Towards Online Change Point Algorithms on-the-edge for Real-Time Fault Detection in Power Converters

"The views expressed are those of the author and do not reflect the official policy or position of the Department of War or the U.S. Government".

Zhymir Thompson[1, 2], Cameron Ball[3], Austin R.J. Downey[2, 4], Kerry Sado[3], Enrico Santi[3], and Jason D. Bakos[1]

[1]Department of Computer Science and Engineering, University of South Carolina, Columbia, USA
[2]Department of Mechanical Engineering, University of South Carolina, Columbia, USA
[3]Department of Electrical Engineering, University of South Carolina, Columbia, USA
[4]Department of Civil and Environmental Engineering, University of South Carolina, Columbia, USA

ABSTRACT

Power systems consist of interconnected electrical components. A flaw or damaging event can quickly cascade leading to extensive damage. The repairs required can be expensive, and the time required to complete repairs acts as a second wave of economic damage to the system. Online real-time fault detection, when coupled with timely corrective actions, addresses this by minimizing the extent of damage. Algorithms for online real-time fault detection can be used to discern normal and anomalous grid behavior and to identify issues when they first arise so successive systems can respond appropriately. Online change point detection algorithms (e.g. Bayesian Online Change Point detection, Cumulative Summation, Expectation Maximization) need to be adapted to operate at micro-second latencies for applications in power electronics and systems. Also, a consistent interface would allow for current algorithms to be swappable and make incorporating future algorithms trivial. Here we show that online change point detection is applicable to fault detection in power systems, and we provide an API implementation for transmitting predictions over a network (i.e. Ethernet, WiFi, etc.). These algorithms are more flexible than manual thresholds and with micro-second latency are fast enough for power system applications. The online change point detection algorithms selected only use sample observations and statistical parameters to discern normal and abnormal observations. The performance of the selected change point algorithms are demonstrated via simulated data from a modeled DC-DC boost converter. The modeled converter was built to simulate normal operation and behavior under different fault conditions. Online change point detection algorithms used measurements of the stepped-up voltage to determine whether a fault occurred. One such algorithm detected the onset of a fault with a maximum delay of 5 ms from initiation. All algorithm predictions were received within 1 ms of the measurement data being sent.

Keywords: change point, fault detection, power converters, power systems, real-time, edge, anomaly detection

1. INTRODUCTION

Change point detection is defined as "The identification of abrupt changes in the generative parameters of sequential data".[1] This type of detection looks for abrupt changes to the distribution of data. Quick changes in amplitude and frequency could both qualify as an abrupt change. What is important, from the perspective of the detection model, is how much an observation(s) deviate from what is expected. Change point detection algorithms can be broadly categorized as either an offline or online detection method.[2] Offline detection algorithms observe an entire sequence of data to make decisions. They find the optimal locations to partition the given data such that

Further author information: (Send correspondence to Austin R.J. Downey)
E-mail: austindowney@sc.edu

Signal Processing, Sensor/Information Fusion, and Target Recognition XXXV, edited by Ivan Kadar, Erik P. Blasch
Lynne L. Grewe, Proc. of SPIE Vol. 14049, 140491A · 2026 Published by SPIE
0277-786X · doi: 10.1117/12.3096306

data within each partition is similar and adjacent partitions are dissimilar.[3] The locations where the partitions lie are the change points. Since the algorithms look at all the data at once, they tend to be more accurate at the expense of speed. Online detection observes data sequentially. They may either take in a singular point of data or a window of data. In either case, the detection algorithm will decide for each new observation whether that latest observation marks a change point. The decision is made upon observation. Online detection algorithms tend to be faster than offline detection algorithms, albeit less accurate. As an example, imagine that we monitor voltage for a house. We have an online detection algorithm taking the voltage measurements as input. One day, a power surge occurs. The online detection algorithm detects the surge when it occurs and sends an alert. The following day, we collect the voltage measurements taken the day of the surge. Given our prior knowledge that a surge occurred, we use our offline detection algorithm to search for two change points. The first should mark the beginning of the surge, and the second should mark the return to normality.

