-in equipment in the model. It dispatches reactive power, models the inverter losses, and limits the rate of charge and discharge based on its ratings.
5. State: This component signifies the state in which the storage element functions. There are three states for a storage element, which are charging, discharging, and idling
6. Idling Losses: This component of the storage element represents storage self-depletion losses, the losses when the element is in the idle state.

As mentioned by the ‘State’ component of the model, the storage element operates between three possible states. It can either be charging, discharging, or idle. Depending on the state, the storage element is connected to the grid differently. During the charging state, the storage element acts as a constant power load to the grid. During the discharging state, it acts as a generator that provides power to the grid. During the idle state, the storage element is modeled as being disconnected from the grid.

We modeled the storage element in our distribution network to be connected to the network at bus 840. The other parameters necessary to model the storage element in OpenDSS are mentioned in **Table 2.4** below, along with the values we chose for our distribution network model. We chose our values after running some test runs and observing the network's power flow without the storage element.

Table 2.4: OpenDSS Modelling Parameters for Storage Element.

Parameter	Definition	Value
kWrated	kW rating of inverter active power output.	1000 kW
%charge	Charging rate, given by a percentage of kWrated.	100%
%discharge	Discharging rate, given by a percentage of kWrated.	100%
kWhrated	The rated storage capacity in kWh.	50,000 kWh
%stored	Amount of stored energy present in the element.	100%

%reserve	Amount of kWhrated to be reserved for normal functioning.	10%

We modeled a storage controller for our storage element in the network for controlling the dispatch of the storage element. The controller monitors a terminal of one of the components in our network and measures a quantity, which can be the power or current flowing at that terminal. This measured value is then compared to a target value, and the controller dispatches the storage element to operate based on the difference between the measured and target values.

There is a wide range of charging and discharging modes available for us to choose from when designing our storage controller. The discharging mode for our storage element is called the 'PeakShave' mode. We set a target value (in kW), and the controller requests the storage element to dispatch to maintain the power in the monitored element at or below the target value. The storage element also turns off when it runs out of stored energy or reaches its reserve value. The charging mode for our storage element is called the 'PeakShaveLow' mode. Similar to the discharging mode, we have a target value (in kW), and the controller requests the storage element to dispatch to maintain the power in the monitored element at or above the target value.

The parameters and the values chosen for the storage controller model are given in **Table 2.5**. We chose our values after running some test runs and observing the power flow in the monitored element without the storage element.

kW.

Table 2.5: OpenDSS Modelling Parameters for Storage Controller.

Parameter	Definition	Value
Element	Circuit element monitored by the controller.	Line 1 (Bus 800-Bus 802).
Terminal	The terminal of the circuit element being monitored, can be 1 or 2.	1
modedis	Mode of action for the discharge operation.	PeakShave
kWtarget	The target for discharging. The storage element is dispatched to keep the power in the monitored terminal at or below kWtarget.	1,600 kW
modcharge	Mode of action for the charging operation.	PeakShaveLow
kWtargetLow	The target for charging. The storage element is dispatched to keep the power in the monitored terminal at or above kWtargetLow.	800 kW

The storage controller monitors the power delivered by the substation. Given the parameter definitions, the storage controller monitors terminal 1 of the line between Bus 800 and Bus 802. When the terminal's power goes beyond 1600 kW, the storage element is activated and discharges to keep the power at or below 1600 kW. Similarly, when the terminal's power goes below 800 kW, the storage element is activated and charges to keep the power at or above 800 kW.

Figures 1-4 display the effect of the storage element and controller in the distribution network.

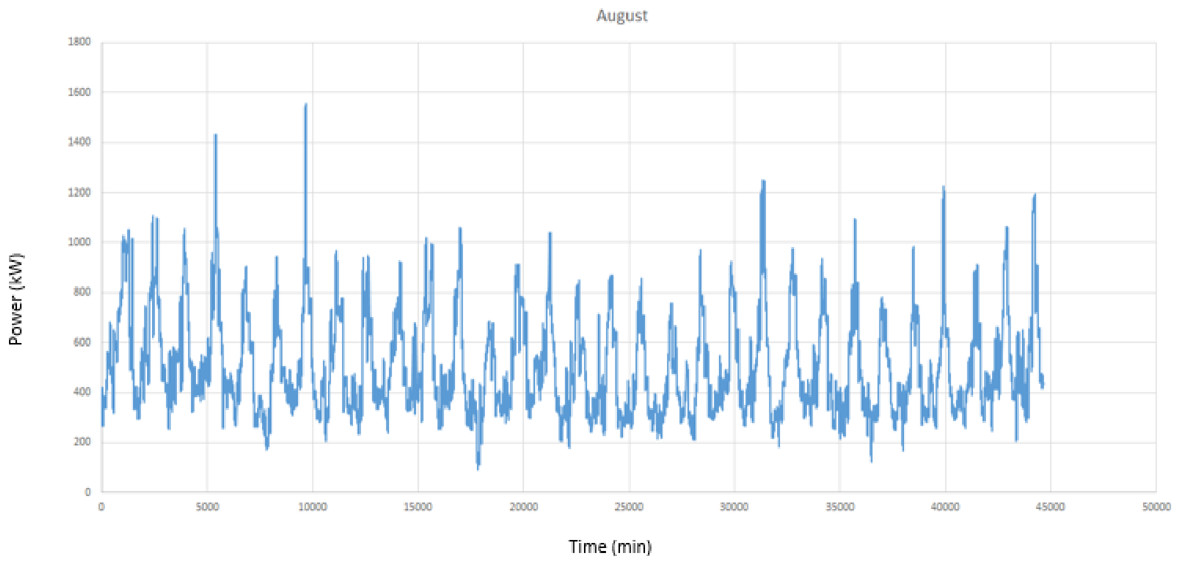


Figure 1: Power Flow from Substation in August (without Storage)

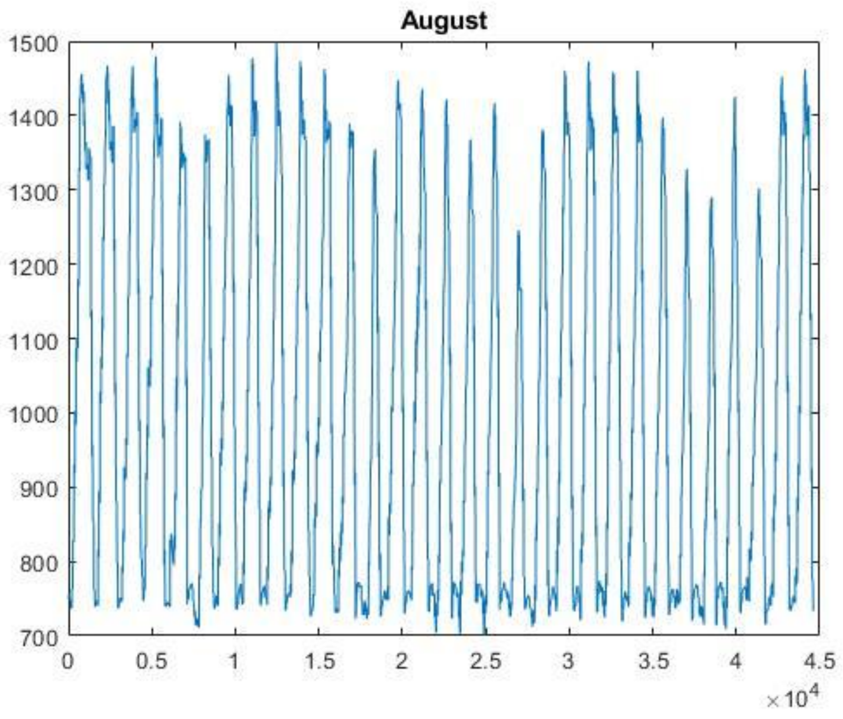


Figure 2: Power Flow from Substation in August (with Storage)

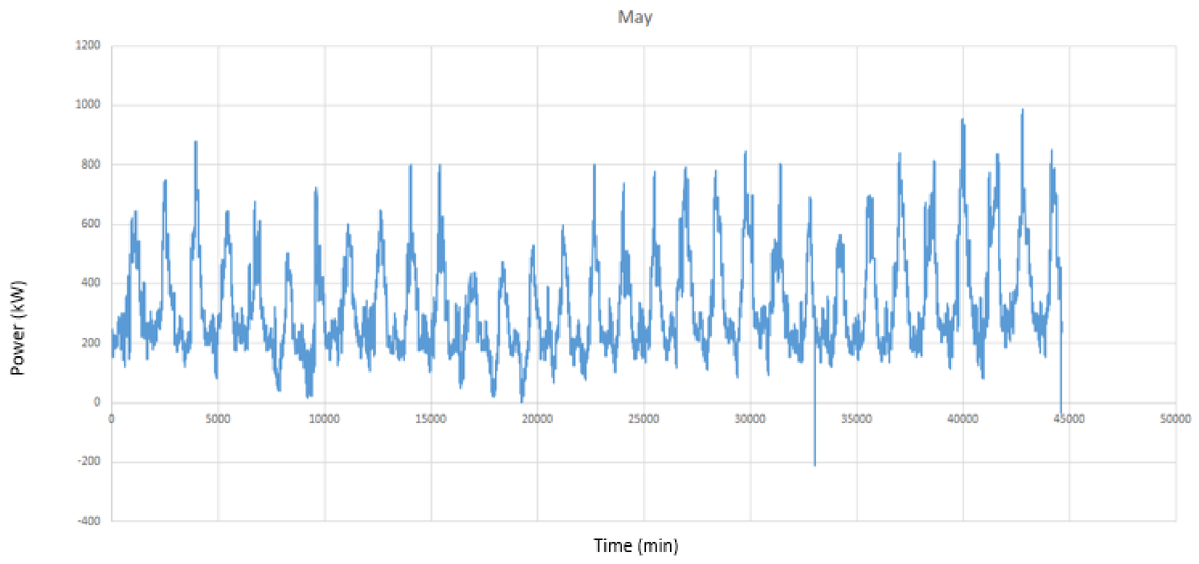


Figure 3: Power Flow from Substation in May (without Storage)

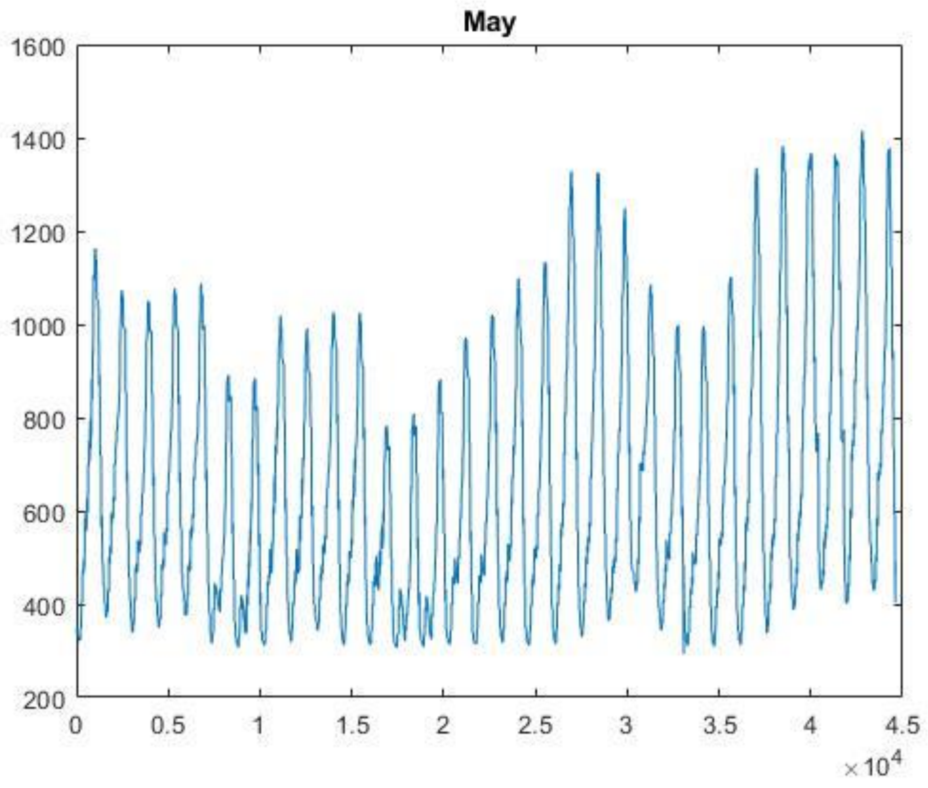


Figure 4: Power Flow from Substation in May (with Storage)

As it can be seen, the power flowing from the substation is maintained at the target values we specified in **Table 2.5**. The storage element is activated by the storage controller and dispatched to maintain the power flow at the target values.

2.1.3 Voltage Regulators

A voltage regulator is a device which generates a constant voltage of a preset value. There are two voltage regulators on the IEEE 34 System at buses 814 and 852. In OpenDSS, voltage regulators are modelled using the “Transformer” objects and controlled with the “RegControl” objects. The main parameters of the “Transformer” object in OpenDSS are as follows:

1. Phases: Number of phases.
2. Windings: Number of windings.
3. Bus: The bus number to which the winding is connected.
4. Kva: Apparent power rating of the winding.

The “RegControl” object is attached to a particular winding of a transformer. It adjusts the transformer taps in that winding according to the control settings. The main parameters of the “RegControl” object are as follows:

1. Transformer: The transformer to control.
2. Winding: The winding of the transformer to control.
3. Vreg: Voltage regulator setting, in Volts, for the winding being controlled.
4. Ptratio: The factor converting the winding voltage to the regulator voltage.

As the IEEE 34 System is an unbalanced power system, especially with other DERs involved, the voltage per phase tends to have different values, ergo, they require independent regulation. For this reason, we modelled an individual “Transformer” and “RegControl” pair for each phase of the regulator. Thus, for each regulator, we have three “Transformer” - “RegControl” pairs.

Based on the IEEE 34 System datasheet [11], **Table 2.6** and **Table 2.7** show the values for “Transformer” and “RegControl” objects for the regulators at buses 814 and 852 respectively.

Table 2.6: OpenDSS Modelling Parameters for Regulator at Bus 814.

Transformer Object		RegControl Object	
Parameter	Value	Parameter	Value
Phases	1	Transformer	Regulator 1
Windings	2	Winding	2
Bus	814	Vreg	122
Kva	2000	ptratio	120

Table 2.7: OpenDSS Modelling Parameters for Regulator at Bus 852.

