2022

A Means for Tuning Primary Frequency Event Detection Algorithms

Sean Keene
Portland State University

Landon Hanks
Portland State University

Robert B. Bass
Portland State University, robert.bass@pdx.edu

Citation Details

Published as: S. Keene, L. Hanks, and R. B. Bass, "A Means for Tuning Primary Frequency Event Detection Algorithms," in 9th IEEE Conference on Technologies for Sustainability, 2022

A Means for Tuning Primary Frequency Event Detection Algorithms

Sean Keene, Landon Hanks, Robert B. Bass
Department of Electrical & Computer Engineering
Portland State University
Portland, Oregon USA
robert.bass@pdx.edu

***Abstract*—Power system balancing authorities are routinely affected by sudden frequency fluctuations. These frequency events can precipitate cascading outages and cause damage to both customer-owned and utility equipment. In this document, we describe an Algorithm Evaluation Environment that uses a suite of metrics to evaluate an algorithm and quantify its efficacy. Using the Algorithm Evaluation Environment, a detection algorithm can be tuned to best match the definition of a frequency event as defined by experts within the context of their own balancing area. We demonstrate the utility of the Algorithm Evaluation Environment using a regression-based frequency event detection algorithm. This algorithm can detect frequency events within a short period of time after the onset of an event. The algorithm has four parameters that can be adjusted, making it highly tunable and therefore suitable for demonstration of the Algorithm Evaluation Environment.**

***Index Terms*—primary frequency response, frequency event, event detection, synchrophasor, phasor measurement unit**

I. INTRODUCTION

This document presents an Algorithm Evaluation Environment (AEE) in which a frequency detection algorithm can be tuned and evaluated against assessment from industry experts. The algorithm in question resides within a Real-time Automation Controller (RTAC) that continuously monitors data from a Phasor Measurement Unit (PMU) and looks for deviations that are indicative of a frequency event [1]. An example of such an event is shown in Figure 1. Upon detecting a frequency deviation, the algorithm generates a flag, which can then be used to initiate dispatch of an energy response asset in time to help arrest the frequency event. This regression-based frequency detection algorithm has four parameters, which are adjusted to align event detection with opinions of industry experts. If properly tuned, the algorithm rapidly detects frequency events with high accuracy and sensitivity, and with low false detection rates.

Every Balancing Authority (BA) has different tolerance to frequency events due to differences in system inertia, inter-BA connections, and intra-BA response assets [2]; as such, the definition of a frequency event varies by BA. The North American Electric Reliability Corporation (NERC) discusses frequency event characteristics extensively in its Frequency Response Initiative Report [3]. The NERC BAL-003-1 Frequency Response

This research was supported by Portland General Electric.

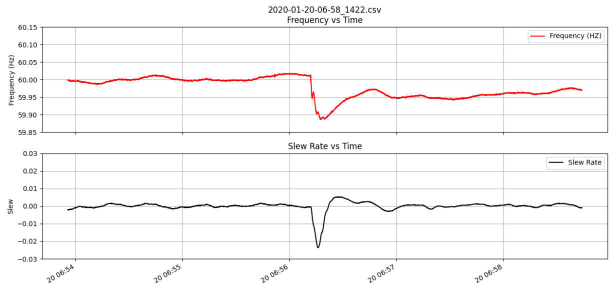

Fig. 1. An example of an under-frequency event, characterized by a rapid decrease in frequency (top, red) and a corresponding surge in the slew rate (bottom, black, a.u.)

Standard (FRS) background document also provides means for quantifying frequency events [4]. However, the FRS does not provide a universal definition of a frequency event, so it is the role of experts from individual BAs to determine what qualifies as a frequency event.

By not setting a universal definition of what constitutes a frequency event, NERC acknowledges that the stability of different BAs may be more or less sensitive to frequency events. Large continental-scale interconnections have enormous rotational inertia, making them less sensitive to sudden changes in system topology [5], [6]. On the other hand, smaller or isolated BAs, such as islands and BAs with limited regional integration, have much lower rotational inertia and therefore greater sensitivity to frequency events [7], [8]. As such, frequency response detection algorithms need to be tuned uniquely for each authority. The AEE facilitates this customization.

