AN ABSTRACT OF THE THESIS OF

William Daigle Stark for the degree of Master of Science in Electrical and Computer Engineering presented on June 9, 2020.

Title: Computing Substation Fragility as a Function of Seismic Component Failure

Abstract approved: _____

Ted K.A. Brekken

The electrical grid of the Western Interconnection is vulnerable to earthquake damage, especially large-magnitude megathrust events, due to the unique seismic profile of the region. However, the size of the Western Interconnection makes it difficult to model seismic failure in electrical substations to the level of detail necessary to improve existing protection. Hazus data, which represents the failure methods used on the national level, uses a calculation of failed components as a proportion of total. This is useful for repairs but can be misleading when analyzing power capacity, as it does not take into account component-level redundancy and configuration variations between substations. This research creates code which translates component-level fragility data sourced from local utilities into failure curves for a variety of common substation configurations. Two forms of analysis are then performed. Power capacity failure analysis works with power flow simulations

to determine if the substation can continue to function during an event. The component failure methodology uses the same logic as existing Hazus data to directly compare the code's component-based failure against the current standard. Further work has been done to allow for component-based failure to be directly implemented into existing code for work currently being done at OSU. This report will cover up to the current state of the code and discuss limitations and next steps. As this is mainly intended as an intermediary function for use in larger systems, the output is mainly being assessed for accuracy and ease of use. As it currently stands, both the standalone and integrated versions of the code are fully workable and can translate any given component failure data into substation level fragility reports.

Computing Substation Fragility as a Function of Seismic Component Failure

by

William Daigle Stark

A THESIS

submitted to

Oregon State University

in partial fulfillment of
the requirements for the
degree of

Master of Science

Presented June 9, 2020
Commencement June 2021

Master of Science thesis of William Daigle Stark presented on June 9, 2020.

APPROVED:

Major Professor, representing Electrical and Computer Engineering

---

Head of the School of Electrical Engineering and Computer Science

---

Dean of the Graduate School

I understand that my thesis will become part of the permanent collection of Oregon State University libraries. My signature below authorizes release of my thesis to any reader upon request.

William Daigle Stark, Author

ACKNOWLEDGEMENTS

I would like to thank Dr. Ted Brekken for being my major advisor and providing guidance and encouragement throughout my time at Oregon State University. You first offered me a path to reaching this point, and it is thanks to you that I have arrived here. As well, Dr. Eduardo Cotilla-Sanchez, whose enthusiasm is infectious and has provided help and support more times than I can count.

Thank you also to Dr. Michael Olsen and Dr. Armin Steudlein, for not only serving on my committee but for helping with the Earthquake Resiliency of the Western Interconnection project. Also to David Glennon and Eve Lathrop for working alongside me on this project and throughout my time in the program.

My warmest thanks to everyone in the Energy Systems group, for your support and camaraderie the last few years. You are all first-rate scholars, engineers, people, and friends.

To Mom and Dad and Luke, for your love and support. From the first day to this one, thank you.

Finally and always, Robyn. Without you, none of this would have been possible.

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

LIST OF FIGURES

LIST OF FIGURES (Continued)

LIST OF TABLES

LIST OF APPENDIX FIGURES

Chapter 1: Introduction

1.1 Project Description

Earthquake Resilience of the Western Power Grid is a National Science Foundation (NSF)-funded project currently being undertaken by Oregon State University (OSU) with the aim of creating a model of the Western Interconnection that is capable of estimating responses to a large-scale seismic disaster. One of the principle challenges of such a model is the size of the Western Interconnect, over 40,000 individual buses, where a bus is here defined as an electrical substation or specialized grid component. Even using a synthetic reduced-node model, at least 10,000 buses will need to be analyzed.

The work in this thesis is a part of that overall project, namely creating substation models that can incorporate real-world component data while still being accessible to larger simulations. This will allow for greater flexibility going forward, as well as providing a method to compare various fragility functions.