Transformer Object		RegControl Object	
Parameter	Value	Parameter	Value
Phases	1	Transformer	Regulator 2
Windings	2	Winding	2
Bus	852	Vreg	124
Kva	2000	ptratio	120

2.1.4 Capacitors

The two capacitors in the IEEE 34 System are situated at buses 846 and 848 respectively. In OpenDSS capacitors are modelled using the “Capacitor” object. The main parameters for this object are as follows:

1. Bus: Bus connection for the first terminal of the capacitor.
2. Phases: Number of phases.
3. Kvar: Rated apparent power rating of the capacitor.

Table 2.8 shows the values of the parameters for the capacitors on the network based on the IEEE 34 System datasheet [11].

Table 2.8: OpenDSS Modelling Parameters for Both Capacitors.

Parameter	Value	
	Capacitor 1	Capacitor 2
Bus	846	848
Phases	3	3
Kvar	300	450

Capacitors are controlled using the “CapControl” object. It monitors the voltage and current at the capacitor’s terminal and sends switching messages to a “Capacitor” object. The following are the main parameters of this object:

1. Capacitor: Name of the capacitor to control.
2. Type: Control type (Current, Voltage, Time, Power Factor).
3. ONsetting: Value at which the control arms to switch the capacitor ON.
4. OFFsetting: Value at which the control arms to switch the capacitor OFF.

The IEEE 34 System datasheet has no information about capacitor controls. Therefore, we modelled the “CapControl” objects based on our observations. Furthermore, “CapControl” objects can only be placed downstream of the capacitor. Since there is no space downstream the capacitor at bus 848, we only modelled the controller for the capacitor at bus 846.

Our approach to model the capacitor control is to try different settings for the parameters and check which set of parameters yields bus voltages which are within the American National Standards Institute (ANSI) limits, i.e, the per unit voltage of the buses must be within the 0.95 and 1.00. By observing the plot in **Figure 5**, we set the ONsetting value to 14,800 V and OFFsetting to 14700 Volts. After carrying out a trial and error process of acquiring per unit bus voltages within the ANSI limits, we found the best ONsetting value at 14,600 V and OFFsetting value at 14,300 Volts. This means, the capacitor turns on and injects reactive power when the voltage at bus 846 is above 14,600 V and turns off when the voltage at bus 846 falls below 14,300 V.

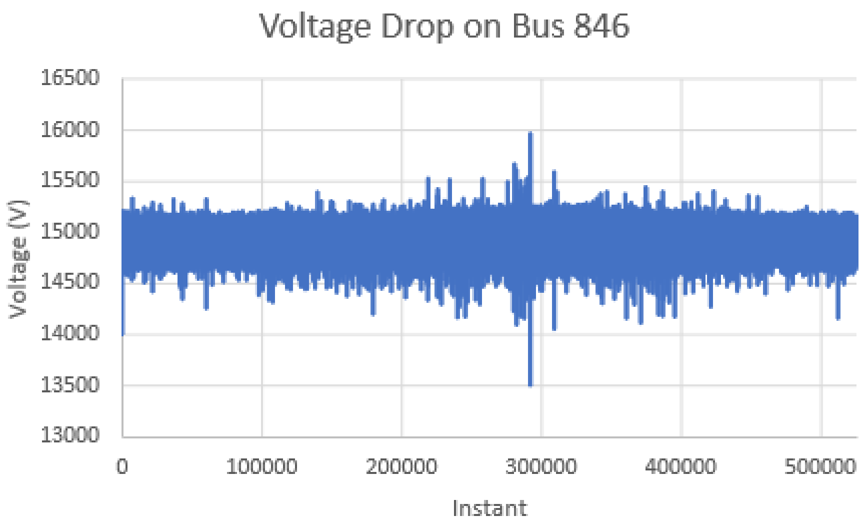


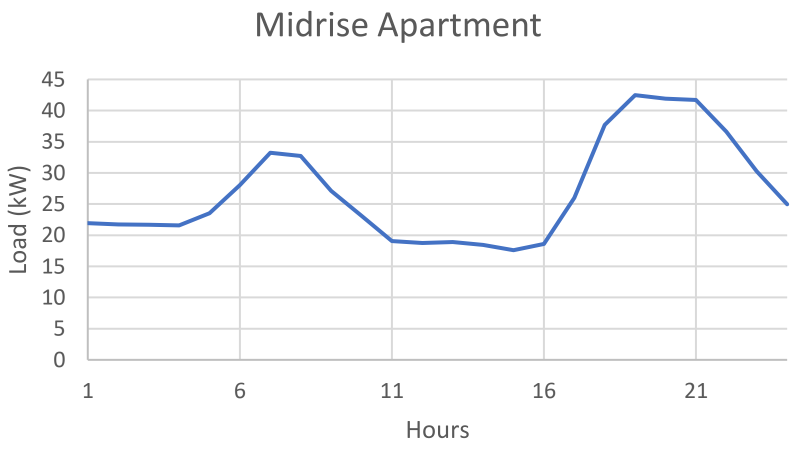
Figure 5: Variation of Voltage at Bus 846 throughout an annual simulation without capacitor control

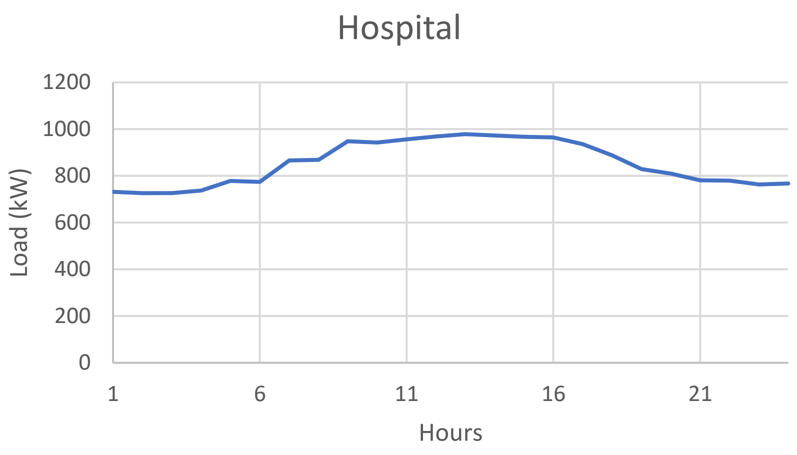
2.1.5 Load Modeling

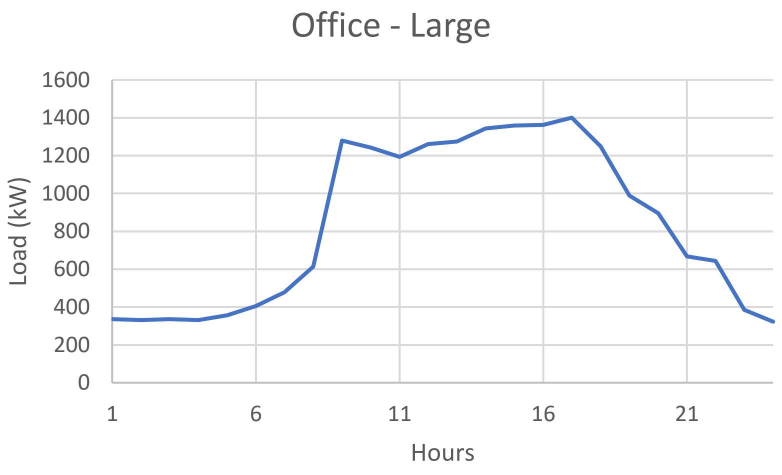
The loads are modelled based on the details provided by the IEEE 34 System datasheet [11]. However, there are no load profiles associated with the loads in the datasheet. To run a time-series simulation, OpenDSS requires a load profile for each load model to simulate the behavior of the load with respect to time.

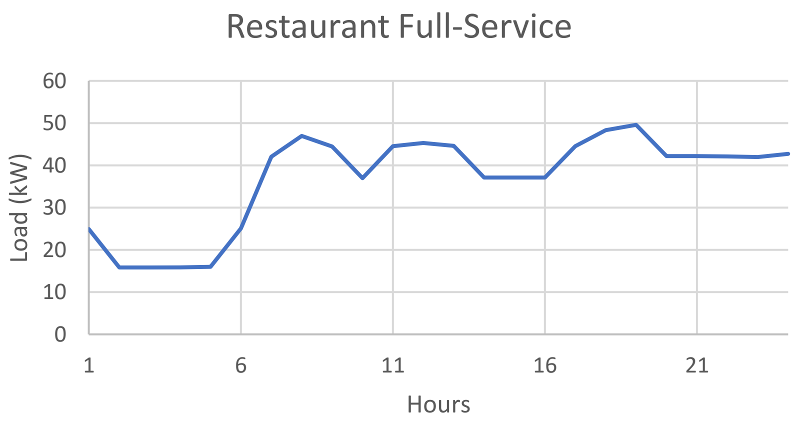
We used a webtool, called REopt Lite, developed by the National Renewable Energy Laboratory (NREL) [12], to obtain the load profiles. The tool generates an annual load profile in intervals of one hour for a given climate zone and building type. There are 16 climate zones in the webtool and we chose the one represented by Phoenix, Arizona. This is because among all the representative cities, the weather of Arizona is similar to that of Doha, Qatar. Among the 17 building types, we picked five - Midrise Apartment, Hospital, Office, and Restaurant. The other building types yielded load profiles which have the shape similar to at least one of the aforementioned five building types.

Figure 6 shows the daily load profiles of the five chosen building types for our system. The load profiles are randomly assigned to the load models of the IEEE 34 Bus System in OpenDSS.




Hospital
1200
1000
800
600
400
200
0
Load (kW)
1
6
11
16
21
Hours


Office - Large
1600
1400
1200
1000
800
600
400
200
0
Load (kW)
1
6
11
16
21
Hours


Restaurant Full-Service
60
50
40
30
20
10
0
Load (kW)
1
6
11
16
21
Hours

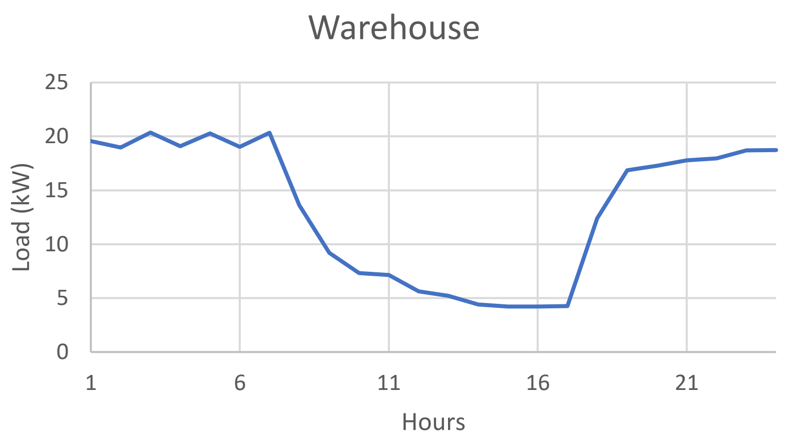


Figure 6: Load Profiles of the Representative Loads on the first 24 hours of the year.

2.2 Simulation

2.2.1 Base Case

The base case represents the model's behavior during normal operating conditions. We require a base case data to serve as the reference dataset for the rest of the research. The base case data is obtained using the algorithm described in **Figure 7** below.

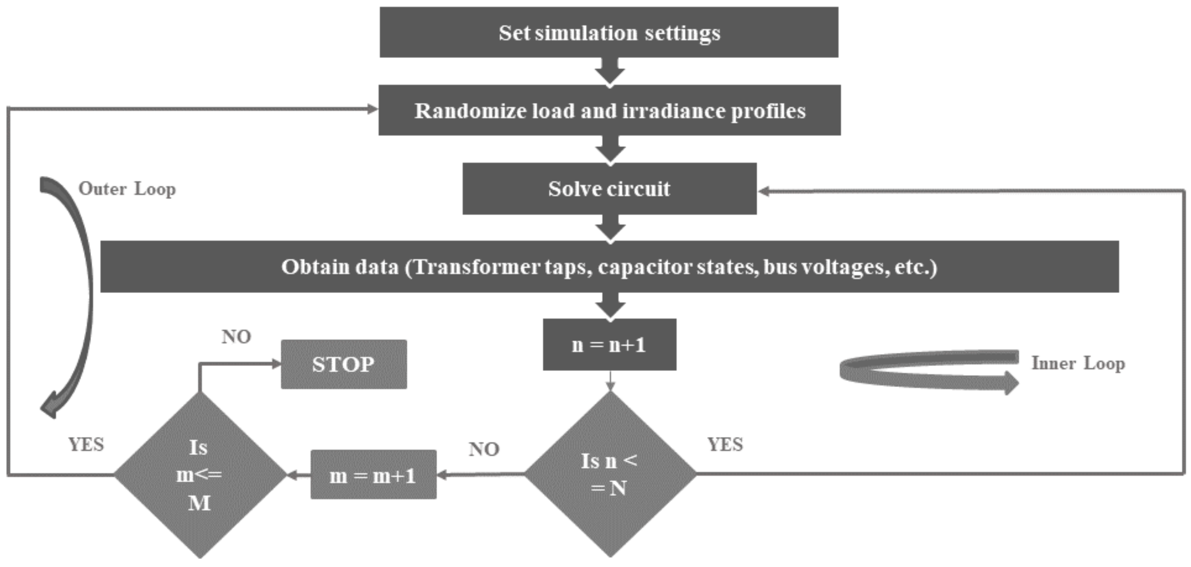

Set simulation settings
Randomize load and irradiance profiles
Solve circuit
Outer Loop
Obtain data (Transformer taps, capacitor states, bus voltages, etc.)
n = n+1
NO
STOP
Inner Loop
YES
Is m<= M
m = m+1
NO
Is n < = N
YES

Figure 7: Base Case Algorithm

Essentially, we run 100 Monte Carlo iterations of annual simulations where the time step in each annual simulation is one minute, corresponding to the time step of the irradiance and temperature profile of the PV system, which has the smallest time step resolution in the model. Thus, we have 525,600 (365 days * 24 hours * 60 minutes) time instants in one annual simulation.