Digital twins would benefit from online change point detection. They rely on an internal representation of the state of the system that they mirror.[4] One notable benefit is that online change point detection can support digital twin models by detecting significant outliers in the model's state. Change point detection can track an inverter's state by monitoring output signals like voltage, current, and frequency. Sudden deviations may indicate faults such as switch failures or capacitor degradation, while gradual drifts signal aging components. Consider a power inverter with a digital twin that seeks to track its state of health by monitoring the converter's output. The digital twin would be inaccurate if a significant change to the physical component were to occur. It is necessary to detect when a change in the converter occurs. Change point detection algorithms achieve this by detecting when measured data becomes abnormal. Subsequently, other methods would be used to update the model parameters in the converter's digital twin.

Related works have investigated fault detection in various systems, and some have used change point detection algorithms for their approach. Tochev et. al used BOCPD to detect solder buildup in pipes by using vibration data collected from the pipes.[5] One benefit of their approach was minimal disruption to the target system. They could detect blockages effectively without disturbing the original system by measuring vibrations from the outside of the pipe. This work measures the output voltage of the target system, so faults can be detected with minimal disruption, too. The voltage measurements are also used directly. Change point detection is one of multiple possible methods of fault detection. As part of their work, Lazzaretti et. al detected faults in celestial-radiation-based energy harvesting plants by comparing the measured output power to an expected output power.[6] The expected power was calculated using a linear Auto-regressive with exogenous input data model (ARX model). The primary advantage that this work holds is the reduced need for prior data. The proposed approach only requires mean and variance. Su et. al demonstrated fault detection earlier in switching models.[7] Their research involved fault detection and identification for switched systems. They modeled the expected output given inductance and capacitance. When the difference between the expected and measured value exceeded a threshold, a fault was reported.

The purpose of this work is to detect the occurrence of shifts in measured data that could indicate a fault. Rapid detection allows for quicker responses that can limit the extent of cascading damage and reduce future cost to repair damage. This work includes three change point detection algorithms. First, Bayesian Online Change Point Detection (BOCPD) which uses a probabilistic model to infer change points in real-time, updating beliefs about the likelihood of a change occurring at each step.[1] Second, Expectation-Maximization (EM) which is an iterative algorithm that estimates hidden variables and parameters in probabilistic models.[8] It is a broadly applicable algorithm which can be adapted to change point detection by choosing the Gaussian mixture to be an incomplete set of observations from normal scenarios and fault scenarios. Third, the Cumulative Summation (CUSUM) algorithm is used to detect shifts in the mean of some measured process.[9] It reports a positive detection whenever the mean tracked by the model shifts beyond a predefined threshold. The model was added for its advantages in detecting faults that may have more subtle fault behavior. The original design of the algorithms required modification so that they could work closer to the hardware that they are intended to monitor.

In this work, a communication strategy needed to be established given that the fault detection algorithms were not guaranteed to receive the measurement data directly. Socket programming was the inter-process communication (IPC) method used for communicating between processes. Sockets enable sending data between processes as messages.[10] This method was chosen for two primary reasons. First, socket-client connections are

more widely applicable since they can be used for processes on separate devices or processes on the same device.[10] Second, socket programming is one of the easier IPC methods to implement. Data could be shared using shared memory if devices were known to reside on the same device, but that approach makes race conditions possible. User Datagram Protocol (UDP) was used for transporting data between the hypothetical devices. UDP and Transmission Control Protocol (TCP) are both standard protocols for transmitting data across a network,[10,11] but TCP includes acknowledgment of packet delivery where UDP does not. This makes TCP more reliable albeit with slightly increased delay. UDP was chosen over TCP since real-time communication requires high-speed more than reliable transmission. This paper makes two contributions: 1) This work demonstrates how online change point detection algorithms can detect electrical faults via real-time voltage measurements; 2) This work provides a design approach for converting the algorithms introduced so that they can operate on microcontroller-based edge-devices.

2. METHODS

The experiments conducted required a power systems component to evaluate the algorithms on detecting faults in a potential power system. A DC-DC boost converter was used as the surrogate component. A model of the converter was used instead of inducing faults in physical devices. This decision eliminated the cost of replacing components used for testing while simultaneously allowing for reproducible, flexible experiments. The following section is split into three subsections describing the model, algorithms, and networking architecture. Section 2.1 describes the model used to simulate a DC-DC boost converter, Section 2.2 discusses the aforementioned algorithms and how they function, and Section 2.3 explains how data moves about in the detection system.

2.1 Simulation

A boost converter model was used to test the change point detection algorithms. The model was designed so that it could simulate normal behavior in addition to a selection of potential faults. For example, an input short can be set to happen at a set time during the simulation. The illustration shown in Figure 1 represents the model used to simulate a boost converter and to generate data for the change point detection algorithms. The model had a constant power supply and a constant load. Figure 2 shows how the boost converter fit into the overall model. The PID controller was used to monitor and stabilize current and voltage levels. The fault selector controlled if and when a potential fault would occur. The output voltage was measured and recorded.