1.2 Statement of Purpose

This thesis represents a presentation of code designed to bridge the gap between component and high-level substation analysis. This code has been developed and implemented in two ways. One is designed to be a robust and flexible stand-alone

that is both proof of concept and troubleshooting, while the other is a much smaller implementation that can be directly cross-applied to existing simulations.

The primary branch takes component fragility data, constructs a generic substation based on feeder and generator criteria, and then tests this synthetic substation for failure along a range of PGAs. This data is then compared to results from Hazus, a software package developed by FEMA, for both accuracy and the possibility of using this thesis' data in place of Hazus' own. Power capacity failure is also tested, providing information that could be useful for power flow analysis.

The secondary branch is reduced-scale, modeling single lines and transformers, for use in the synthetic WECC model. This code serves only as a part of the larger synthetic model, and undergoes no analysis of results beyond simple accuracy checks.

The Conclusions section includes comparisons to Hazus data, discussions on causes of fragility for substations from a power capacity failure standpoint, and next steps for the code.

1.3 Earthquakes and the Seismic Profile of the Pacific Northwest

A large number of rigid tectonic plates compose the Earth's lithosphere, or outermost sub-atmospheric layer [7]. These plates generate significant forces along the regions where they interact. These interactions can be generally classified in one of three ways:

- **Divergent Boundary:** Two plates moving apart, causing *rifting*, where

new crust is generated from the mantle.

- **Convergent Boundary:** Two plates moving into each other, causing various types of *subduction*, where one plate slides under the other.

- **Transform Boundary:** Two plates moving along each other, with crust being neither created nor destroyed.

The Pacific Northwest region of the United States is located in an area of significant seismic interest [8]. Two major tectonic plates, the Pacific Plate and the North American Plate, are moving along a transform boundary. Meanwhile, the minor Juan de Fuca Plate is being subducted under the North American plate, an action which is responsible for the Cascade Range, the Pacific Ranges, and the Cascade Volcanic Arc [1]. The fault itself, which stretches offshore from Northern Vancouver Island in British Columbia to Cape Mendocino in California, has been termed the Cascadia Subduction Zone [9].