At each Monte Carlo iteration, the load profiles are randomized to account for the varying load demands. The irradiance profile is also randomized to account for the random occurrences of dust and cloud transients. The randomization is done based on the approach described in the following sections.

Thus, the algorithm in **Figure 7** above has two loops - an outer loop and an inner loop. The inner loop solves the OpenDSS model and obtains the data at each time instant of the annual simulation defined as “n”. The inner loop stops execution when n = N = 525,600. The outer loop randomizes the load profiles and the irradiance profiles for each Monte Carlo iteration, “m”. The outer loop stops execution when m = M = 100.

2.2.1.1 Load Randomization

Load randomization is performed to account for the varying behavior of the load in each year. The approach for load randomization is as follows:

1. Select a probability distribution.
2. Sample random numbers from the distribution.
3. Multiply the points in the base profile with the random numbers obtained in step 2.

The idea behind the approach is to overestimate or underestimate the points in the base curve to obtain a randomized curve. However, we don’t want to create fluctuations by overestimating or underestimating the points to a very high extent. The randomized curve must still follow the shape of the base curve. Therefore, we must select a distribution with the following characteristics:

1. Underestimate or overestimate i.e. generate numbers greater than or less than one.
2. No negative numbers.

Based on the above characteristics, we selected the Burr distribution which is characterized by three parameters:

1. alpha (α): Scale parameter.
2. c: Shape parameter.
3. k: Second shape parameter.

To prevent exaggerating the overestimation and underestimation, we need to multiply the points in the base case with numbers close to one. Thus, the distribution must have higher probability for numbers closer to one. By adjusting the three parameters, we can make the distribution achieve this purpose. The Burr distribution with α=1, c =10, and k=1 gives us the best results.

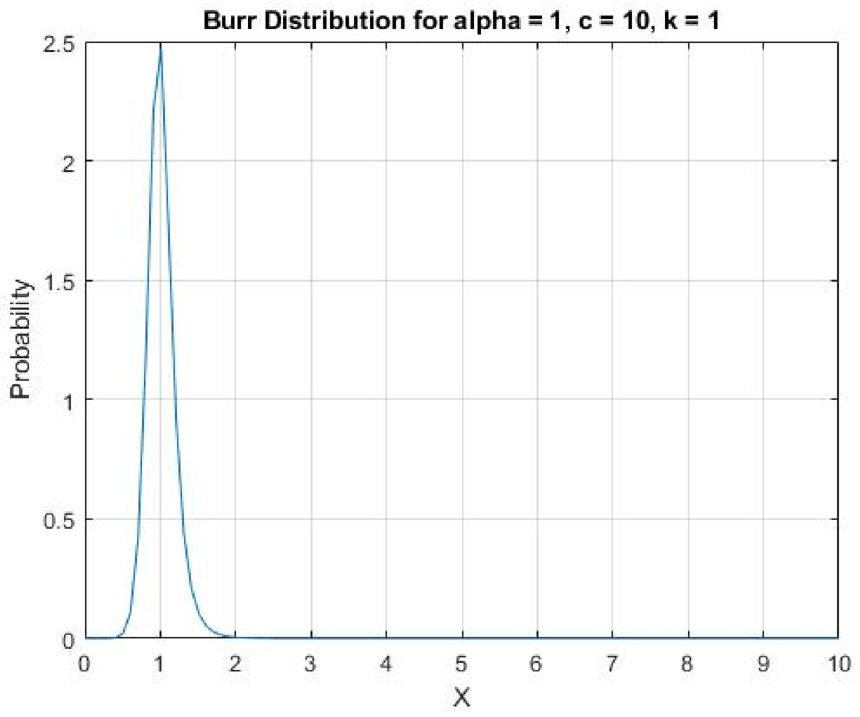


Figure 8: Burr Distribution with α=1, c =10, and k=1

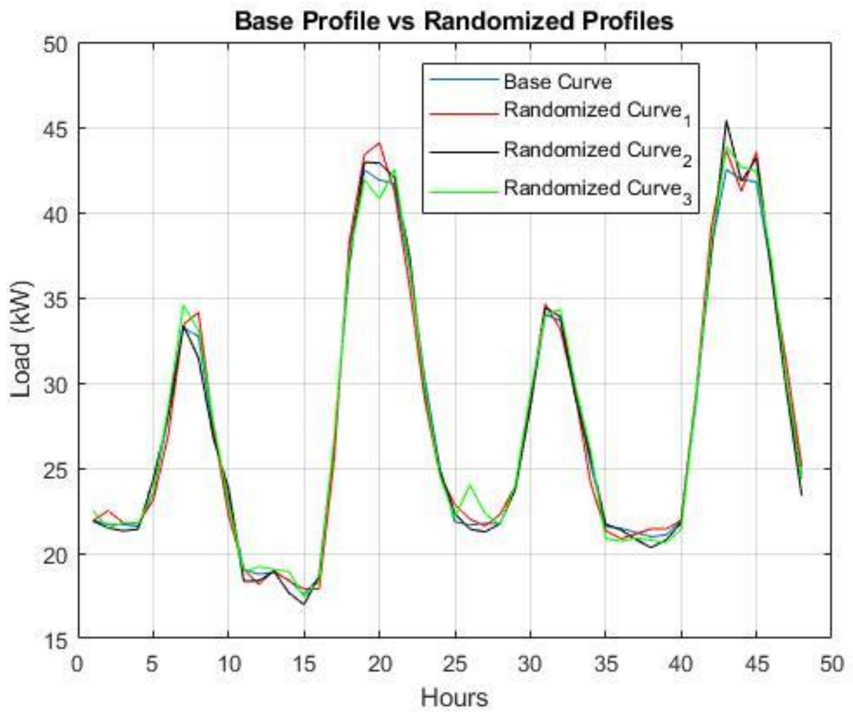


Figure 9: Base profiles and Randomized Profiles

2.2.1.2 Irradiance Randomization

Similar to the load randomization, the irradiance needed to be randomized to account for the varying irradiance. The logic behind irradiance randomization is very similar to that of the load randomization. We first define a probability distribution. Then, random numbers are sampled from the distribution and multiplied by the irradiance values in the base case dataset to obtain the randomized irradiance. Since the irradiance values obtained from QEERI are based on real-life collected data, the randomized irradiance should not be too different from the base case irradiance. Therefore, the random number sampled from the distribution is close to 1. Also, the irradiance randomization was performed for each month separately.

Lastly, we needed to figure out if the irradiance values should be overestimated or underestimated. This was determined by the irradiance value. It was decided empirically that any irradiance value above 900 W/m^2 is underestimated, and any values below that can be either underestimated or overestimated. Additionally, any irradiance values below 700 W/m^2 have to be overestimated. To achieve that, different probability distributions were selected. For the values above 900 W/m^2, a beta probability distribution was selected, which only underestimates. The two parameters for the beta distribution are both shape parameters, which are alpha (α) and beta (β). α was selected to be 10, and β was selected to be 1. To perform either underestimation or overestimation, a burr probability distribution was used, as done for the load randomization. The three parameters for the burr probability distribution are α, c, and k, which are the scale and two shape parameters, respectively. The best values for α, c, and k were determined to be 1.8, 6.5, and 50, respectively. To make sure that the values below 700 W/m^2 were always underestimated, an if statement in the MATLAB script for randomization was utilized to ensure that the random number generated is above 1, which ensures overestimation. Graphs of the beta and burr distributions are shown in **Figures 10-11**, respectively. **Figure 12** show plots of the base case irradiance vs. the randomized irradiance plots for all twelve months.

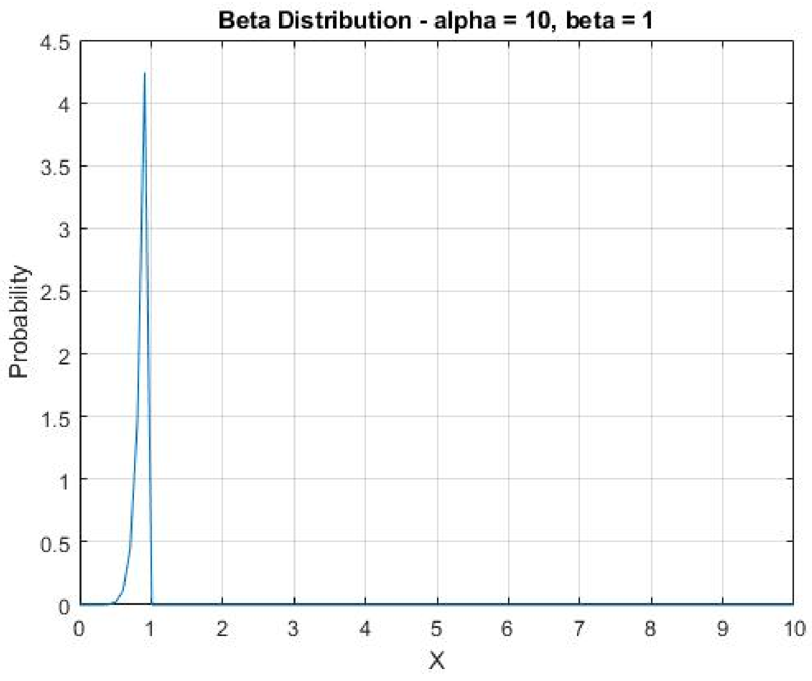


Figure 10: Beta Distribution (α=10, β=1)

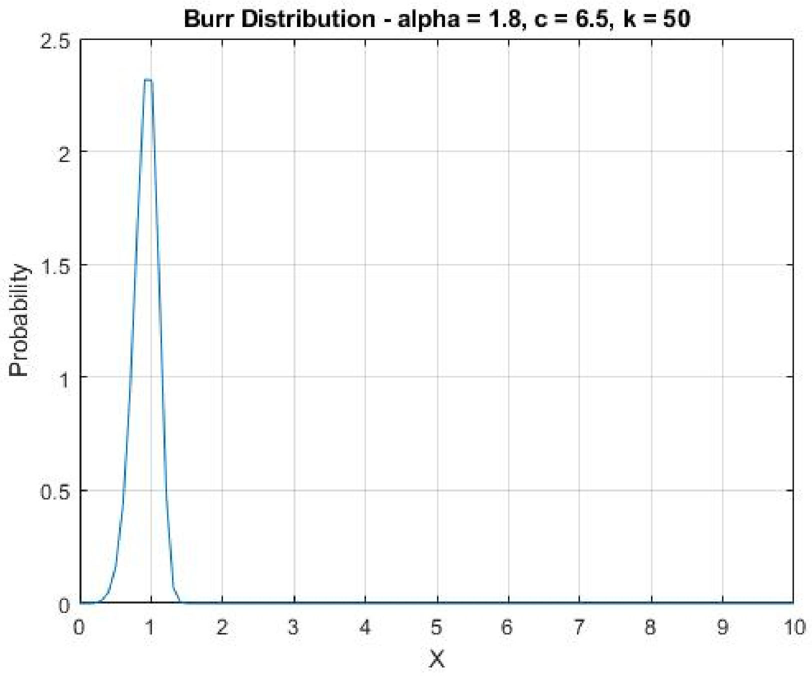
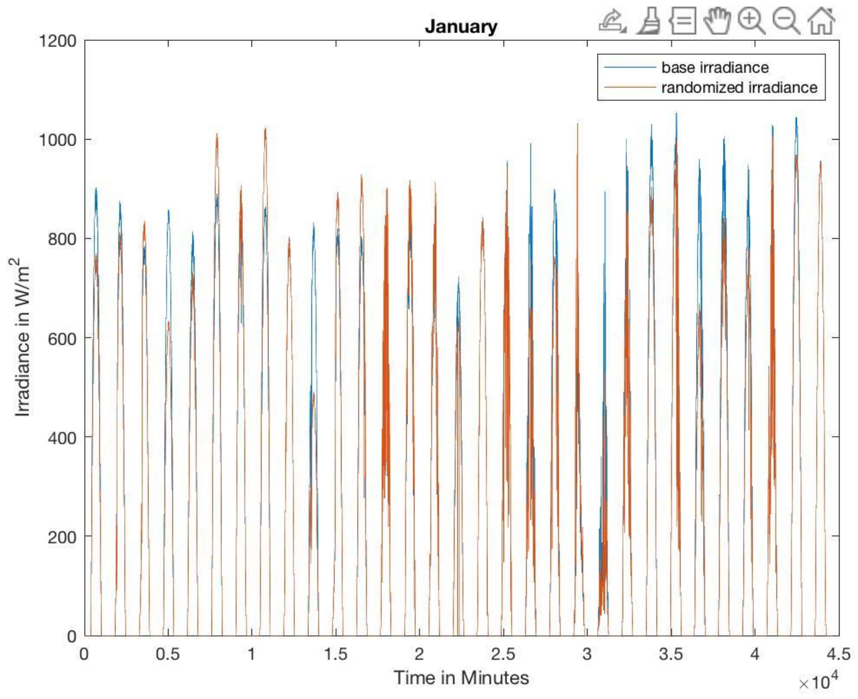
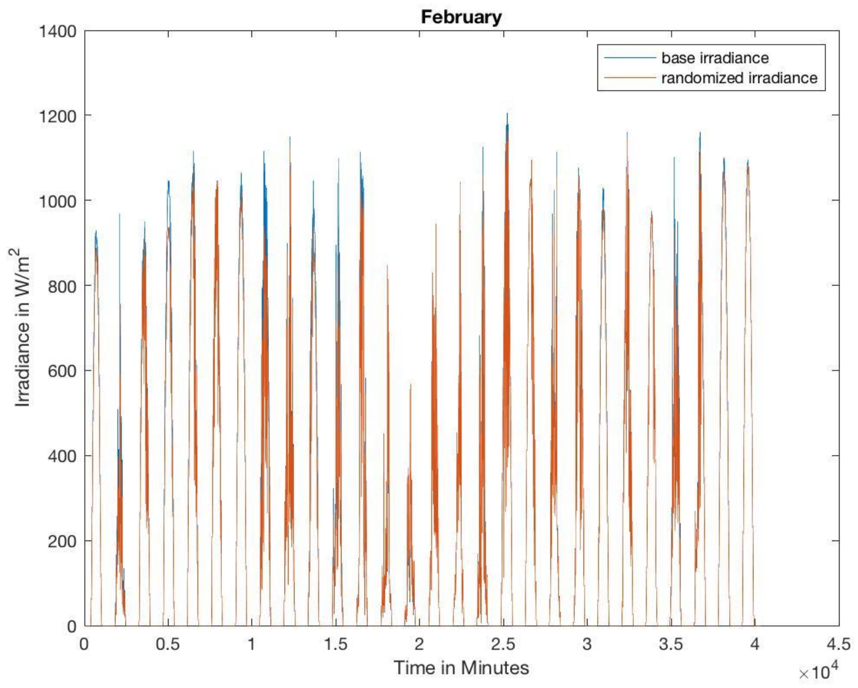
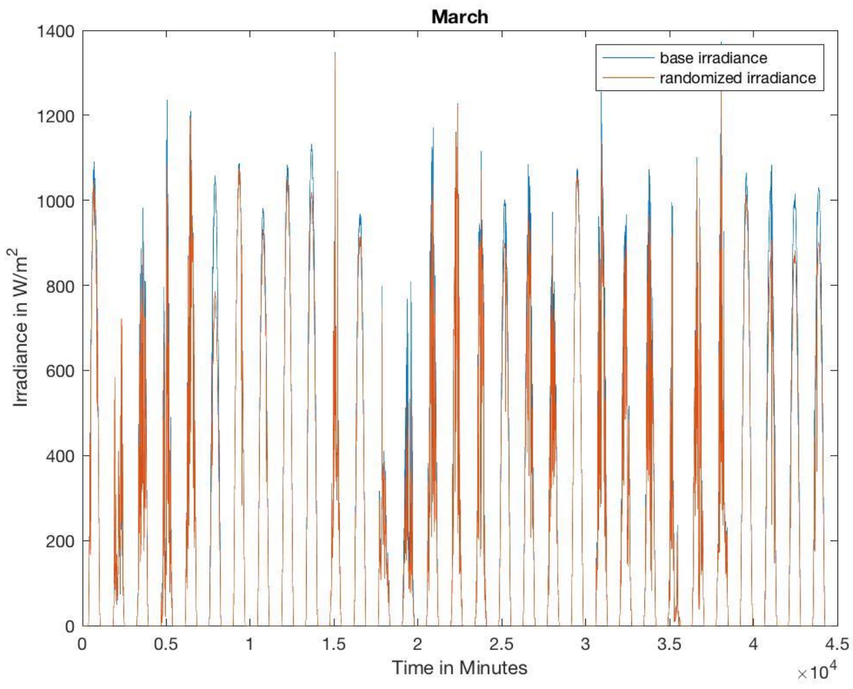
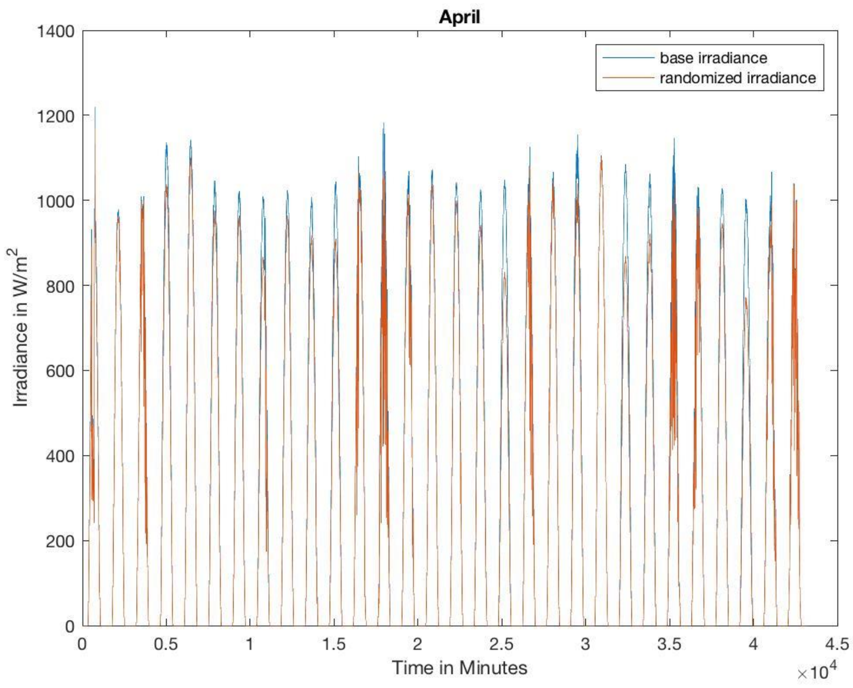