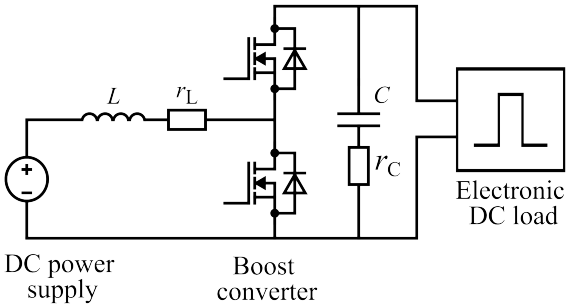

Figure 1. Electrical schematic of the simulated DC-DC boost converter, including source, switching stage, capacitor, and load.

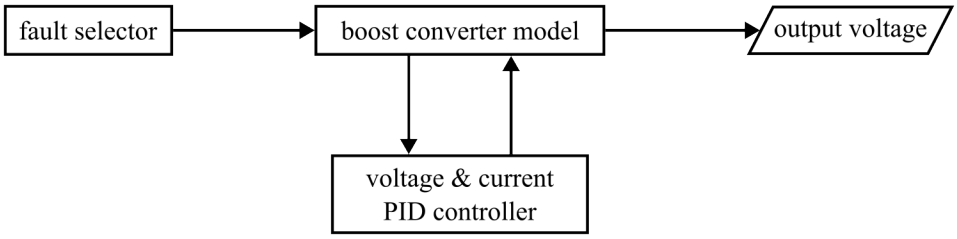

Figure 2. Block-level simulation architecture showing the fault selector, boost-converter model, PID control loop, and measured output voltage used by the detector.

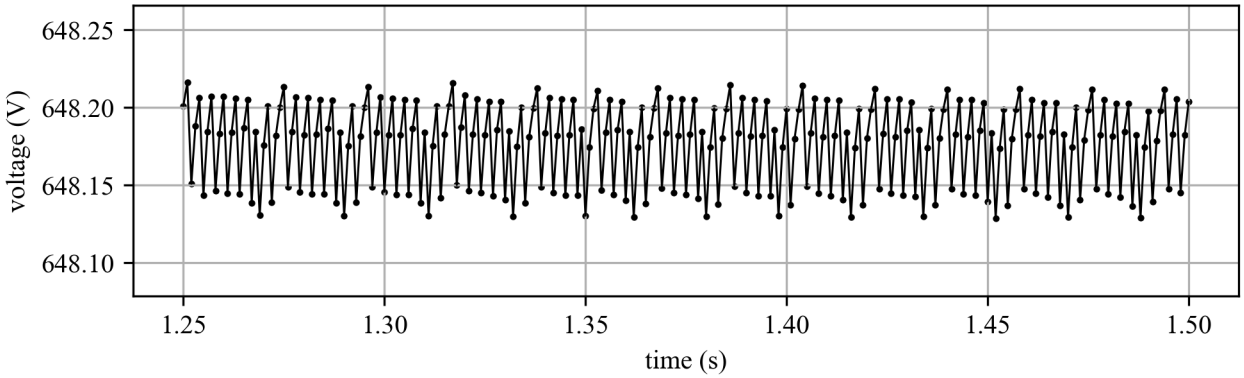

Figure 3. Output-voltage waveform under fault-free operation, sampled every 1 ms from the boost-converter simulation. The waveform exhibits periodic ripple about a nominal steady-state level.

The model had four run-time modes, with each mode corresponding to a different fault. One or more switches could be activated prior to running the simulation, and the respective fault scenarios would happen at the defined time. Multiple faults could be made to occur simultaneously, but the experiments conducted only involved activating one of the potential faults for any given simulation. Each simulation ran for five seconds of virtual time. The model updated once per microsecond (μs). The faults were set to occur one second into the simulation and remained throughout the remainder of the run. Samples were collected every millisecond (ms) yielding 5,001 samples of data per simulation. The samples collected were the inductor current, the voltage across the load, and the stepped-up output voltage. These measurements were tabulated along with the virtual time that the values were recorded.