In this paper, we use a regression-based frequency detection algorithm to demonstrate how the AEE is used to tune algorithm parameters for the optimal detection of events, with events pre-classified by a group of industry experts. This regression algorithm evaluates the "slew rate," which we define as the rate of change of frequency as approximated using a least-sum-of-squares linear regression. The regression algorithm has four variables that can be adjusted to tune results. The four parameters are *window size*, *point separation threshold*, and *series-over threshold*, all of which are discussed and analyzed in Section II. Using the AEE, these four tuning parameters can be adjusted to achieve frequency detection results that closely match expert definitions of frequency events.

II. Frequency Event Detection Algorithm

Our frequency response testing system uses an algorithm to flag frequency events based on data from a PMU. The PMU was invented by Phadke and Thorp in the 1980s [9], [10]. PMUs provide rapid sampling of synchrophasors, which are time-stamped measurements of Steinmetz's current and voltage phasors, from which frequency may be calculated [11]. Every PMU measurement includes a time stamp that is aligned to a common time reference [12]. PMU data provide a high-fidelity representation of frequency, which the frequency response system uses to detect events in real-time.

Our regression detection algorithm is a threshold-detection algorithm that uses the Rate of Change of Frequency (ROCOF) developed from a least-squares linear regression; this rate of change is referred to as the *slew rate*, Equation 1.

$$slew\ rate = \frac{N\sum(xy) - \sum(x)\sum(y)}{N\sum(x^2) - (\sum(x))^2}, \tag{1}$$

where N is number of data points in time over which we calculate the regression (or the "regression window").

The ROCOF was selected as a detection parameter by observation: while frequency after an event may be significantly above or below the nominal value of 60 Hz, frequency events share a characteristic in that the frequency value suddenly and precipitously changes. As such, the rate of change is more useful than frequency alone. Since the rate of change is the value of interest, only the slope of the regression is required.

The purpose of using a linear regression rather than the derivative of frequency is to compensate for noise in the PMU data. Noise is more pronounced when reading synchrophasors at 30 measurements per second, in contrast to the one to four second rate of a SCADA system. Micro-deviations in frequency abound at this sampling rate. These small deviations cause the derivative of frequency to vary considerably between positive and negative on a point-by-point basis rather than smoothly varying with the overall frequency change over time, making threshold detection difficult. Linear regression produces a smoothed derivative of frequency data.

Linear regression is effective at minimizing the influence of noise on the outcome of the algorithm; however, it introduces a time delay into the response since the regression must be calculated over a window of time. For a given period, new values indicative of an event will initially be overwhelmed by the prior values that did not indicate an event. This extends the amount of time between the initiation of the frequency excursion and the point that the calculated slew rate crosses the *slew rate threshold*. To minimize this delay, the detection algorithm relies on the characteristics of the initial slew rate change to declare an event, rather than waiting for a threshold to be crossed. Following are the four tunable parameters of the regression algorithm:

A. Window Size

Slew rate is calculated using PMU data over a predetermined number of data points, called a "window," with a length N being the *window size*. A larger window results in a smoother output function, but a longer time delay.

B. Event Parameter Threshold

The algorithm uses a threshold, called the *event parameter threshold*, to indicate the possible beginning of an event. This threshold is a slew rate value (positive or negative) that, if exceeded, indicates a rapid change in frequency.

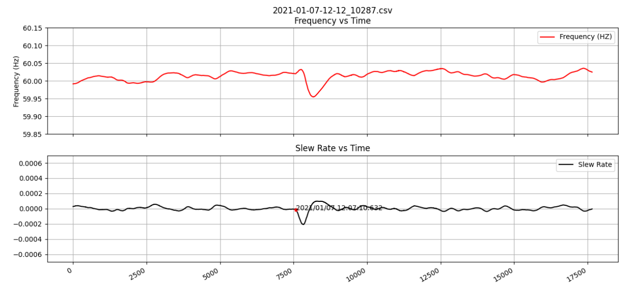

Fig. 2. The *point separation threshold* has been exceeded, red dot, near the beginning of the frequency event.

C. Point Separation Threshold

The algorithm takes into consideration magnitude differences between adjacent data points in the slew data, called the "points separation." An instance where the *point separation threshold* was exceeded is shown as the red dot in Figure 2.