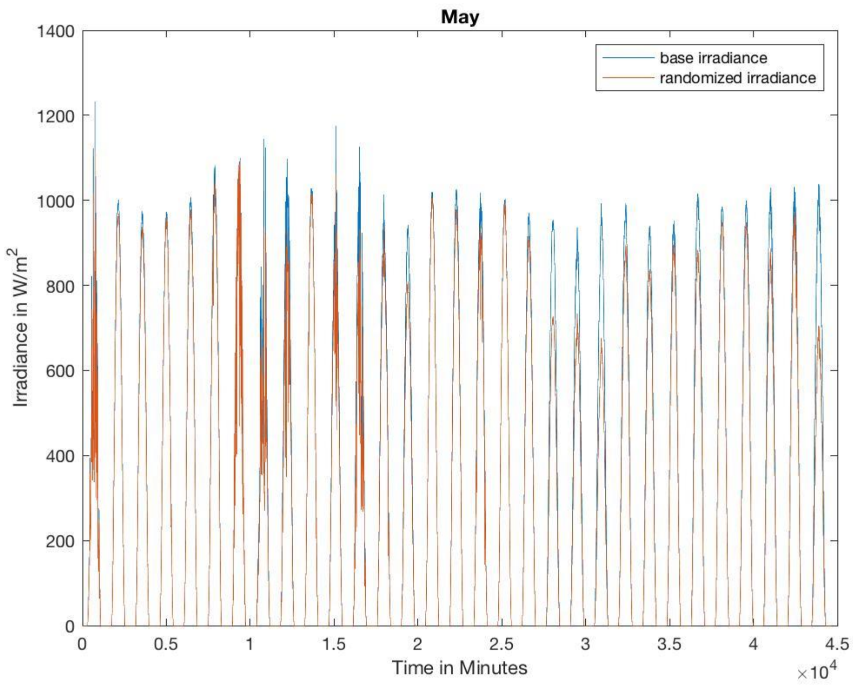
Figure 11: Burr Distribution (α=1.8, c=6.5, k=50)

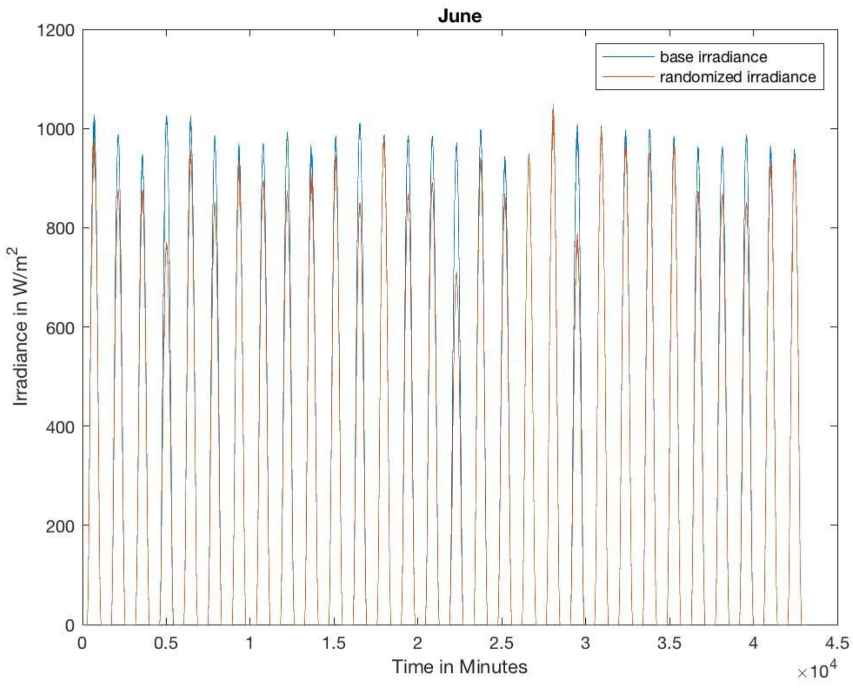

January
base irradiance
randomized irradiance
Irradiance in W/m^2
Time in Minutes
$\times 10^4$
0
200
400
600
800
1000
1200
0.5
1
1.5
2
2.5
3
3.5
4
4.5

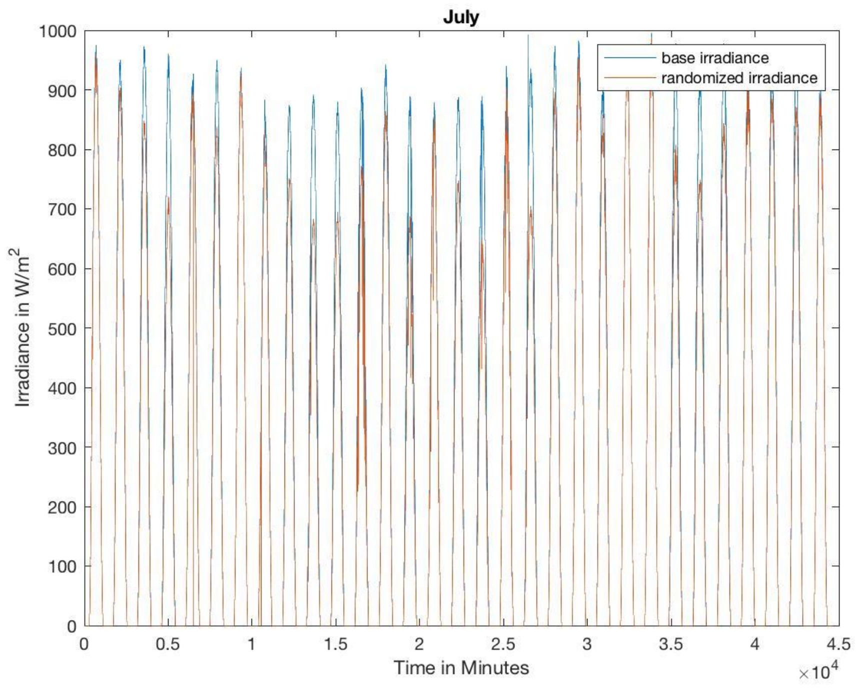

February
base irradiance
randomized irradiance
Irradiance in W/m^2
Time in Minutes
$\times 10^4$
0
200
400
600
800
1000
1200
1400
0.5
1
1.5
2
2.5
3
3.5
4
4.5

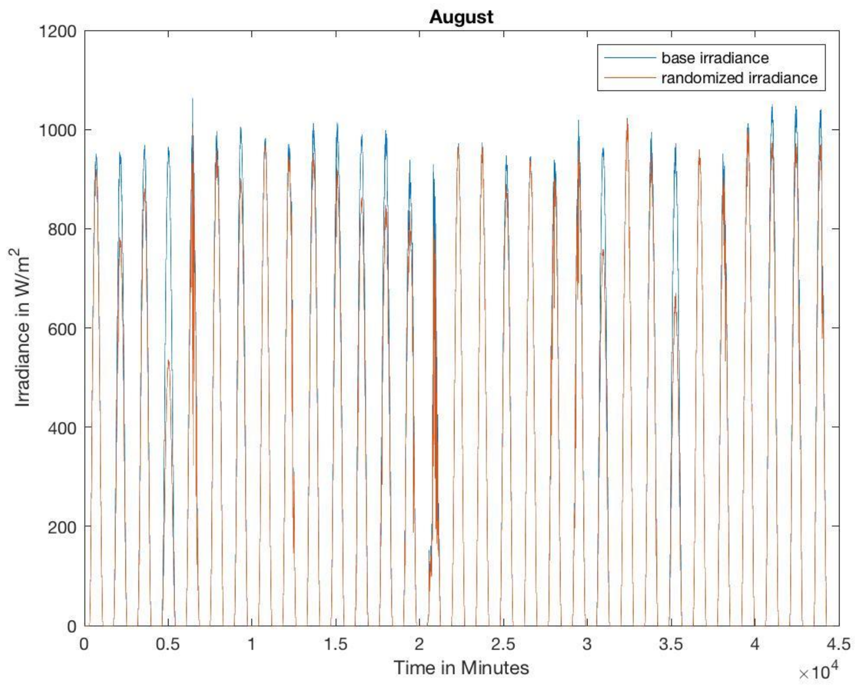

March
base irradiance
randomized irradiance
Irradiance in W/m^2
Time in Minutes
$\times 10^4$
1400
1200
1000
800
600
400
200
0
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5

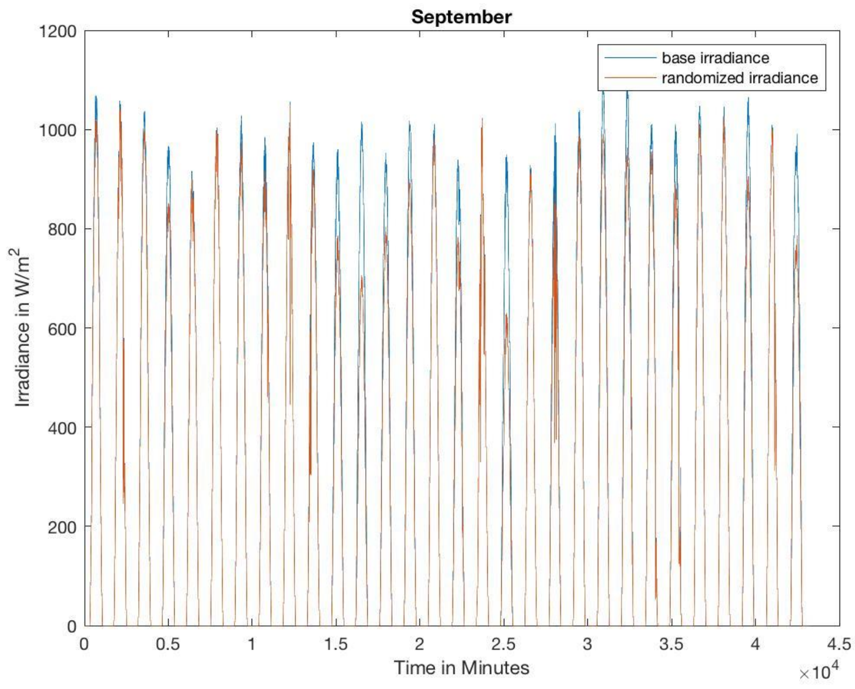

April
base irradiance
randomized irradiance
Irradiance in W/m^2
Time in Minutes
$\times 10^4$
1400
1200
1000
800
600
400
200
0
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5