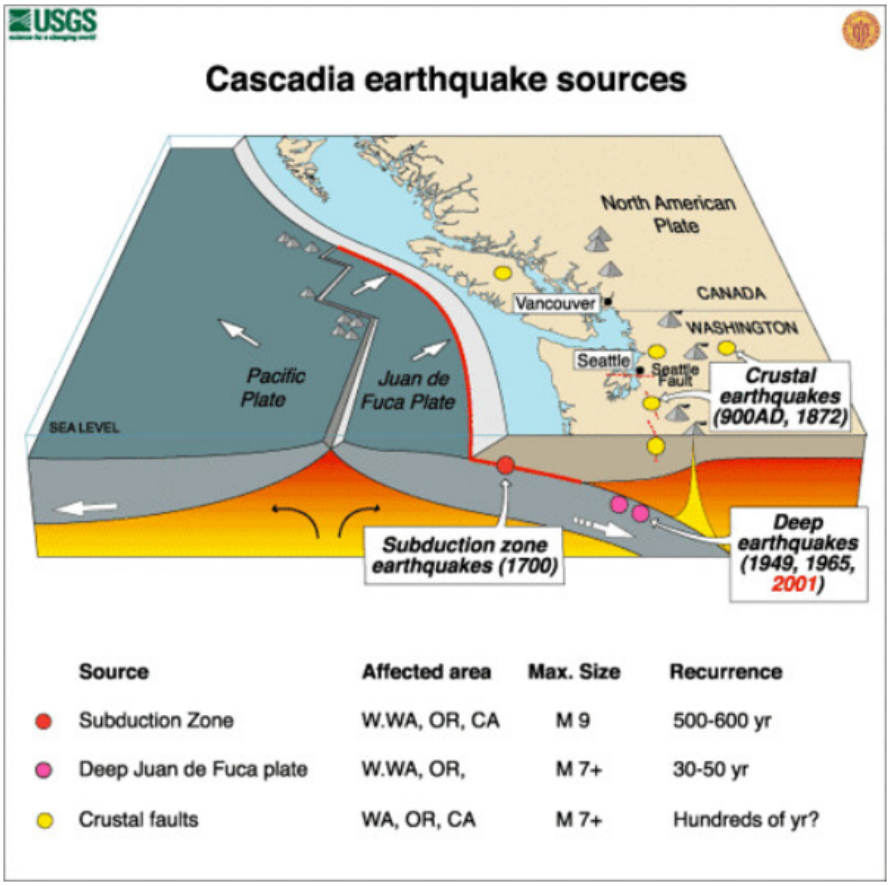


Figure 1.1: Cascadia Subduction Zone and surroundings [1]

The Cascadia Subduction Zone is one of the planet's highest probability spots for a high-intensity megathrust earthquake, due to a number of geophysical characteristics found at the Juan de Fuca/North American fault [10, 1, 8]. Although it is difficult to predict the intensity of earthquakes with any real certainty, the region has characteristics that could allow for a seismic event with an upper limit comparable to the 2011 Tohoku Earthquake or the 2004 Indonesian Earthquake [11].

In certain estimated scenarios, the damage to infrastructure caused by both the earthquake itself and the resulting tsunami would be catastrophic [4]. This report will be looking at generalized seismic events, but given the likelihood of a single major event, certain protection and recovery characteristics should be prioritized.

1.4 Earthquake Impacts on Lifeline Systems

Much of our modern infrastructure is vulnerable to a variety of disasters and mishaps, but earthquakes and other seismic events are capable of causing widespread damage and cascading failures. No matter what the nature of the infrastructure, each has its own points of failure and specific challenges to restoration.

If the immediate aftermath of an earthquake, even transportation to and from affected areas can be difficult or impossible. Roads and highways can be damaged or rendered completely unusable for long stretches, and rail transit can be made impossible if damage to the rails exceeds the relatively narrow limitations needed for a locomotive engine to use it. Bridges and inclines are especially vulnerable, and represent crucial points of failure in any road or rail system [12]. Fortunately, in most cases emergency supplies and personnel can find some way to reach an area, whether by land, air, or sea. Although damage to airports and airstrips is likely to make airplane travel to these regions sporadic at best, the use of helicopters allows for point-to-point contact even in the event of catastrophic infrastructure failure. Particularly, certain regions of the Oregon Coast are especially vulnerable to isolation, as a relatively small number of narrow roads crossing rough terrain

provide the only routes to travel in or out of these communities [4].

More complicated than transit is the flow of water and natural gas through pipelines, either above or below the ground. Because these pipes are pressurized, any rupture can result in significant spills, leading to not only loss of function but also environmental contamination and possibly damage to other nearby infrastructure. Damage can be difficult to locate, depending on the quality of the monitoring hardware, and costly to repair due to remote locations and specialized materials [5, 13].

Electrical infrastructure is more delicate than either transit or pipeline networks [14], but consequentially is designed to posses a far greater level of monitoring and protection. Ideally, important transmission and distribution systems are monitored in real-time, so that any fault or failure can be immediately reported. However, in practice many areas are dependent on indirect monitoring or breakers, which can prevent catastrophic damage but do not generally provide the same level of real-time monitoring. In the case of a widespread disaster like an earthquake, large portions of the grid could fail, leading to cascading outages and overloading the system's protective measures. For this reason, prioritizing key lines for reinforcement and having an accurate measure of vulnerabilities is key to grid protection.

1.5 Electrical Substations

Due to the complexity of the grid, protection considerations, and the inherent characteristics of electrical power, in most cases it is not advisable to directly

connect electrical generation and load. To this end, around 40,000 substations service the Western Interconnection, the synchronous grid that covers the western half of the United States. These substations perform such varied functions as protection, voltage regulation and switching, power factor correction, and relaying and monitoring.