D. Series-Over Threshold

The algorithm monitors how many times in a row that the points separation exceeds the *point separation threshold*. This is called the "series-over" value. If the series-over value exceeds the *series-over threshold*, then the algorithm declares an event. Figure 3 show tracking of series-over values, represented by green dots.

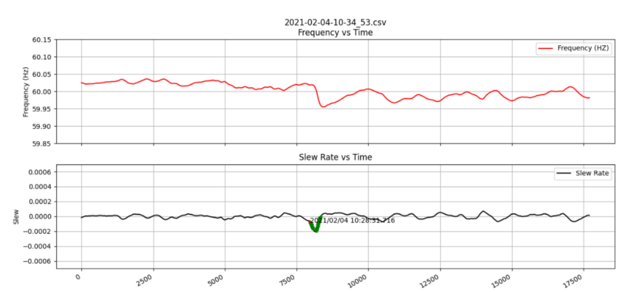

Fig. 3. The *point separation threshold* has been exceeded, red dot, multiple times, green dots, demonstrating the *series-over* detection.

The algorithm characterizes frequency deviations using three parameters: the *event parameter*, *point separation* and *series-over* thresholds. These provide the operator with four degrees of freedom when tuning the algorithm, including the *window size*.

III. Algorithm Evaluation Environment

Prior to being deployed to detect events in real-time, an algorithm must be validated against expert assessments. This is done within the AEE. The flowchart in Figure 4 describes the workflow of the AEE.

To facilitate this validation, we compiled an archive of sample PMU frequency plots, referred to as the Frequency Event Archive (FEA). These plots are a subset of our archive of PMU data. The original PMU archive containes over two years of frequency measurements. To construct the FEA, we obtained timestamps of frequency events from our utility partner, Portland General Electric, as well as from a list of frequency events identified by NERC. The event files containing data at these timestamps were then extracted from the PMU data archive. These event files were compiled along with samples of non-events and near-events to form a set of example files, the FEA. These example files were then reviewed and evaluated by a group of "experts" consisting of the project team members.

The FEA consists of 135 PMU data files. Each data file contains five minutes of PMU frequency data, sampled at 30 frames per second. As shown at the top of the AEE flow diagram, Figure 4, the FEA files are evaluated by both industry experts and an algorithm. The manual evaluations and algorithm classifications are then compared to determine the efficacy of the algorithm. Once tuned to best match expert assessments, the updated algorithm parameters are then transferred to the RTAC for real-time event detection. This process is generalizable to any frequency event detection algorithm, not just the regression algorithm presented in this paper.

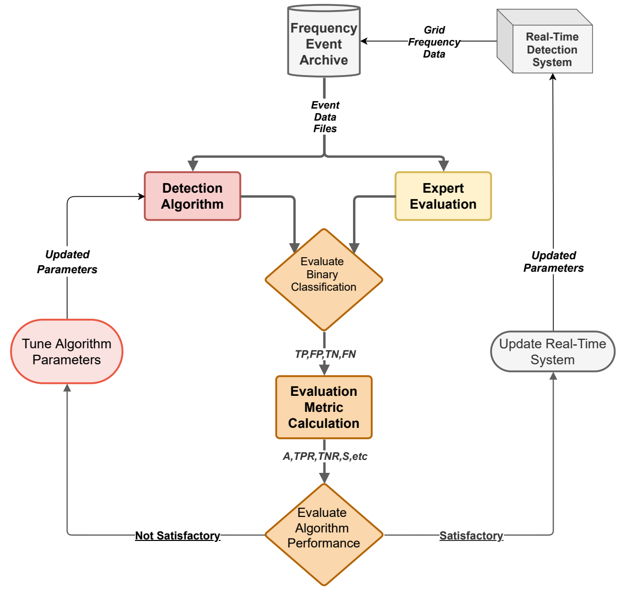

Fig. 4. The Algorithm Evaluation Environment (AEE) flow chart. The validity of an algorithm depends on how well its outputs compare to the assessments of experts.

A. Frequency Event Archive

A set of visualization tools are used to analyze events within the PMU archive and identify example PMU data files. Some of the files within the FEA show obvious frequency events. The list of NERC events included 29 such frequency events. Of these, we found 27 events within our PMU data archive; two were missing from the archive as they occurred during PMU outages or maintenance downtime. Additional frequency events were found by combing through the PMU data archive; these were also added to the FEA.