May
base irradiance
randomized irradiance
Irradiance in W/m^2
Time in Minutes
$\times 10^4$

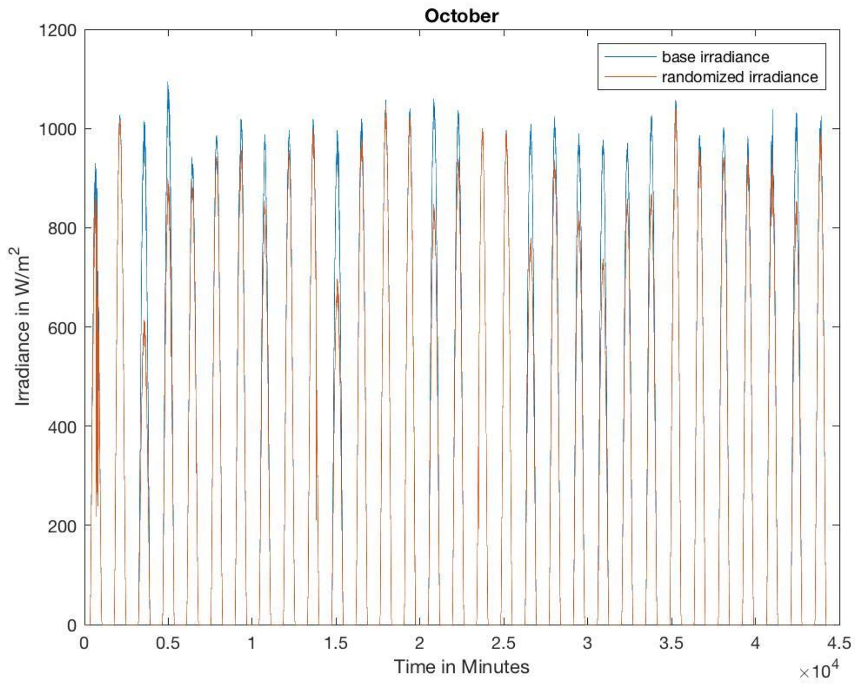

June
base irradiance
randomized irradiance
Irradiance in W/m^2
Time in Minutes
$\times 10^4$


July
base irradiance
randomized irradiance
Irradiance in W/m^2
Time in Minutes
$\times 10^4$

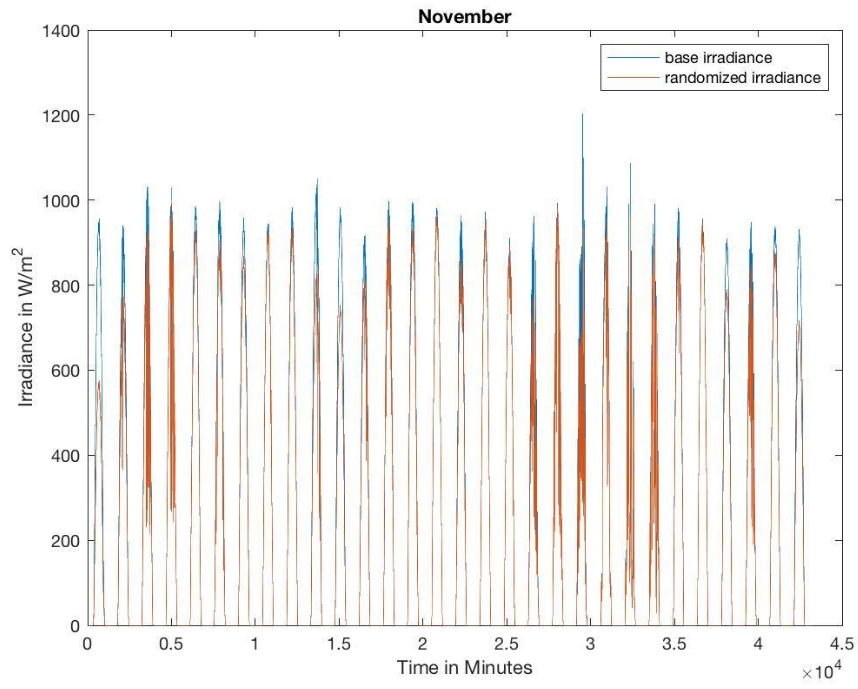

August
base irradiance
randomized irradiance
Irradiance in W/m^2
Time in Minutes
$\times 10^4$



September
base irradiance
randomized irradiance
Irradiance in W/m^2
Time in Minutes
$\times 10^4$
0
200
400
600
800
1000
1200
0.5
1
1.5
2
2.5
3
3.5
4
4.5



October
base irradiance
randomized irradiance
Irradiance in W/m^2
Time in Minutes
$\times 10^4$
0
200
400
600
800
1000
1200
0.5
1
1.5
2
2.5
3
3.5
4
4.5



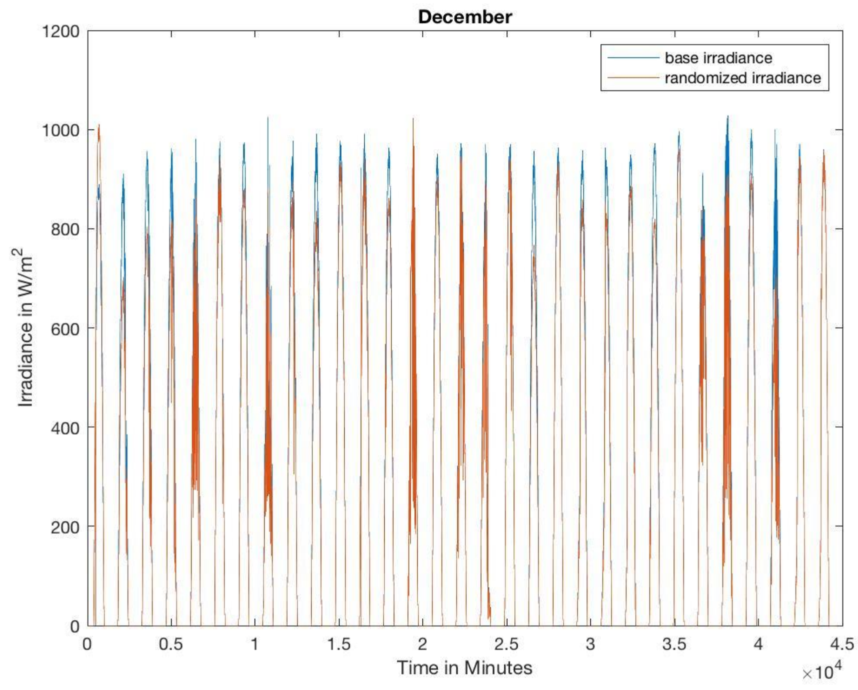


Figure 12: Base Case vs. Randomized Irradiance Plots

2.2.1.3 Running the Base Case

We connected our OpenDSS simulation with MATLAB to issue malicious control signals to the various assets of the distribution network and to save our circuit data more efficiently. We used the COM interface to enable a connection between OpenDSS and MATLAB. With the help of the COM interface, we are able to design our algorithm in another program such as MATLAB and then make OpenDSS do something that is not presently executed within the software.

A few lines of code were written to initialize OpenDSS on MATLAB after which we define MATLAB handles to interact with the OpenDSS objects. The circuit is then compiled and run, which in turn drives the OpenDSS software to solve the circuit.

OpenDSS offers a wide range of circuit data (power, current, voltages etc.) once solving the circuit. Depending on our usage of the network considering our project, we decided on collecting the following data from our network:

1. Bus voltages.
2. Capacitor – States, Reactive power.
3. PV – Irradiance, Real power.
4. Regulator 1 and 2 tap positions (all phases).
5. Storage – Capacity, kWIn (power flowing into the storage element), kWOut (power flowing out of the storage element), States.
6. Sub-bus Power.

Running 100 Monte Carlo iterations of annual simulations, we obtain a large amount of data which we saved into one folder with many sub-folders each corresponding to a single Monte Carlo iteration. Exporting these data directly from OpenDSS to a .CSV file in Excel is more time consuming and the data files will be larger in size. Therefore, we used MATLAB commands to store these data in .mat files.

2.2.2 Test Datasets

To create the test datasets, we would need to first issue malicious commands. The purpose of issuing malicious commands to the simulation is to observe how the system reacts to the malicious set points. This will help us identify the features that could be used in the machine learning algorithm and obtain the malicious data set.

The malicious set is obtained in two steps. First, we issue arbitrary malicious commands to the assets and then identify the system characteristics that deviate from the base case behavior. Then, we issue several hundred random commands which are passed through a test based on our observations that label a command as anomaly or normal.

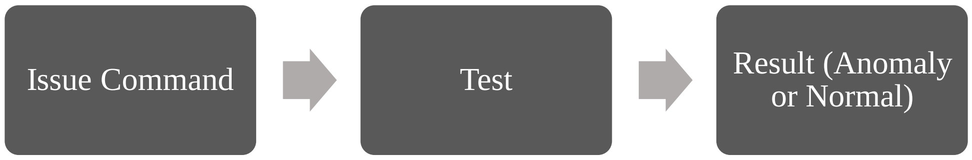


Figure 13: Procedure for Obtaining Malicious Dataset

2.2.2.1 Regulators

We performed various test cases by issuing commands at arbitrary time instants and observed changes on the other assets. One example of such a test case is as follows.

At instant 95600 (March 7, 9 AM), we issued a tap position of 4 at phase “a” of the regulator at bus 814 which we will refer to as Regulator 1. Then, we observed the states of each asset for the next 30 instants and compared them with the base case states. **Table 2.9** shows the observations.

Table 2.9: Base Case and Test Case Comparison for Regulator 1 and Regulator 2.

	Regulator 1 Tap Positions		Regulator 2 Tap Positions	
Instant	Base Phase a	Test Phase a	Base Phase a	Test Phase a
1	-7	4	3	3
2	-7	3	3	2
3	-7	2	3	1
4	-7	1	3	0
5	-7	0	3	-1
6	-7	-1	3	-2
7	-7	-2	3	-3
8	-7	-3	3	-2
9	-7	-4	3	-1
10	-7	-5	3	0
11	-7	-6	3	1
12	-7	-6	3	1
13	-7	-6	3	1
14	-7	-6	3	1
15	-7	-7	3	2
16	-8	-7	3	2
17	-8	-7	3	2
18	-8	-7	3	2
19	-8	-7	3	2
20	-8	-7	3	2
21	-8	-7	3	2
22	-8	-8	3	2
23	-8	-8	3	3
24	-8	-8	3	3
25	-8	-8	3	3
26	-8	-8	3	3
27	-8	-8	3	3
28	-8	-8	3	3

29	-8	-8	3	3
30	-8	-8	3	3

From **Table 2.9**, it can be seen that regulator 1 has switched its tap positions 10 more times than it did in the base case, while regulator 2 has changed its tap positions 8 more times than it did in the base case. Thus, an anomalous command causes an increase in the switching of the tap position. This increased switching of taps can wear out the regulator and decrease its lifetime over the long term.

Table 2.10: Base Case and Test Case Comparison for Capacitor.

Instant	Base Capacitor States	Test Capacitor States
1	1	0
2	0	0
3	0	0
4	0	0
5	0	0
6	0	0
7	1	1
8	0	0
9	0	0
10	0	0
11	0	0
12	0	0
13	1	0
14	0	0
15	0	0
16	0	1
17	0	0
18	0	0

19	1	0
20	0	0
21	0	0
22	0	0
23	0	1
24	0	0
25	1	0
26	0	0
27	0	0
28	0	0
29	0	0
30	0	1

From **Table 2.10** above, the capacitor in the test case switched its state eight times compared to nine times in the base case. Thus, the malicious command to the regulator one does not result in any anomalous behavior on the capacitor.

Other observations include the violations of ANSI voltage limits at some of the buses. We also saw no change in storage states.

To illustrate another example, this time we issued a tap position of -10 at phase “b” of the Regulator at bus 854 (Regulator 2) at instant 227062 (May 5, 4 PM). **Table 2.11 and Table 2.12** illustrate the results.

Table 2.11: Base Case and Test Case Comparison for Regulator 2.

	Base Case			Test Case		
Instant	Phase a	Phase b	Phase c	Phase a	Phase b	Phase c
1	3	4	2	10	-9	11
2	2	3	2	12	-8	12
3	3	4	2	12	-7	12

4	3	4	2	12	-6	12
5	3	4	2	12	-5	12
6	3	4	2	12	-4	12
7	3	4	2	12	-3	12
8	3	4	2	12	-2	11
9	3	4	2	11	-1	11
10	3	4	2	11	0	11
11	2	4	2	11	1	12
12	3	4	2	11	2	12
13	3	4	2	11	3	12
14	3	4	1	11	4	12
15	3	4	3	11	5	11
16	3	4	3	10	6	11
17	3	4	3	11	7	11
18	3	4	3	11	8	11
19	3	4	3	11	9	11
20	2	3	2	11	10	10
21	3	3	3	11	11	11
22	3	3	3	11	11	11
23	3	3	3	11	11	11
24	3	3	3	11	11	11
25	3	3	3	11	11	11
26	3	3	3	11	11	11
27	3	3	3	11	11	11
28	3	3	3	11	11	11
29	3	3	3	11	11	11
30	3	3	3	11	11	11

It can be seen that the number of taps switching in the test case is higher than in the base case. As determined earlier, issuing a malicious command to the regulator may cause it to wear out overtime and decrease its lifetime.

Table 2.12: Base Case and Test Case Comparison for Regulator 1.