The plot in Figure 3 shows the data collected from a simulation. It shows the output voltage measurements for the boost converter operating under normal conditions. The measurements possess a periodic ripple pattern. They also vary by less than 1 V. The periodic nature and consistent variance suggest that the data could be approximately represented by a Gaussian distribution. Consequently, any change to the surroundings within which the component exists that alters those properties could be readily detected.

2.2 Algorithms

In this work, the change point detection algorithms accept the converter's output voltage which was obtained via simulation. The algorithms would take a measurement sample, update model parameters, predict whether the measured value was normal or abnormal and move on to the next measurement sample. The models detected faults in simulated data generated from a MATLAB model. The data generated were saved into files intended to be read by the model. For model experimentation, the measurements from the files were converted into vectors of floating-point values. The models read data from an array value-by-value to simulate incoming data from a sensor. This setup worked well for developing the detection algorithms, but it is not adequate for simulating real-time predictions. First, experimental data is not always processed on the same device that reads the data. The data may be sent to another device for processing, or the data may be read from an attached sensor and processed on the host device. In either case, the data must be sent out from one process and received by another. This means that interprocess communication (IPC) is required to better simulate realistic scenarios.

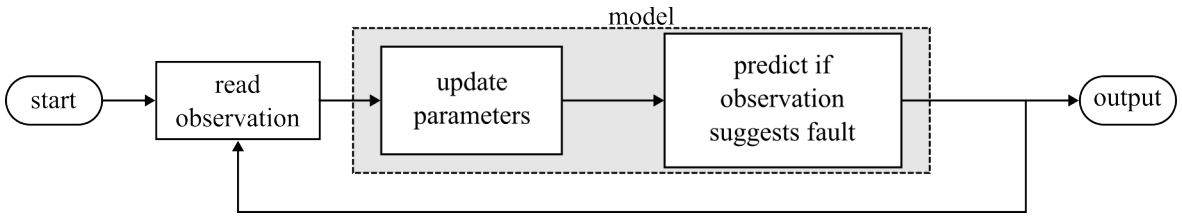

Figure 4. Diagram illustrating how an observation is transformed into a prediction of fault.

2.3 Transmission

To prepare the models for working with experimental data, two processes are made to run concurrently. Each process has two sockets: one socket transfers data one way, and the second socket transfers data the opposite way. One process is a Simulink model that sends out simulation data and receives a response. The second model processes received data and sends the predictions. Stream processing can be broken into three major steps:

1. Sending data to be processed
2. Processing the data
3. Sending the result to where it will be used

User Datagram Protocol (UDP) connections were used to address steps 1 and 3 (send data to be processed and send the result of processing to where it would be used). UDP is a protocol for data transmission over a network. UDP was chosen due to its speed and its ability to send data to remote computers. UDP is fast in part because it does not check for confirmation of data transmission .[11] Three UDP instances were constructed: one to send data from the simulation of the device to the detection model, one to receive data from the simulation for the model process, and one to send the probability from the model to the target.

The diagram in Figure 4 shows the overall process of converting an observed value into a model prediction. First, a device (possibly a sensor) reads a measurement from the system being observed. Then, the measurement value is sent to the model. Next, the model updates its internal parameters using the provided value, and then the model outputs a prediction representing whether the measurement is indicative of a fault condition. The output is sent off to be used elsewhere. The process loops by starting anew with a fresh measurement. The main takeaway of this diagram is that the process should be agnostic to the method(s) of transmitting data to and from the model. Further, the process should work regardless of the implementation details of the model used. These abstractions maximize the flexibility of the overall fault prediction workflow. An interface is the method chosen to abstract away the model implementation details.