The FEA also includes examples where no event is present. These non-events are included within the FEA to test algorithm susceptibility to false positives and its ability to exclude true negatives. Aside from clear events and non-events, also included within the FEA are more ambiguous near-events, which are more challenging for both the industry experts and the algorithms to identify. These cases show marginal frequency fluctuations that may or may not be identified as frequency events, depending on the expert evaluator. Figures 6 and 5 show examples of a non-event and a near-event, respectively

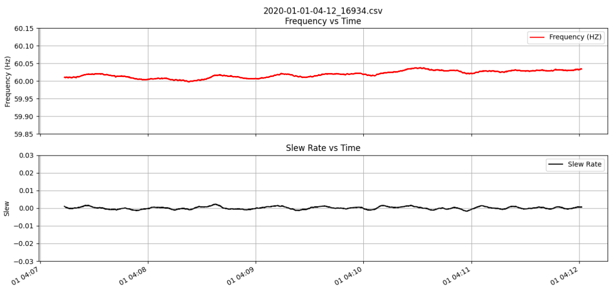

Fig. 5. Example of a non-event. Including non-events tests algorithm identification of false positives and exclusion of true negatives.

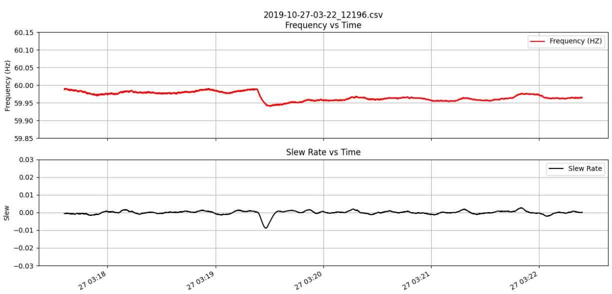

Fig. 6. Example of a near-event. The ambiguous characteristics of such events challenge expert opinions and algorithm efficacy.

B. Event Identification

As depicted on the AEE flowchart, an industry expert identifies each file from the FEA as either an event or a non-event; the algorithm evaluates and identifies the same data set. Some criteria that evaluators can use to indicate an event include rapid departure from expected frequency, a large difference between initial and final frequency, and sudden deviations in the rate of change of frequency. If a file is determined to show an event, both the expert and the algorithm assess the event as either as “under-frequency” or “over-frequency” event.

C. Industry Expert Evaluation

The process for compiling industry expert assessments of frequency events uses an online assessment survey. The survey sequentially presents the FEA events to an expert and records their responses within a database. A screen shot of the survey is shown in Figure 7.

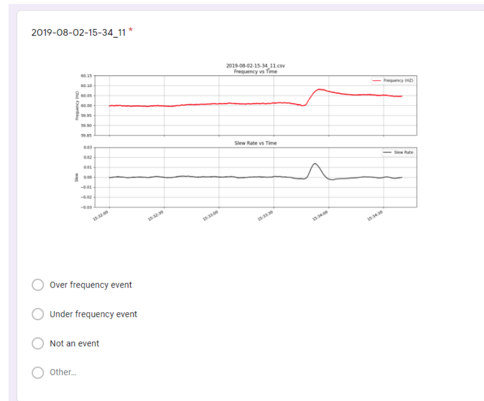

Fig. 7. The online survey interface. An example of an assessment question.

The survey prompts the expert for their name and an estimate of their relative expertise at assessing frequency events on a scale from one to five. This estimate is later used to weigh the expert’s assessments against those of other experts. Then, the survey presents the expert with candidate frequency event plots that show both frequency and slew rate. The survey prompts the expert to assess each candidate as either an over-frequency event, an under-frequency event, or a non-event. This continues until the expert has assessed all the files within the FEA.

After the survey is complete, the data are processed by a spreadsheet tool. Users’ assessments are weighted by their self-reported expertise level. The weighted assessments for each event file are summed. The weighted value represents the relative sureness of the assessment.

D. Binary Classification

After both the expert and algorithm event identifications have been completed for all files, the algorithm results are classified against the expert assessments using a binary classification scheme. The classification metrics are True Positive (TP), True Negative (TN), False Positive (FP), or False Negative (FN):

- TP: the algorithm and the expert both agree an event occurred.
- TN: the algorithm and the expert both agree an event did not occur.
- FP: the algorithm declared an event that the expert did not identify as an event.
- FN: the algorithm did not declare an event that the expert identified as an event.