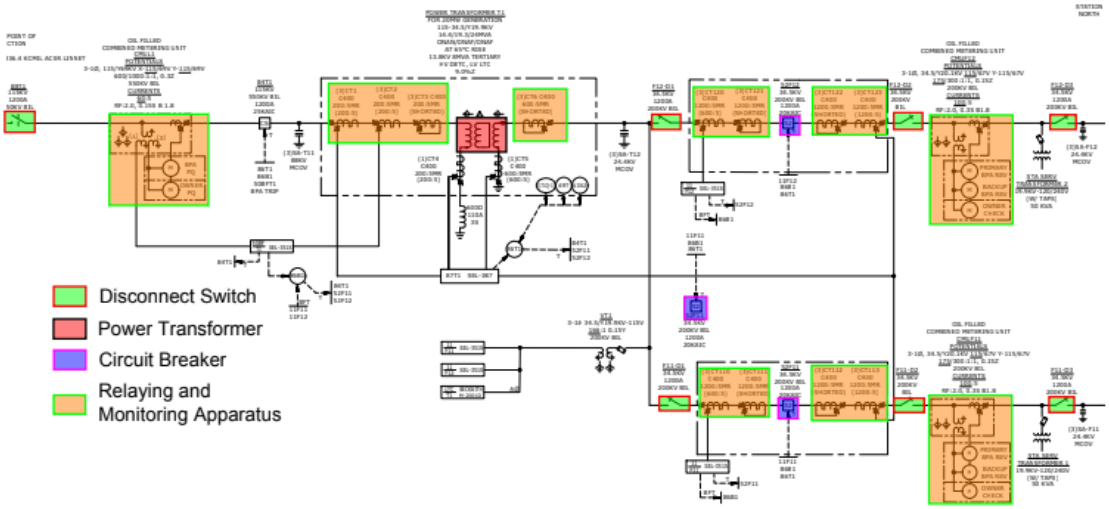


Figure 1.2: Substation detail, components highlighted. Provided with permission of DEA, Inc.

Fig. 1.2 shows that a significant amount of the substation's infrastructure is dedicated to relaying and monitoring, and the actual power flow protection involves a number of redundant components. Each one is vital, but also represents a possible failure point in the event of seismic activity. There are numerous configurations and permutations found in substation design, but a few key components are almost always present.

- **Disconnect Switch:** Physically isolate a line or component for repair or inspection. Not designed to respond to fault conditions.

Figure 1.3: Disconnect Switch, 115 kV[2]

- **Circuit Breaker:** Interrupt the flow of current when fault conditions occur. Usually connected to relaying apparatus for monitoring and communication. A common alternative that performs the same function is the circuit switcher.

Figure 1.4: Circuit Breaker, Gas, 57 kV[2]

- **Transformer:** Step voltage up or down, allowing transfer between transmission and distribution lines at different voltage levels.

Figure 1.5: Transformer, 57 kV[2]

- **Relaying and Monitoring Apparatus:** A variety of current transformers, potential transformers, sensors, control systems, and communication devices that track any abnormalities or fluctuations in substation operation.

Figure 1.6: Current Transformer, 57 kV[2]

- **Control Enclosure:** The building in which all complex electrical and computerized components reside. Isolated from the active substation lines, and the initial human interface point for the substation as a whole.

Figure 1.7: Substation Control House Exterior and Interior[2]

Because electrical substations are a key part of the electrical grid, their protection is essential to energy security. This report will analyze substations as a collection of components, providing a level of detail that not only fills a crucial gap in existing fragility analysis, but also works within larger projects currently being undertaken to model the Western Interconnect as a method of identifying vulnerabilities and taking proactive steps to protect them.

Chapter 2: Key Materials

2.1 GEER Anchorage

Anchorage makes a useful testbed for seismic protection and detection schemes, both due to its isolation and the frequency with which seismic events occur. Over 280 sensors of various sorts have been set in place to monitor seismic events and record as much data as possible. Furthermore, the Anchorage electrical grid is largely not connected to other regions, meaning that it can be viewed as a network of its own [3].

A Magnitude 7.1 earthquake occurred on November 10, 2018 along the subducting Pacific Plate [3], sending tremors through the greater Anchorage area. This event can serve as an indicator of current methodologies for earthquake reporting, and show gaps in how electrical systems are integrated into these large-scale reports.