	Base Case			Test Case		
Instant	Phase a	Phase b	Phase c	Phase a	Phase b	Phase c
1	-7	4	-8	6	-2	1
2	-7	4	-8	6	-2	1
3	-7	4	-8	6	-2	1
4	-7	4	-8	6	-2	1
5	-7	4	-8	6	-2	1
6	-7	4	-8	6	-2	1
7	-7	4	-8	6	-2	1
8	-7	4	-8	6	-2	1
9	-7	4	-8	6	-2	1
10	-7	4	-8	6	-2	1
11	-7	4	-8	6	-2	1
12	-7	4	-8	6	-2	1
13	-7	4	-8	6	-2	1
14	-7	4	-8	6	-2	1
15	-7	4	-8	6	-2	1
16	-7	4	-8	6	-2	1
17	-8	3	-8	6	-2	1
18	-8	3	-8	6	-2	1
19	-8	3	-8	6	-2	1
20	-8	3	-9	6	-2	1
21	-8	3	-9	6	-2	1
22	-8	3	-9	6	-2	1

23	-8	3	-9	6	-2	1
24	-8	3	-9	6	-2	1
25	-8	3	-9	6	-2	1
26	-8	3	-9	6	-2	1
27	-8	3	-9	6	-2	1
28	-8	3	-9	6	-2	1
29	-8	3	-9	6	-2	1
30	-8	3	-9	6	-2	1

In the previous example, issuing a malicious command to regulator 1 caused anomalous behavior in regulator 2, but not the other way around. Some other observations include the violation of ANSI limits at some buses. The capacitor states remain unchanged in the test case.

From these observations, we identified that the regulator tap positions and ANSI limit violations are the strongest indicators of a malicious command. In the next step, we issued random commands to the regulator and passed them through a test to label them as anomaly or normal. The flowchart in **Figure 14** shows the procedure.

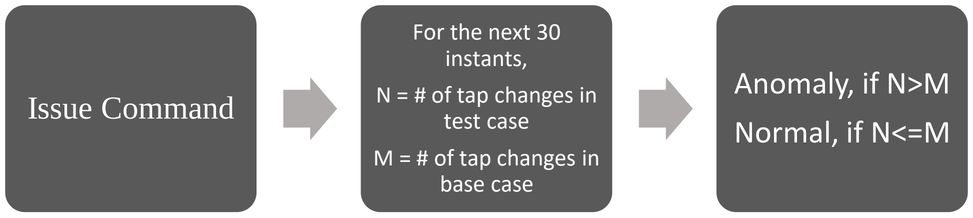

Issue Command
For the next 30 instants,
N = # of tap changes in test case
M = # of tap changes in base case
Anomaly, if N>M
Normal, if N<=M

Figure 14: Procedure for Obtaining Malicious Set for Regulators

2.2.2.2 Photovoltaic System

To create the malicious dataset for the PV system, a similar approach to issuing the commands for the regulators was followed. The command that was launched to the PV system involved changing the percentage of PV production, which is also known as curtailment. To perform curtailment, different commands with different percentages of PV production were issued, which included dropping the percentage of PV production by 0% (turning the PV system off), 30%, 50%, and 70% from its original PV production percentage in the base case. Each of these commands were issued at multiple instances throughout the year, which are:

1. January 1st at 12 PM (minute 699 out of the total 525,600 minutes in a year).
2. July 8th at 5 PM (minute: 271,686).
3. March 27th at 11 AM (minute: 123,046).
4. June 20th at 12 PM (minute: 245,493).
5. July 19th at 10 AM (minute: 287,123).
6. August 5th at 12 PM (minute: 311,719).
7. December 21st at 12 PM (minute: 510,481).

To observe the effect of the commands issued, we observed the following 30-minute time interval after the command is issued. We first started by issuing the command to turn off the PV system for one time instant (one minute). Across all the days mentioned above, issuing such a command did not affect the system negatively, as in the regulator tap positions, capacitor states, storage unit state, and bus voltages remained the same. This can be seen in **Table 2.13** below, where issuing the command for one minute did not affect the regulator tap position switching. As such, we had to launch the commands for longer time instants. Therefore, we launched the same commands mentioned above for 5, 10, and 20 consecutive time instants. Across all the days mentioned above, the only assets that were affected by these commands were the regulators, where the regulator tap positions switching increased after launching the malicious commands. We also observed that launching the commands for 20 consecutive minutes resulted in the most malicious activity in the network. An example of the test of issuing commands can be shown in **Table 2.13** below for December 21st.

Table 2.13: Base Case and Test Case Comparison for PV System on December 21st.

First Instant	# of Instants	Command (Percentage of PV Production)	Reg. 1 Tap Changes - A (base case)	Reg. 1 Tap Changes - A (test case)	Reg. 1 Tap Changes - B (base case)	Reg. 1 Tap Changes - B (test case)	Reg. 1 Tap Changes - C (base case)	Reg. 1 Tap Changes - C (test case)	Reg. 2 Tap Changes - A (base case)	Reg. 2 Tap Changes - A (test case)	Reg. 2 Tap Changes - B (base case)	Reg. 2 Tap Changes - B (test case)	Reg. 2 Tap Changes - C (base case)	Reg. 2 Tap Changes - C (test case)
510,481	1	0	0	0	0	0	0	0	0	0	4	6	10	10
510,481	20	0	0	2	0	2	0	2	6	14	10	16	12	8
510,481	20	30	0	2	0	2	0	0	6	10	10	12	12	12
510,481	20	50	0	0	0	0	0	0	6	12	10	14	12	6
510,481	20	70	0	0	0	0	0	0	6	6	10	10	12	6

Table 2.13 shows the number of times that the regulator tap position switched for phases A, B, and in both voltage regulators, where voltage regulator 1 is the voltage regulator located at bus 814 and voltage regulator 2 is located at bus 852. As can be seen from **Table 2.13** and is a pattern consistent across all the days tested, issuing the command for a longer period of time results in a more malicious activity, where it can be seen that for all phases of regulator 1 and for phases A and B of regulator 2, the number of tap position switching increased after the command was launched (also known as the test case).

After having performed all these tests, we have concluded that both regulators are the two assets affected by issuing malicious commands to the PV system. As such, we started creating the malicious dataset for developing the anomaly detector for this asset. The steps followed are also quite similar to generating the malicious dataset for the regulators, where a random instant is first selected. Afterward, a command with a randomly selected percentage of PV production is launched at that random instant. A test is then run to determine if the command that was issued is malicious, which involves comparing the number of tap position switching in both regulators before and after launching the command. If the number of tap position switching is higher after launching the command, then the command is determined to be malicious and is used to create the malicious dataset that we will then use for the cross-validation and testing phases.

2.2.2.3 Storage Element

For developing the anomaly detection algorithm, it is important to choose the right features so that we are able to identify when a malicious command has been issued. To identify these features for the storage element, we ran tests to study the behavior of storage element properties. The test was done by comparing the storage element property values before and after issuing the malicious command. We also considered how the behavior of other assets changed before and after issuing the malicious command to the storage element.

The malicious command we issued to the storage element was the following,

Command: At n=2, set the storage element state to be discharging.

We picked this time instant because the storage element is set to be in the idle state during this given time period. Therefore, an anomalous command for the storage element would be to switch to discharging state. Upon issuing this command, we obtained the property values and listed it on **Table 2.14** to compare the values before and after the malicious command was issued.

Table 2.14: Power Discharge in the Storage Element Before/After Applying the Malicious Command.

Instant	Base kWOut	Test kWOut	Base kWIn	Test kWIn
2	0	0	0	0
3	0	200	0	0
4	0	0	0	363.105
5	0	0	0	0
6	0	0	0	0
7	0	0	0	0
8	0	0	0	0
9	0	0	0	0
10	0	0	0	0
11	0	0	0	0
12	0	0	0	0
13	0	0	0	0
14	0	0	0	0
15	0	0	0	0
16	0	0	0	0
17	0	0	0	0
18	0	0	0	0
19	0	0	0	0

20	0	0	0	0
21	0	0	0	0
22	0	0	0	0
23	0	0	0	0
24	0	0	0	0
25	0	0	0	0
26	0	0	0	0
27	0	0	0	0
28	0	0	0	0
29	0	0	0	0
30	0	0	0	0
31	0	0	0	0

The storage element property ‘kWOut’ corresponds to the power flowing out or discharging from the storage element. The property ‘kWIn’ corresponds to the power flowing in or charging into the storage element.

Table 2.14 depicts the power flow in the storage element after applying the malicious command.

The only noticeable observation to be made after running the test for the storage element with this malicious command was that there was an extra charge-discharge cycle. At instant 3, the storage element flows out 200 kW power and at instant 4, 363.105 kW power is flowed into the storage element. It could be noted that more power is charged into the storage than discharged.

After issuing the command to the storage element, it was also found that there weren’t any noticeable changes in the behavior of other assets in the network.

In specific,

- No ANSI violations: It was found that there were no ANSI violations in the dataset after the command was issued. Therefore, ANSI limits would not be a good feature to include for the storage element algorithm since it won’t help to identify the malicious command.
- No change in capacitor states: It was found that there were no changes in the capacitor states after the command was issued. Therefore, capacitor states would not be a good

feature to include for the storage element algorithm since it won't help to identify the malicious command.

Since there weren't any storage element properties through which we were able to identify a malicious command, we decided not to include an anomaly detection algorithm for the storage element.

2.2.2.4 Capacitor

For developing the anomaly detection algorithm, it is important to choose the right features so that we are able to identify when a malicious command has been issued. To identify these features for the capacitor, we ran tests to study the behavior of capacitor properties. The test was done by comparing the capacitor property values before and after issuing the malicious command. We also considered how the behavior of other assets changed before and after issuing the malicious command to the capacitor.

The malicious command we issued to the capacitor was the following,

Command: At n=2, the capacitor state is set to ON.

We picked this time instant after studying the capacitor states for our normal dataset. This time instant corresponded to a point where the capacitor state should be turned OFF. Therefore, an anomalous command for the capacitor at this time, would be to switch to its ON state. Upon issuing this command, we obtained the property values and compared it with the values before and after the malicious command was issued.

We found that the malicious test case and the normal case showed little-to-no difference in its property values. The number of capacitor states were very similar and would not be a good feature to be included in the detection algorithm. We then looked at how the behavior of other assets changed upon issuing the command.

In specific,

- No ANSI violations: It was found that there were no ANSI violations in the dataset after the command was issued. Therefore, ANSI limits would not be a good feature to include for the capacitor algorithm since it won't help to identify the malicious command.

- No change in storage states, kWOut and kWIn: It was found that there were no changes in the storage states after the command was issued. Therefore, storage states would not be a good feature to include for the capacitor algorithm since it won't help to identify the malicious command.

- Slight change in reactive power: There was a slight change observed in the reactive power. The test case showed a difference of 2-3 kVAR in the 30-minute time frame. However, this is not a significant enough difference for it to be considered as a feature for the algorithm.

- Slight change in regulator taps: Similarly, there was only a slight change observed in the number of regulator taps. The test case displayed a few more extra taps switching for the regulator. However, as in the case of reactive power, there wasn't a significant enough difference for it to be considered as a feature for the capacitor algorithm.

Since there weren't any capacitor properties through which we were able to identify a malicious command, we decided not to include an anomaly detection algorithm for the capacitor.

2.3 Machine Learning

2.3.1 Local Outlier Factor algorithm

For our anomaly detection algorithm, we are using an unsupervised learning method called Local Outlier Factor (LOF) algorithm. The LOF algorithm computes the local density deviation of a given point with respect to its neighbors. Each point in the dataset is given an LOF score and if the LOF score is greater than a certain threshold, the point is an anomaly. The LOF score is based on density calculations and therefore the samples that have a much lower density than their neighboring points are classified as outliers. For example, in **Figure 15**, points P1 and P2 are considered outliers with respect to clusters C1 and C2, respectively, due to them having a much lower density than their neighboring points.

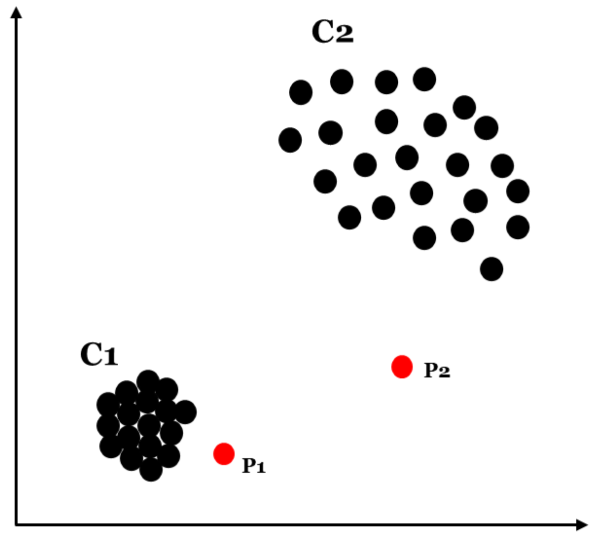


Figure 15: Visualization of LOF Algorithm

The algorithm consists of several steps to classify the points in the dataset,

1) Compute k-distance of the point ($dist_k(A)$)

The first step of the algorithm is to compute the k-distance of the point. The 'k' parameter is a parameter we set up to acquire a k-distance. K-distance of a point corresponds to the distance between a point, 'A' and its k-th nearest neighbor.

For example, given a test point 'A' and 'k' value of 4, the k-distance of 'A' would be,

$$dist_4(A) = Distance\ between\ A\ and\ its\ 4th\ nearest\ neighbor$$

Another term to understand when using the LOF algorithm is the concept of 'k-neighborhood'. The 'k-neighborhood' corresponds to the set of all points that lie in or on the circle with a radius of the k-distance.

For example, given a test point ‘A’ and a ‘k’ value of 4, the k-neighborhood of A would be,

$$N_4(A) = \{set\ of\ all\ points\ that\ lie\ in\ or\ on\ circle\ with\ radius\ 4, center\ A\}$$

2) Compute Average Reachability Distance ($Average\ RD(X_i)$)

For a given test point Xi and Xj, the reachability distance of Xi with respect to Xj is calculated as follows,

$$RD(X_i, X_j) = max\ (dist_k(X_j), dist(X_i, X_j)) \tag{2.3}$$

If a point lies within the k-neighborhood of Xj, the reachability distance will be the k-distance of Xj. If a point lies outside the k-neighborhood of Xj, the reachability distance will be the Euclidean distance between the points Xi and Xj.

In this step, we compute the average reachability distances of all neighbors of the test point.

$$Avg.RD(X_i) = \frac{sum\left(RD(X_i, X_j)\right)}{N} \tag{2.4}$$

Where, N is the number of points in the k-neighborhood of Xi.