E. Evaluation Metrics

The AEE uses the binary classification metrics to calculate evaluation metrics. Evaluation metrics quantify the efficacy of an event detection algorithm in relation to the assessment of the industry experts. The evaluation metrics are *Accuracy*, *Sensitivity*, *Precision*, *Specificity*, and *False Discovery Rate (FDR)*, which are defined as follows [13].

Accuracy measures the ability to correctly identify data sets containing events and data sets containing no events.

$$Accuracy = \frac{TP + TN}{SetSize} \times 100\% \tag{2}$$

Sensitivity measures the ability to correctly identify frequency events.

$$Sensitivity = \frac{TP}{TP + FN} \times 100\% \tag{3}$$

Precision measures how many of the positively identified events were true positives.

$$Precision = \frac{TP}{TP + FP} \times 100\% \tag{4}$$

Specificity measures the ability to correctly identify events. It is similar to *Sensitivity*, though it considers the ratio of TNs to the number of total negatives.

$$Specificity = \frac{TN}{TN + FP} \times 100\% \tag{5}$$

FDR measures the propensity to erroneously identify an event. FDR is equivalent to $1 - Precision$.

$$FDR = \frac{FP}{FP + TP} \times 100\% \tag{6}$$

Accuracy, *Sensitivity*, *Precision*, and *Specificity* range from 0% to 100%, with ideal values of 100%. FDR also ranges from 0% to 100%, but has an ideal value of 0%.

IV. Algorithm Analysis

To analyze the regression algorithm parameters and their effects on the evaluation metrics, around 20 test runs were conducted per parameter. Only one parameter was adjusted at a time while all others were held constant.

A. Analysis: Window Size

A change in *window size* correlates to a significant change in the evaluation metrics, Figure 8. As the *window size* increases, the FDR decreases while the other four evaluation metric values converge at about 80%. However, detection time increases as window sizes become larger.

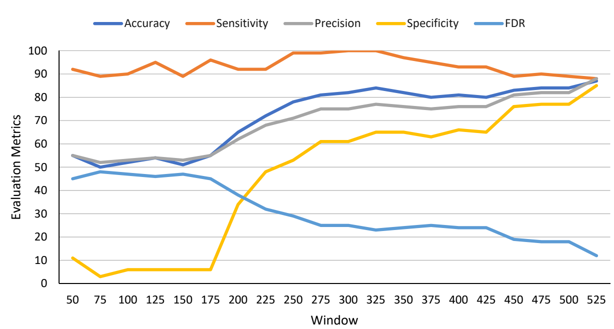

Fig. 8. Evaluation metrics as a function of *window size*. *Accuracy*, *sensitivity*, *precision*, and *specificity* have an ideal value of 100%, while False Discovery Rate is ideally 0%. For this data set, the *event parameter threshold* is $6.5 \cdot 10^{-6}$, the *point separation threshold* is 15, and the *series over threshold* is 18 (a.u).

B. Analysis: Event Parameter Threshold

Analysis of the *event parameter threshold*, Figure 9, shows the evaluation metrics converging at around 85% for values around $1.1 \cdot 10^{-6}$. The evaluation metrics deviate sharply for lower *event parameter threshold* values.

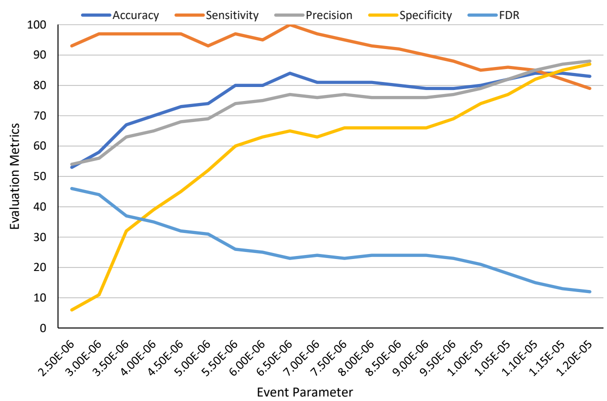

Fig. 9. Evaluation metrics as a function of *event parameter threshold*. For this data set, the *window size* is 325, the *point separation threshold* is 15, and the *series over threshold* is 18.