From an electrical network standpoint, reports like the GEER analysis for Anchorage focus on areas other than substation and transmission resiliency. Only 1 page out of 90 pages of analysis is devoted to utilities [3].

The gap between the utility-level monitoring and the extensive protection models used for regional protection and vulnerability detection represents a key gap in coverage for analysis of the electrical grid. This project aims to provide the level of

Figure 2.1: Non-exclusive map of available seismic monitoring hardware in downtown Anchorage, AK. [3]

connectivity that would allow local utilities to communicate effectively with organizations like GEER that examine regional seismic activity, and increase coverage and cross-pollination of data.

2.2 Oregon Resilience Plan

The Oregon Resilience Plan is a report presented to the 77th Legislative Assembly for the State of Oregon in February of 2013 by the Oregon Seismic Safety Policy Advisory Committee (OSSPAC)[4]. It aims to be a comprehensive analysis of the impact that a Cascadia Subduction Zone megathrust earthquake could have on the infrastructure, people, and economy of the state of Oregon.

Of particular interest to this thesis is Chapter 6, titled *Energy*. Notable herein is the lack of any specific seismic codes for Critical Energy Infrastructure (CEI) in situations where such codes exist for other types of buildings and infrastructure. The chapter lays out some general observations on vulnerabilities in energy infrastructure, and how a megathrust event from the CSZ would be different from the smaller regional events that many seismic plans anticipate.

Notably, a CSZ megathrust event would cause damage widespread enough that it would disrupt existing disaster relief supply chains, greatly increasing the time it would take to effect repairs on electrical systems [4].

The current estimate for these sites puts substations at the most vulnerable to damage, and taking the longest time to repair. Further, substations in the coastal region (outside of the tsunami impact zone, which adds an additional level of damage beyond the scope of this project) suffer from the possibility of remote location from major supply chains. In areas where the majority of infrastructure lies along the Pacific coast, it is feasible that the damage could outstrip available resources by several orders of magnitude. In addition, because the coast is relatively lightly

populated, reinforcing those parts of the infrastructure tends to be of lower priority [4]. In conjunction with the damage cause to transportation methods by an y earthquake of significant size, it could be up to 6 months before all parts of the coast have power restored.

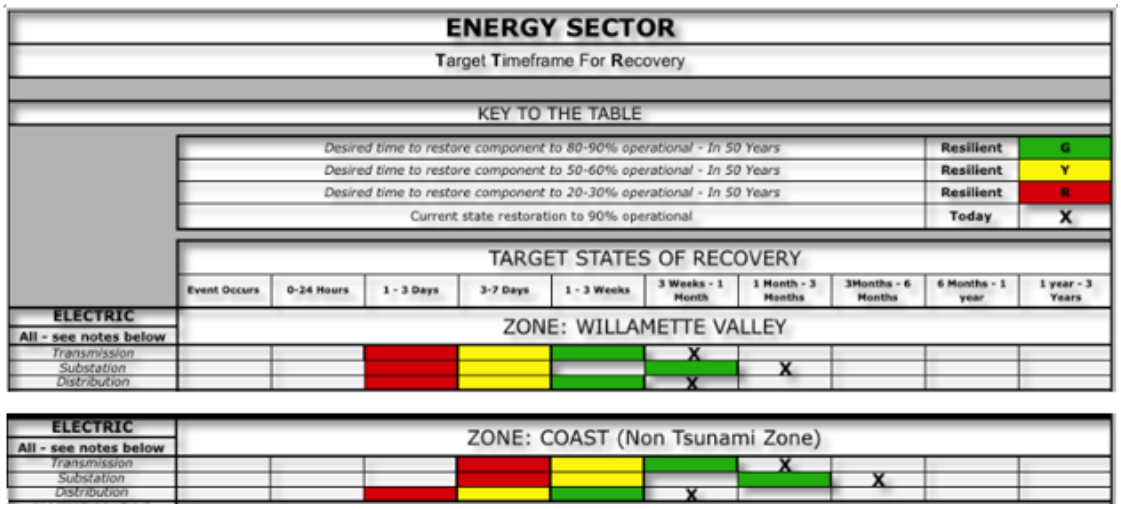


Figure 2.2: Estimated recovery time for electrical systems. [4]

The findings of the Oregon Resilience Project indicate a high level of vulnerability among energy utilities, especially those in rural areas. Work like this thesis aids in highlighting the most critical points where reinforcement should be prioritized, and could be used to allocate funds more efficiently. Regardless of the exact method used, the ORP should be recognized as providing examples of situations where this research can be applied to tangible effect.