3) Compute Local Reachability Distance ($LRD\ (X_i)$)

This is the inverse of the average reachability distances of all neighbors of the test point.

$$LRD\ (X_i) = \frac{1}{Average\ RD(X_i, X_j)} \tag{2.5}$$

If the LRD (Xi) value is low, it indicates a high value for the Average RD (Xi). This signifies that the test point, Xi is far away from its neighbors. There is less density of points around Xi and the closest cluster is far from Xi.

4) Compute the LOF score $(LOF_k(Xi))$

The Local Outlier Factor (LOF) score is the ratio of the average LRD of all the k-neighbors of the test point, Xi to the LRD of Xi.

$$LOF_k(Xi) = \frac{Avg.LRD}{LRD_{test}} \tag{2.6}$$

Where, Avg. LRD is the average LRD of all the neighboring points and LRD_{test} is the LRD of the test points.

If the test point is not an outlier,

- The density of the point and its neighbors are roughly the same.
- We obtain an LOF score of nearly 1.

If the test point is an outlier,

- The LRD of the point will be smaller than the average LRD of its neighboring points.
- We obtain a high LOF value, generally greater than 1.

Therefore, in general, if the LOF score is greater than 1, the point is considered an anomaly. If the LOF score is less than or equal to 1, the point is considered not an anomaly.

To summarize the steps behind the LOF algorithm, at first, the algorithm calculates the reachability distance from all the neighbors (in the k-distance neighborhood of the given point) to the given point. Then, we proceed to calculate the local reachability distance of the point. The LOF score of the point is calculated using the local reachability of its neighbors and the point itself. If their LOF score is greater than the threshold value set, the point is considered an anomaly. These steps are then repeated for every point in the dataset.

2.3.2 Features

When running the tests to develop the malicious datasets for the regulators and the PV system, we were able to better understand our system, which helped us determine the features for each anomaly detector. As such, the features for each asset are as follows:

1. Regulator 1 (Bus 814) and Regulator 1 (Bus 852): time instant, command issued, and regulator tap position in the previous time instant.

2. Photovoltaic (PV) System: time instant, command issued, PV production in the previous time instant, and regulator tap position in the previous time instant.

2.3.3 Dataset

Another thing that needed to be determined before training and testing the algorithms were the sizes of the training, cross-validation, and testing datasets. The dataset sizes for each of the assets were determined as follows:

1. Training Dataset: around 3.6 million points.

2. Cross-Validation Dataset: around 1 million points.

3. Testing Dataset: around 1 million points.

The training dataset contains the normal operating conditions of our model, which is also referred to as the base case data. The cross-validation and testing datasets of normal operating points and the malicious points that were obtained from the tests discussed earlier.

2.3.4 Detailed Algorithm

After having discussed the required parameters to develop the anomaly detectors for the assets, this section will discuss the flowchart of the algorithm. One important thing to discuss before detailing the specifics of the algorithm is determining the k-value. The ‘k’ parameter is a parameter we set up to acquire a k-distance [13]. K-distance of a point corresponds to the distance between a point, ‘A’ and its k-th nearest neighbor. This is the first computational step of the algorithm. Various literature [14]-[15] suggests running the algorithm over a range of k instead of a single value of k due to the LOF score varying for different values of k. Therefore, there exists the need to define a range; a lower bound and an upper bound for the values of k. The lower bound, ‘min. k’ is set so that it is greater than the minimum number of objects a cluster has to contain, so that other objects can be local outliers with respect to this cluster. The upper bound, ‘max. k’ is set so that it is smaller than the maximum number of nearby objects that can possibly be local outliers. It is the maximum number of objects that you want to be considered outliers if clustered together. As such, when considering the lower bound, we selected a k-value of 1 to ensure that no clusters are missed. For example, if there was a cluster of 3 datapoints, and we selected the minimum k-value to be 4, that cluster would not be identified by the algorithm. When considering the upper boundary for the k-value, the correct value to be selected should be equal to the number of Monte Carlo deviations. In the case for our project, we should select the upper value to be 7, since we have 7 Monte Carlo deviations in the training dataset. This is justified by the fact that clusters are made up of similar values, and since we are running the same simulation, with some deviations, it makes sense that clusters would consist of around 7 points. However, after studying the datasets, it was noticed that multiple points in the duration of one year are similar. Therefore, it is very likely that clusters would consist of points greater than 7. As such, we selected the upper bound k-value to be 15 for the algorithms of the three assets in the system. Therefore, the range of k-values was from 1 to 15.

After determining the range of k-values, we ran the LOF algorithm, which is described in **Figure 16** below. The LOF algorithm utilized in this paper was obtained from a MATLAB toolbox [16]. Our anomaly detector model assumes a developed anomaly detector for each of the three assets, where the assets communicate with each other to share their operating status.

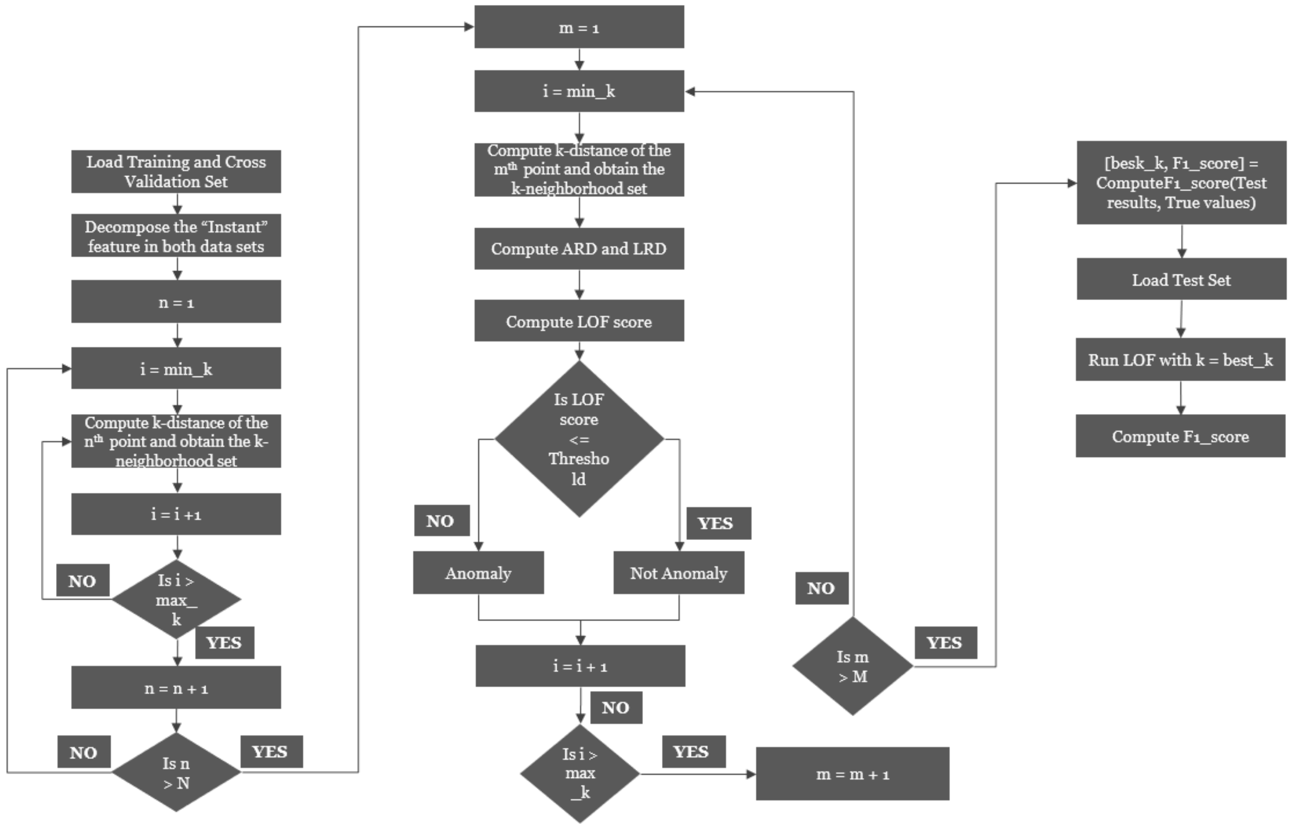

Load Training and Cross Validation Set
Decompose the "Instant" feature in both data sets
n = 1
i = min_k
Compute k-distance of the nth point and obtain the k-neighborhood set
i = i +1
Is i > max_k
NO
YES
n = n + 1
Is n > N
NO
YES
m = 1
i = min_k
Compute k-distance of the mth point and obtain the k-neighborhood set
Compute ARD and LRD
Compute LOF score
Is LOF score <= Threshold
NO
YES
Anomaly
Not Anomaly
i = i + 1
NO
Is i > max_k
YES
m = m + 1
Is m > M
NO
YES
[besk_k, F1_score] = ComputeF1_score(Test results, True values)
Load Test Set
Run LOF with k = best_k
Compute F1_score

Figure 16: Local Outlier Factor Algorithm Flowchart

Figure 16 describes in detail how our algorithm works. We first load the training and cross-validation datasets. We then decompose the "Instant" feature in both datasets by splitting it in the months, days, and minutes. Afterward, we compute the k-distance of the n-th point and obtain the k-neighborhood set. The average reachability and local reachability distances are then computed. We then use these to compute the LOF score. If the LOF score is less than the threshold, the command is not an anomaly, but if the LOF score is greater than the threshold, then the command is an anomaly. After all these computations, a small section of the code is used to obtain the best k-value that gives the highest F1 score. After determining the optimal k-value, the code runs the testing phase with that k-value, and the F1 score is calculated.

2.3.5 Evaluation Metrics

After developing our anomaly detection algorithm, we needed to evaluate how well the algorithm does on detecting the malicious commands. For evaluating our anomaly detection algorithm, we utilize the method of F1-Score. We chose this evaluation metric over other common methods such as accuracy due to its performance in real-life scenarios.

In a real-life setting, the number of anomalous samples are typically far lower than the number of good samples. For instance, imagine a dataset of 100 samples with 5 anomalous samples and the remaining 95 as good samples. If our detection algorithm predicts all the samples as being good samples, we will obtain an accuracy of 95% which gives an impression that the algorithm is accurate and performs really well. However, the algorithm will have failed its primary goal of detecting anomalies in the dataset. On the other hand, if we calculate the F1-Score for this case, we will obtain zero which tells us correctly that the algorithm has failed in detecting anomalies.

The F1-Score is computed using precision and recall. Precision is the ratio of correctly classified anomalous samples to the total number of samples classified as anomalies. It is calculated as follows,

$$Precision\ (P) = \frac{TP}{TP + FP} \tag{2.7}$$

Recall is the ratio of correctly classified anomalous samples to the total number of anomalous samples in the dataset.

$$Recall\ (R) = \frac{TP}{TP + FN} \tag{2.8}$$

Where,

- True Positives (TP): This value corresponds to the number of true positives obtained after running the algorithm. True positives are the outcomes where the algorithm correctly classifies a point as an anomaly.
- False Positives (FP): This value corresponds to the number of false positives obtained after running the algorithm. False positives are the outcomes where the point is not an anomaly but the algorithm incorrectly classifies the point as an anomaly.
- False Negatives (FN): This value corresponds to the number of false negatives obtained after running the algorithm. False negatives are the outcomes where the point is an anomaly but the algorithm incorrectly classifies the point as not an anomaly.

The F1-Score is computed using the following equation involving Precision (P) and Recall (R),

$$F1\ Score = \frac{2 * P * R}{P + R} \tag{2.9}$$

The F1-Score gives us a value between zero and one. The algorithm is considered perfect if the F1-Score is 1, and it is considered a failure when the F1-Score is 0.

3. RESULTS

3.1 Regulator 1

As we have mentioned, a Monte Carlo simulation on the base case data was run to build the training, cross-validation, and testing datasets for all the anomaly detectors. For the first regulator (situated at Bus 814), the size of the training dataset was 7 Monte Carlo deviations, where each deviation consists of 525,600 points (corresponding to one-year's set of data). Therefore, the training dataset size was around 3.6 million points. The malicious dataset comprised of 376 malicious points, which meant the number of anomalies or actual positives are 376. The cross-validation dataset comprised of these anomalous points, in addition to 1 Monte Carlo deviation (525,600 points making up the actual negatives). As such, the cross validation dataset size was 525,976 points. Similarly, the testing dataset comprised of 525,600 normal points (actual negatives) and 400 malicious points (actual positives).

Training and testing for the anomaly detector of Regulator 1 have been finalized. **Table 3.1** shows the cross-validation results obtained for this regulator. The cross-validation analysis was done through altering the threshold value for each run.

Table 3.1: Cross-Validation Results for Regulator 1.

Run	Threshold	True Positives	False Positives	False Negatives	True Negatives	Precision	Recall	F1 Score
1	1.26	376	29,678	0	495,922	0.0125	1	0.025
2	1.6	371	1,002	5	524,598	0.27	0.9867	0.424
3	1.7	369	335	7	525,265	0.5224	0.9814	0.683
4	1.8	365	147	11	525,453	0.71	0.997	0.822
5	1.96	357	8	19	525,592	0.97	0.94	0.955

The best F1 score is obtained for a threshold of 1.96. However, upon analysis of the results, we determined that there should be a higher cost associated with false negatives in the electrical grid, where recall must have more importance than precision. Missing an anomaly is more critical than flagging a regular point as an anomaly, as that one anomaly can have disastrous effects on the entire grid. As such, we determined that there is a trade-off between the number of false positives and false negatives. We can see that as the threshold decreases, the number of false negatives decrease and the value of recall increases. Moreover, we also see that precision decreases as the number of false positives increase. Therefore, we determined that the threshold of 1.8 would be an apt choice, as it only had 11 false negatives. This meant that out of the total 376 malicious commands, it only missed 11 malicious commands. Moreover, it only had 147 false positives, meaning that out of the 525,600 normal points, it only identified 147 normal points as anomalies. For the cross-validation analysis, we used a k-value of 10 which was concluded by our machine learning code to be the optimal k-value for the given dataset.