C. Analysis: Point Separation Threshold

Analysis of the *point separation threshold*, Figure 10, shows the evaluation metrics converging at around 80% for separation values between 8 and 9. The evaluation metrics deviate sharply for both lower and higher *points separation threshold* values.

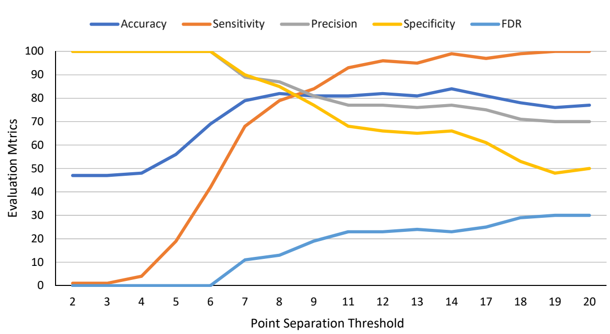

Fig. 10. Evaluation metrics as a function of *point separation threshold*. For this data set, the *window size* is 300, *event parameter threshold* is $6.5 \cdot 10^{-6}$, and the *series over threshold* is 18.

D. Analysis: Series-Over Threshold

Analysis of the *series-over threshold* parameter, Figure 11, shows that the evaluation metrics were largely insensitive to variation in this parameter. This may be due to the influence of the other parameters. In this test, *window size* was set to 300 and *point separation threshold* was set to $6.5 \cdot 10^{-6}$. These values are quite high, and it is plausible that a smaller *window size* or *point separation threshold* would lead to more pronounced effects due to *series-over threshold* variations.

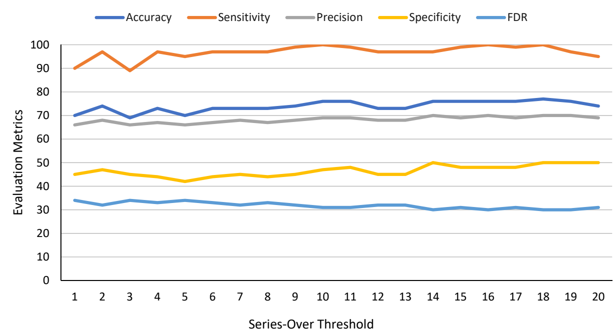

Fig. 11. Evaluation metrics as a function of the *series-over threshold*. For this data set, the *window size* is 300, *event parameter threshold* is $6.5 \cdot 10^{-6}$, and the *point separation threshold* is 20.

V. Future Work

There are several avenues for further development. For instance, the process of tuning a frequency detection algorithm should be automated and optimized. Since the algorithm has four tunable parameters, a hypervolume of suitable solution sets likely exists. However, considering the size of the search space, finding suitable solution sets within this four-dimensional space is a challenge. Manually tuning these parameters to find an optimal set would be time consuming and likely unsuccessful. An automated optimization process would be preferred. We are currently developing an optimization process, using a Grey Wolf optimizer, that has demonstrated reasonable time to convergence for acceptable solutions [14].

Once an algorithm has been tuned, testing is required prior to deployment. Algorithms should be tuned and tested against real-time frequency events within a physical implementation. However, testing against live grid data is inefficient due to the infrequency of events and the inability to control their magnitude, time duration, and other relevant parameters. Our real-time frequency response testing system, Figure 12, provides controlled testing capabilities.

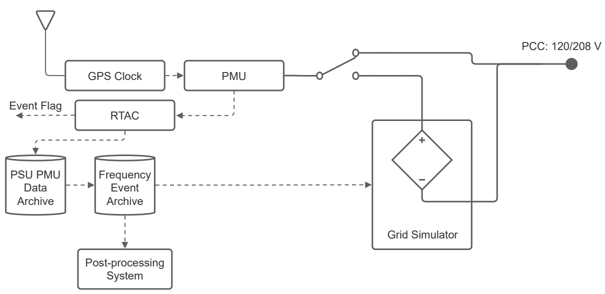

Fig. 12. Representation of a real-time frequency response testing system

The system consists of an SEL-3555 RTAC, an SEL-351 PMU, an SEL-2407 GPS clock, and a GPS antenna, which comprise the Real-time Frequency Response Testing System; the PMU event archive, which contains the archived PMU frequency events; and an NHR 9410-12 four-quadrant grid simulator, which provides a means for emulating specific voltage and frequency grid conditions. This testing system provides programmable, customized real-time simulations of frequency events, including recreations of frequency events from the FEA. This allows algorithms to be programmed into the RTAC and rapidly tested against historical events in real-time, eliminating the need to wait for events to occur stochastically on the grid.