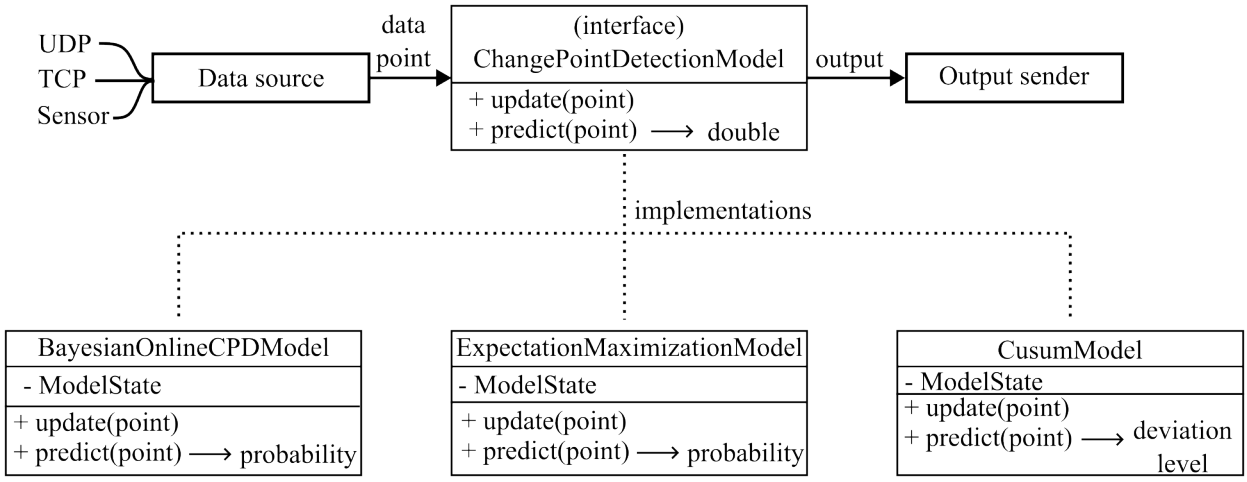

Figure 5. An interface description for the change point detection models. It comprises an update method and predict method.

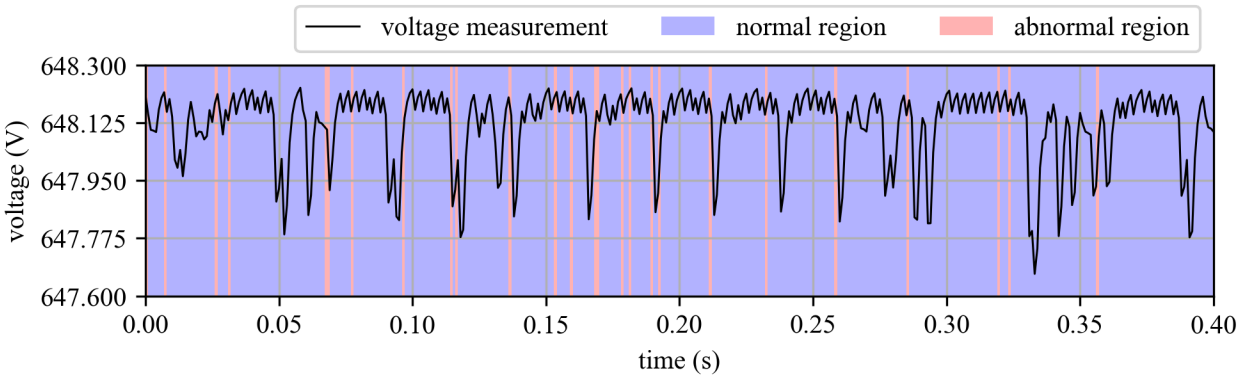

Figure 6. Normal and abnormal regions detected by Bayesian Online Change Point Detection (BOCPD) algorithm run on input short.

The model interface shown in Figure 5 has two methods. The update method uses the given input to update the parameters for the model. The prediction outputs a value which represents the model prediction. This interface is simple yet comes with two major benefits. First, the implemented change point detection models are interchangeable. The exact same method call can be used with all three models without any modification. Second, adding future models should require minimal effort. The only requirements are to implement the detection model as a class and implement the methods from the interface.

The proposed setup was run for different detection models and faults. The experiment involved two processes. The first process was responsible for the following: simulating the behavior for the component under study (in this case is a DC-DC boost converter), sending a voltage measurement every millisecond (10^{-3} seconds), and listening for a response at the designated port every tenth of a millisecond (10^{-4} seconds). The process listens at ten times the sending rate since there is no guarantee of when a response will come. Rapid polling provided a more accurate record of when responses were returned. The second process listened for the voltage measurement, sent it through the change point detection model for processing, and sent the model output back to the first process. These were written as asynchronous functions based on the assumption that the largest bottleneck for the process would be waiting to send and receive data.

3. RESULTS

The detection outputs illustrated in Figure 6 show how the naive BOCPD model performed for an input short. The blue dots show the measured voltage values at each corresponding time, and the orange dashed line represents the threshold for the model. For the Bayesian model, points below the threshold are reported abnormal while those above are normal. Figure 7 shows BOCPD model predictions for an input short. The short occurred one second into the simulation run. Most of the predictions are above the threshold line. This indicates that the model reported normal behavior for most of the time before the fault happened. Subfigure (d.) shows that the model reported the fault within 5 ms of its initiation.