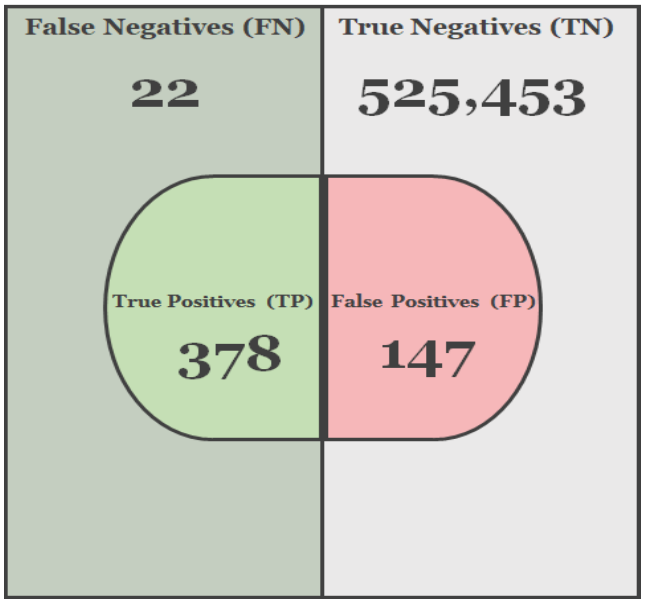


Figure 27: Test Results for Regulator 1

We utilized the optimum k-value of 10 along with our threshold value of 1.8, to begin the testing phase for our Regulator 1 anomaly detector. The results of the test are displayed in **Figure 17**. Using this threshold, we were able to obtain only 22 false negatives and 147 false positives. The value of precision was 0.72, recall was 0.945, and the F1 score was 0.8173. These are acceptable values given the trade-off between false positives and false negatives discussed earlier. Therefore, we finalized these values as the parameters for our Regulator 1 anomaly detector.

3.2 Regulator 2

For the second regulator (situated at Bus 852), we followed a similar approach as to the one taken for Regulator 1 to develop the anomaly detector. The size of the training dataset was 7 Monte Carlo deviations, where each deviation consists of 525,600 points (corresponding to one-years' set of data). Therefore, the training dataset size was around 3.6 million points. The malicious dataset comprised of 360 malicious points, that meant the number of anomalies or actual positives are 360. The cross-validation dataset comprised of these anomalous points, in addition to 1 Monte Carlo deviation (525,600 points making up the actual negatives). As such, the cross validation dataset size was 525,960 points. Similarly, the testing dataset comprised of 525,600 normal points (actual negatives) and 359 malicious points (actual positives).

Training and testing for the anomaly detector of Regulator 1 have been finalized. **Table 3.2** shows the cross-validation results obtained for this regulator. The cross-validation analysis was done through altering the threshold value for each run.

Table 3.2: Cross-Validation Results for Regulator 2.

Run	Threshold	True Positives	False Positives	False Negatives	True Negatives	Precision	Recall	F1 Score
1	1.3	359	21,935	1	503,665	0.016	0.997	0.032

2	1.6	353	1,032	7	524,568	0.25	0.981	0.405
3	1.7	350	275	10	525,325	0.56	0.972	0.71
4	1.8	345	63	15	525,537	0.845	0.958	0.89
5	1.9	342	9	18	525,591	0.974	0.95	0.962
6	2.0	336	1	24	525,599	0.997	0.933	0.96

Upon analysis of the cross-validation results, we reached a similar conclusion as to the previous regulator where there was a trade-off between false positives and false negatives. The best F1 score is obtained for a threshold of 1.9. However, due to the high cost of false negatives, we determined the optimal threshold to be 1.8, as it only had 15 false negatives. This meant that out of the total 360 malicious commands, it only missed 15 malicious commands. Moreover, it only had 63 false positives, meaning that out of the 525,600 normal points, it only identified 63 normal points as anomalies. Once again, our machine learning code concluded the optimal k-value to be 10, which was therefore used in this analysis.

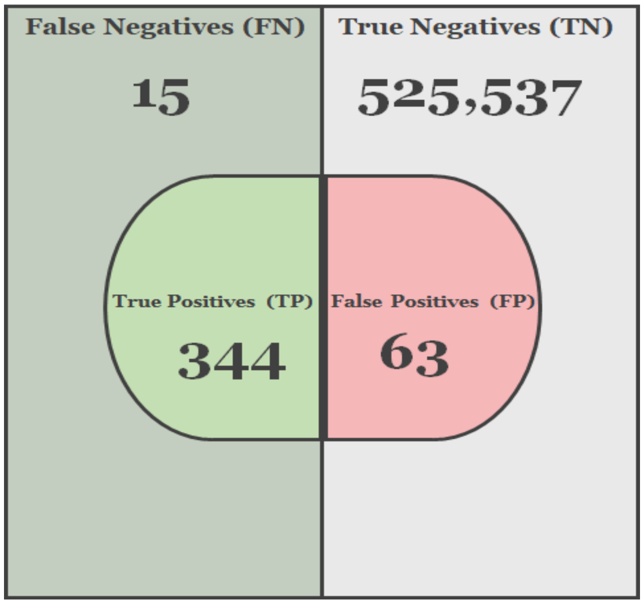


Figure 18: Test Results for Regulator 2

We utilized the optimum k-value of 10 along with our threshold value of 1.8, to begin the testing phase for our Regulator 2 anomaly detector. The results of the test are displayed in **Figure 18**. Using this threshold, we were able to obtain only 15 false negatives and 63 false positives. The precision was 0.8452, recall was 0.9582, and the F1 score was 0.8982. These are acceptable values given the trade-off discussed, and therefore, we finalized these values as the parameters for our Regulator 2 anomaly detector.

3.3 Photovoltaic System

For the photovoltaic system, we followed a similar approach as to the ones done for both the regulators to develop the anomaly detector. The size of the training dataset was 7 Monte Carlo deviations, where each deviation consists of 525,600 points (corresponding to one-years' set of data). Therefore, the training dataset size was around 3.6 million points. The malicious dataset comprised of 100 malicious points, which meant the number of anomalies or actual positives are 100. The cross-validation dataset comprised of these anomalous points, in addition to 1 Monte Carlo deviation (525,600 points making up the actual negatives). As such, the cross validation dataset size was 525,700 points. Similarly, the testing dataset comprised of 525,600 normal points (actual negatives) and 100 malicious points (actual positives).

Training and testing for the anomaly detector of the photovoltaic system have been finalized, and **Figure 19** shows the cross-validation results obtained for the system. The cross-validation analysis was done through altering the threshold value for each run.

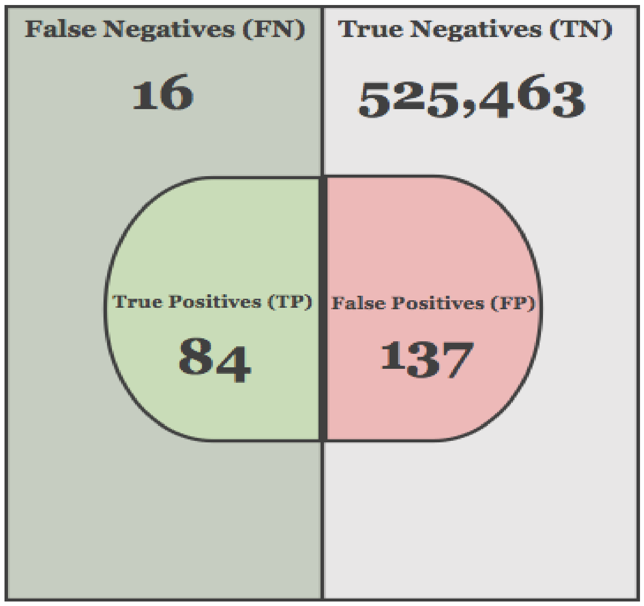


Figure 19: Cross-Validation Results for Photovoltaic system

Upon analysis of the cross-validation results, we reached a similar conclusion as to the previous regulator where there was a trade-off between false positives and false negatives. We determined the optimal threshold to be 2.0, as it only had 16 false negatives. This meant that out of the total 100 malicious commands, it only missed 16 malicious commands. Moreover, it only had 137 false positives, meaning that out of the 525,600 normal points, it only identified 137 normal points as anomalies. Using this threshold value, the precision was 0.3801, recall was 0.84, and the F1 score was 0.5234. Our machine learning code concluded the optimal k-value to be 15, which was therefore used in this analysis.

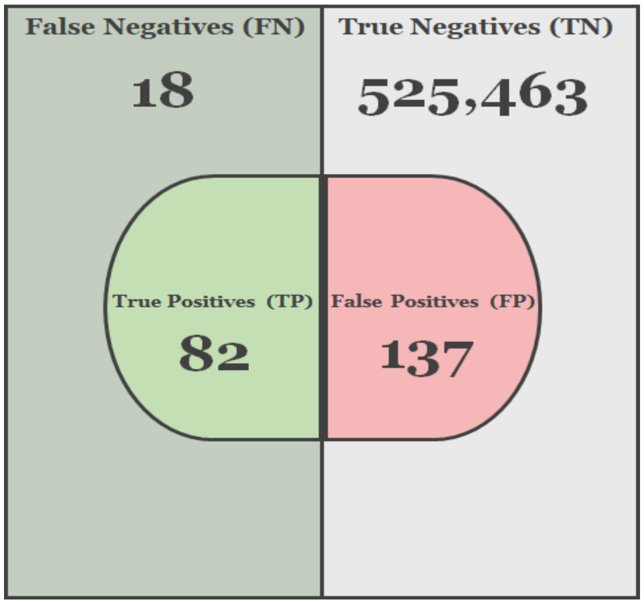


Figure 20: Test Results for Photovoltaic System

We utilized the optimum k-value of 15 along with our threshold value of 2.0, to begin the testing phase for our PV system. The results of the test are displayed in **Figure 20**. Using this threshold, we were able to obtain only 18 false negatives and 137 false positives. The precision was 0.3744, recall was 0.82, and the F1 score was 0.5141. These are acceptable values given the trade-off discussed, and therefore, we finalized these values as the parameters for our photovoltaic system anomaly detector.

4. CONCLUSION

Considering the increased implementation of distributed energy resources, there exists a high need for a change in the existing grid paradigm. Smart grids offer an intelligent, reliable, sustainable, economic and secure electric supply for the electric network. However, the grid might be subjected to various coordinated attacks, at the cyber or the physical level. Implementing a robust security system to defend against malicious activities is integral to the success of smart grids. Current anomaly detection techniques are focused on attacks that result in real-time consequences to the network. However, an intelligent, well-coordinated attack that doesn’t cause immediate changes in network performance can bypass these detection techniques.

Our project offers a defensive framework against this by detecting the malicious commands issued to assets in a smart distribution network. We have developed a machine learning based anomaly detector capable of identifying a malicious command issued to the asset’s controller. Our approach of detecting anomalies in the commands rather than anomalies in the monitored data enhances the system’s cyber-physical resilience by detecting any command that does not make sense based on the historical patterns of the received commands and/or the current status of the network. This enhances the integrity and improves the acceptance of the paradigm at the market, community and socio-political level.

REFERENCES

[1] R. S. d. Carvalho and D. Saleem, "Recommended Functionalities for Improving Cybersecurity of Distributed Energy Resources," in 2019 Resilience Week (RWS), San Antonio, 2019.

[2] K. Shallenberger, "DER aggregation: Sector experts identify emerging trends in a nascent market," Utility Dive, 24 June 2017. [Online]. Available: https://www.utilitydive.com/news/der-aggregation-sector-experts-identify-emerging-trends-in-a-nascent-marke/447670/. [Accessed 15 August 2020].

[3] L. Cui, Y. Qu, L. Gao, G. Xie and S. Yu, "Detecting false data attacks using machine learning techniques in smart grid: A survey," Journal of Network and Computer Applications, vol. 170, 2020.

[4] SP Energy Networks, "Non Technical Loses," SP Energy Networks, [Online]. Available: https://www.spenergynetworks.co.uk/pages/non_technical_losses.aspx. [Accessed 10 August 2020].

[5] U.S Department of Energy, "Financial Opportunities: Funding Opportunity Exchange," 5 February 2020. [Online]. Available: https://eere-exchange.energy.gov/Default.aspx?Search=landscape&SearchType=. [Accessed 13 September 2020].

[6] M. E. Chamie, K. G. Lore, D. M. Shila and A. Surana, "Micro-PMUs, Physics-Based Features for Anomaly Detection in Power Grids with," in 2018 IEEE International Conference on Communications (ICC), Kansas, 2018.

[7] Pandey, S., Srivastava, A., &amp; Amidan, B. (2020). A Real Time Event Detection, Classification and Localization using Synchrophasor Data. *IEEE Transactions on Power Systems*, 1-1. doi:10.1109/tpwrs.2020.2986019.

[8] Deng, R., Xiao, G., Lu, R., Liang, H., &amp; Vasilakos, A. V. (2017). False Data Injection on State Estimation in Power Systems—Attacks, Impacts, and Defense: A Survey. *IEEE Transactions on Industrial Informatics, 13*(2), 411-423. doi:10.1109/tii.2016.2614396.

[9] "ENF Ltd.", Enfsolar.com, 2021. [Online]. Available: https://www.enfsolar.com/pv/panel-datasheet/crystalline/40186. [Accessed: 25- Jan- 2021].

[10] "Grid Support Inverter List- Full Data", California Energy Commission, 2021. [Online]. Available: https://www.energy.ca.gov/media/2365. [Accessed: 25- Jan- 2021].

[11] IEEE Power Engineering Society, "Resources | PES Test Feeder," 1992. [Online]. Available: https://site.ieee.org/pes-testfeeders/resources/. [Accessed 13 September 2020].

[12] "REopt Lite | REopt Energy Integration & Optimization | NREL", *Reopt.nrel.gov*, 2021. [Online]. Available: https://reopt.nrel.gov/tool/. [Accessed: 25- Jan- 2021].

[13] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, “LOF: identifying density-based local outliers,” ACM SIGMOD Record, vol. 29, no. 2, pp. 93–104, May 2000.

[14] C. C. Aggarwal, “Proximity-Based Outlier Detection,” in Outlier Analysis, New York, NY: Springer New York, 2013, pp. 101–133.

[15] O. Alghushairy, R. Alsini, T. Soule, and X. Ma, “A review of local outlier factor algorithms for outlier detection in big data streams,” Big Data Cogn. Comput., vol. 5, no. 1, pp. 1–24, 2021, doi: 10.3390/bdcc5010001.

[16] M. M. Breunig, H.P. Kriegel, R. T. Ng, and J. Sander, "LOF: Identifying Density-Based Local Outliers," presented at the SIGMOD 2000 Int. Conf. On Management of Data, Dallas, TX, 2000. [Online]. Available: http://doi.acm.org/10.1145/335191.335388.