VI. Conclusion

This document presents a process for evaluating the ability of an algorithm to detect grid-significant frequency events. The Algorithm Evaluation Environment uses an event archive consisting of 135 PMU data sets to quantify the ability of an algorithm to reproduce the observations of industry experts. The event archive includes examples of frequency events, near-events, and non-events. Industry experts evaluate these examples and classify each as either an event (over- or under-frequency) or a non-event. This same set of events is then evaluated by an algorithm, which also classifies each as either an event or a non-event. These classifications are then compared and validated by a set of evaluation metrics. The algorithm parameters can then be tuned to improve the metrics. Once the algorithm has been tuned, it may then be used in a real-time event detection system.

The AEE was demonstrated using a frequency event detection algorithm based on a least-squares linear regression that produces a smoothed rate of change of frequency, or *slew rate*. Once calculated, the algorithm monitors the slew rate behavior to sense when frequency deviations begin to occur. The algorithm has four parameters that can be tuned to improve agreement between the algorithm output and manually-classified expert assessments. Algorithm behavior was analyzed by sweeping each parameter while holding the other parameters constant. Adjustments to the *window size* and *point separation threshold* parameters each significantly affected the detection performance. However, the *series-over threshold* parameter had only a marginal effect on the detection performance, though this is likely due to the tuned values of the other parameters.

References

[1] "IEEE standard for synchrophasor data transfer for power systems," *IEEE Std C37.118.2-2011 (Revision of IEEE Std C37.118-2005)*, pp. 1–53, 2011.

[2] "Frequency response issue paper," California Independent System Operator, Tech. Rep., August 2015.

[3] "Frequency response initiative report: The reliability role of frequency response," North American Electric Reliability Corporation, Tech. Rep., October 2012.

[4] "Frequency response standard background document," North American Electric Reliability Corporation, Tech. Rep., November 2012.

[5] N. W. Miller, M. Shao, R. D'aquila, S. Pajic, and K. Clark, "Frequency response of the US Eastern Interconnection under conditions of high wind and solar generation," in *IEEE Green Tech. Conf.*, 2015, pp. 21–28.

[6] "Underfrequency load shedding assessment report, Western Interconnection," Underfrequency Load Shedding Review Group, Western Electricity Coordinating Council, Tech. Rep., December 2013.

[7] L. C. Dangelmaier, "System frequency performance of the Hawaii Electric Light System," in *IEEE Power and Energy Soc. General Meeting*, 2011, pp. 1–8.

[8] W. Y. Choi, K. S. Kook, and G. R. Yu, "Control strategy of BESS for providing both virtual inertia and primary frequency response in the Korean power system," *Energies*, vol. 12, no. 21, 2019.

[9] A. Phadke, "Synchronized phasor measurements - a historical overview," in *IEEE PES Asia Pacific T & D Conf. and Exhibition*, vol. 1, Oct 2002, pp. 476–479 vol.1.

[10] J. De La Ree, V. Centeno, J. Thorp, and A. Phadke, "Synchronized phasor measurement applications in power systems," *IEEE Trans. Smart Grid*, vol. 1, no. 1, pp. 20–27, 2010.

[11] C. P. Steinmetz, "Complex quantities and their use in electrical engineering," in *Proc. American Institute of Electrical Engineers*, Chicago, IL, 1893, pp. 33–74.

[12] E. O. Schweitzer, D. Whitehead, G. Zweigle, K. G. Ravikumar, and G. Rzepka, "Synchrophasor-based power system protection and control applications," in *2010 Modern Electric Power Systems*, 2010, pp. 1–10.

[13] T. Fawcett, "An introduction to ROC analysis," *Pattern Recognition Letters*, vol. 27, no. 8, pp. 861–874, 2006.

[14] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey Wolf optimizer," *Advances in Engineering Software*, vol. 69, pp. 46–61, 2014.