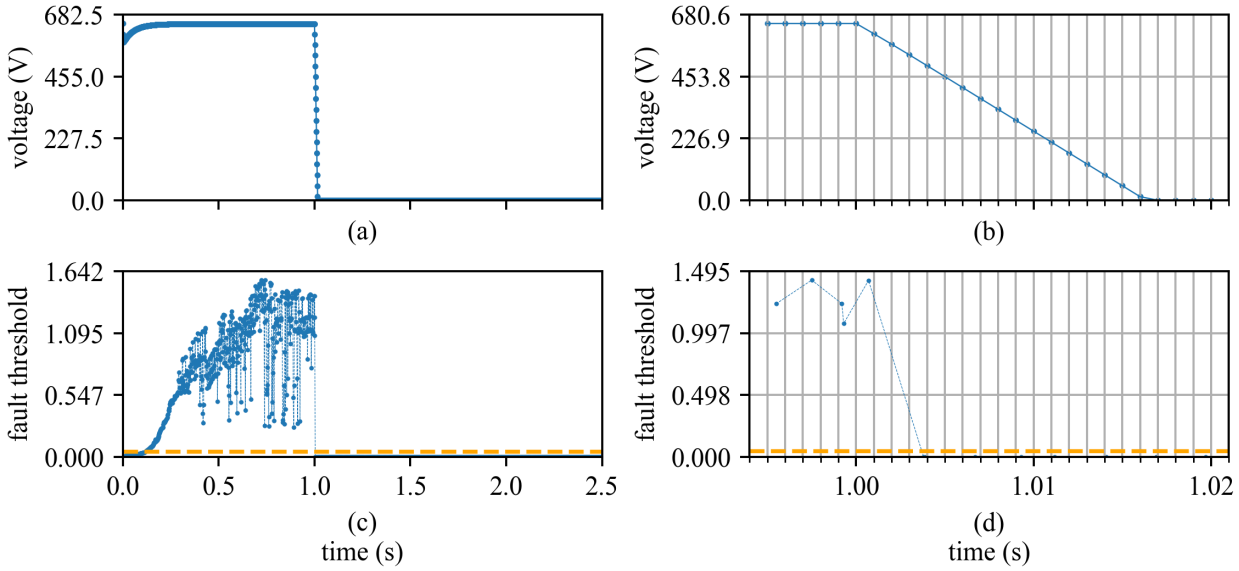

Figure 7. Detection for Bayesian Online Change Point Detection (BOCPD) algorithm on input short. Top left (a) shows voltage measurements. Top right (b) shows zoomed in view of voltage measurements around fault. Bottom left (c) shows model output and a horizontal bar representing threshold. Bottom right (d) shows zoomed in view of model response around threshold for zoomed in section.

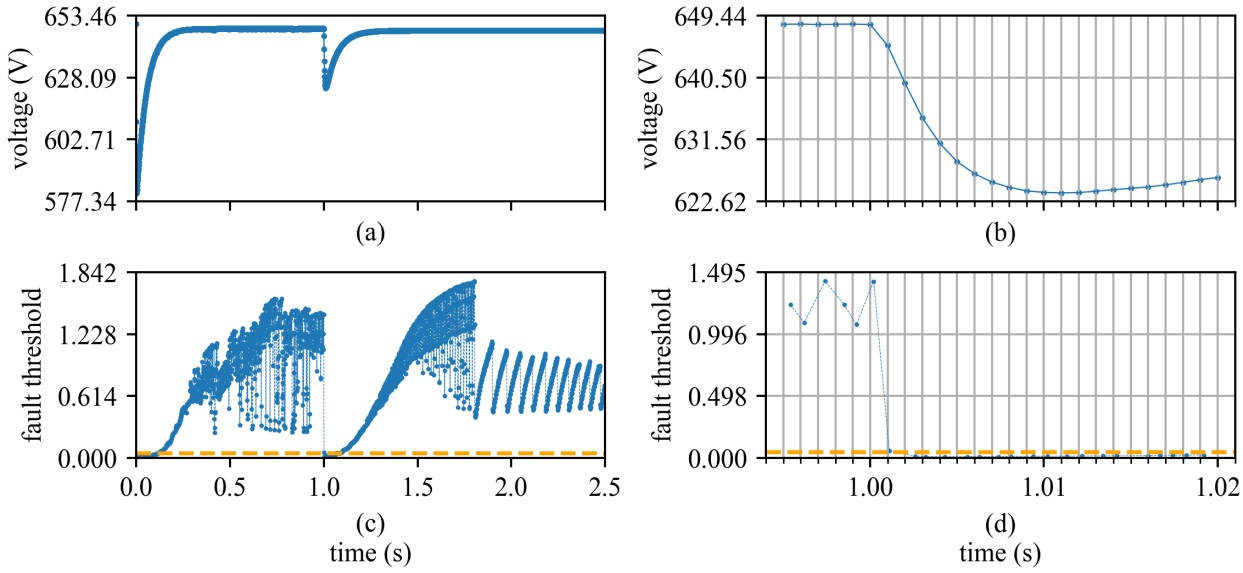

Figure 8. Detection for Bayesian Online Change Point Detection algorithm on input fault. Top left (a) shows voltage measurements. Top right (b) shows zoomed in view of voltage measurements around fault. Bottom left (c) shows model output and a horizontal bar representing threshold. Bottom right (d) shows zoomed in view of model response around threshold for zoomed in section.

Figure 8 shows the same model detecting faults for an input fault condition. In this case, the model detects the fault in about 3 ms. This was likely due to the steep voltage drop that occurred at the fault. The model reports normal behavior after 10 ms. This correlates with the voltage steadily increasing toward the original level. Figure 9 reports the results for an output short condition. The behavior shown in Figure 9 is very similar to that shown in Figure 7. In this case, fault detection happens in 2 ms. This is one or two ms faster than the input short. This is further evidence that the model can consistently perform in an online setting.

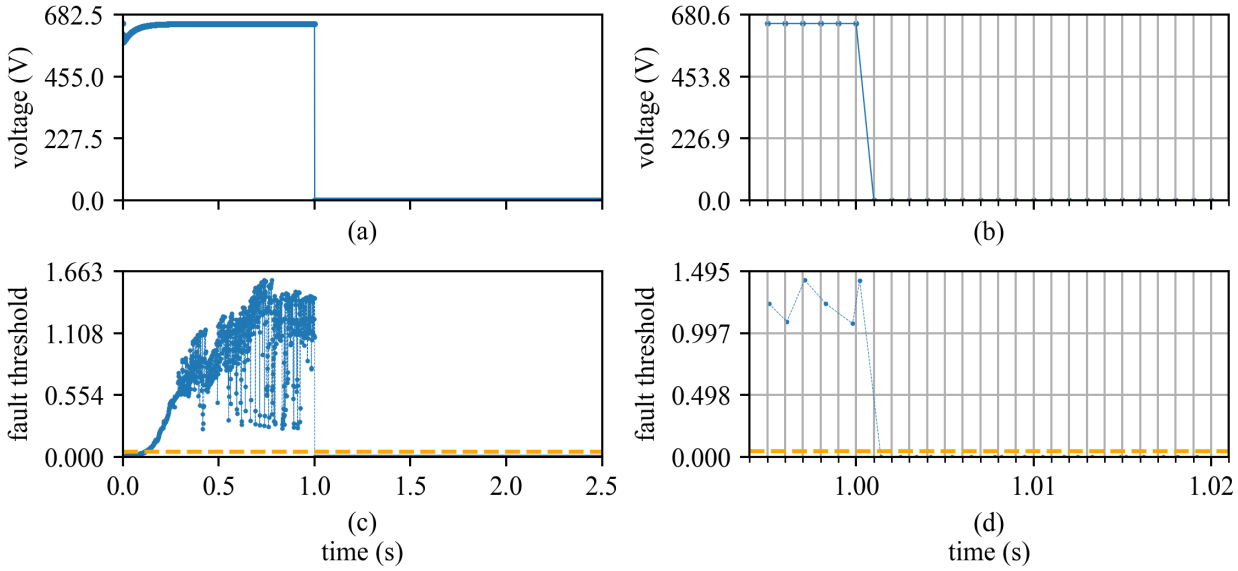

Figure 9. Detection for Bayesian Online Change Point Detection algorithm on output short. Top left (a) shows voltage measurements. Top right (b) shows zoomed in view of voltage measurements around fault. Bottom left (c) shows model output and a horizontal bar representing threshold. Bottom right (d) shows zoomed in view of model response around threshold for zoomed in section.

4. CONCLUSION

This work demonstrates online change point detection algorithms being used for online adaptive fault detection. Each of the algorithms presented maintain and update internal parameters with each new observation. In this way, the algorithms can adapt to new, unexpected measurements. The online change point detection algorithms discussed show promise for fault detection in power system applications. The algorithms only require readily accessible statistical properties for the data of interest. They have been shown to detect faults using a well-chosen observed variable. They are also fast enough to produce real-time predictions of fault occurrences. The primary disadvantage of the algorithms presented is that the data must follow a finite mixture of distributions. Data that does not fit well into a particular statistical distribution would likely yield inaccurate results.

Additionally, a preliminary design is discussed for converting these online change point algorithms to operate on edge-devices. The models excel in situations where the predictions can yield quick reactions. For that reason, a framework was designed so that the model implementation was separate from the communication method. The design is flexible since it is intentionally agnostic to the choice of detection model or mode of data transmission. As of writing, UDP is the only implemented communication method, but more options will be added later as necessary.

In future work, online change point algorithms will be ported to edge-devices. The aim of the export is for the algorithms to use micro-controller hardware to be more energy efficient and faster by parallelizing some computations. Future work will also include implementing algorithms that can handle more complex data. The algorithms currently implemented use only a single variable of measurement and assume an approximately Gaussian distribution. More algorithms will be included to work well inside these relaxed constraints. The current transmission method assumes that data is received and transmitted with no packet loss. Subsequent transmission schemes must operate under the assumption of potential packet loss.

5. ACKNOWLEDGMENTS

This work was supported by the Office of Naval Research (ONR) under contract NOs. N00014-23-C-1012, and N00014-24-C-1301, The support of the ONR is gratefully acknowledged. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of

the United States Navy. Approved, DCN# 2026-3-17-2082. DISTRIBUTION STATEMENT A. Approved for public release: distribution is unlimited.

REFERENCES

[1] Adams, R. P. and MacKay, D. J., "Bayesian online changepoint detection," *arXiv preprint arXiv:0710.3742* (2007).

[2] Aminikhanghahi, S. and Cook, D. J., "A survey of methods for time series change point detection," *Knowl Inf Syst* **51**(2), 339–367 (2017).

[3] Thompson, Z., Comert, G., Satme, J. N., Downey, A. R. J., and Bakos, J. D., "Real-time shock event classification from univariate structural response measurements," in [*ASME 2024 Conference on Smart Materials, Adaptive Structures and Intelligent Systems*], *SMASIS2024*, American Society of Mechanical Engineers (Sept. 2024).

[4] Attaran, M. and Celik, B. G., "Digital twin: Benefits, use cases, challenges, and opportunities," *Decision analytics journal* **6**, 100165–10 (2023).

[5] Tochev, E., Pfifer, H., and Ratchev, S., [*Indirect System Condition Monitoring Using Online Bayesian Changepoint Detection*], 81–92 (04 2021).

[6] Lazzaretti, A. E., Costa, C. H. d., Rodrigues, M. P., Yamada, G. D., Lexinoski, G., Moritz, G. L., Oroski, E., Goes, R. E. d., Linhares, R. R., Stadzisz, P. C., Omori, J. S., and Santos, R. B. d., "A monitoring system for online fault detection and classification in photovoltaic plants," *Sensors (Basel, Switzerland)* **20**(17), 4688 (2020).

[7] Su, Q., Li, C., Guo, X., Zhang, X., and Li, J., "Robust fault diagnosis for dc-dc boost converters via switched systems," *Control engineering practice* **112** (2021).

[8] Dempster, A. P., Laird, N. M., and Rubin, D. B., "Maximum likelihood from incomplete data via the em algorithm," *Journal of the Royal Statistical Society. Series B (Methodological)* **39**(1), 1–38 (1977).

[9] Comert, G., Rahman, M., Islam, M., and Chowdhury, M. A., "Change point models for real-time cyber attack detection in connected vehicle environment," *IEEE Transactions on Intelligent Transportation Systems* **23**, 12328–12342 (2020).

[10] Toll, W. E., "Socket programming in the data communications laboratory," *SIGCSE Bull.* **27**, 39–43 (Mar. 1995).

[11] Garcia, N. M., Gil, F., Matos, B., Yahaya, C., Pombo, N., and Goleva, R. I., "Keyed user datagram protocol: Concepts and operation of an almost reliable connectionless transport protocol," *IEEE Access* **7**, 18951–18963 (2019).