2.3 SEFT Report and component-level analysis

Portland General Electric (PGE) commissioned a document intended to identify crucial failures among their equipment in the greater Portland area. This report, created by SEFT Consulting, is responsible for the component-level data that will be used in this report.

Each component is examined, has an SAP2000 model created, and is submitted to eigenvalue analysis under simulated seismic conditions. Motion is registered along x-, y-, and z-axes at key failure points, and compared to given tolerances. From this, failure points are calculated and a fragility curve is generated for that component.

SEFT analyzes three voltage levels for this report, and provides a set of components at each. Table 2.1 shows a complete listing of all components gathered from the SEFT report.

Table 2.1: SEFT Components by Voltage Level [2]

57 kV	115 kV	230 kV
Circuit Breaker, Gas	Circuit Breaker, Gas	Circuit Breaker, Gas
Circuit Breaker, Oil	Circuit Breaker, Oil	Circuit Breaker, Oil
Circuit Switcher	Circuit Switcher	-
Switch, Disconnect	Switch, Disconnect	Switch, Disconnect
Transformer, Instrument	Transformer, Instrument	Transformer, Instrument
Transformer, LTC Power	Transformer, LTC Power	Transformer, LTC Power (I)
-	Transformer, non-LTC Power	Transformer, LTC Power (II)
Control House	Control House	Control House

Each of these curves is a useful analysis and serves as vital information for the utility, but can do very little for the large-scale analysis that is vital for finding the vulnerability of the Western Interconnection.

2.4 HAZUS and node-level analysis

Hazus is software made available by the Federal Emergency Management Agency (FEMA) for disaster analysis, and serves as a vital tool for many state and local agencies when examining the potential damage to infrastructure that could be caused by any number of various disasters. Their analysis of the electrical grid views each generator, substation, and distribution network as a node, and gives

each node various states of failure [15].

For electrical substations, Hazus views each node as falling into one of five damage states: none, slight, moderate, severe, and complete. This is calculated mostly based on a tally of failed components as a percentage of all components present. The three major areas examined in this way are switches, circuit breakers and transformers. If five percent of switches fail, the entire substation is considered slightly damaged. Likewise for circuit breakers and transformers, though failure rates below that would register identically to an undamaged substation. Thresholds for moderate, severe, and complete damage are at 40%, 70%, and 100% respectively[5].

Hazus data, while useful for its intended purpose, has the notable flaw of not taking power capacity into consideration in any manner. For the purposes of a simulation that is entirely concerned with power flow in the aftermath of an earthquake, this is less than ideal. Consequentially, this thesis explores a method of bringing component-level data up to the flexibility of Hazus' substation fragility functions, and allow it to be used with additional tools that Hazus provides.

Chapter 3: Modeling Substation Failure

3.1 Fragility Functions

A common analysis tool used for indicating the probability of failure of a component, element, or systems due to seismic events is the fragility function [16]. This is typically represented as a 2-dimensional plot that shows the probability of a component being in a certain damage state as a function of energetic input. Peak ground acceleration, often shortened to PGA, is the measure of greatest change in velocity during an earthquake, and is a useful shorthand for the direct damage-causing ability that a seismic event possesses at a given location.

For this thesis, most results will be presented as fragility functions, since they are an efficient way to view seismic resiliency. For this purpose, it is useful to be familiar with them and their component parts.

Note that fragility curves are probabilistic, meaning that the value of a point is the likelihood of failure at a given intensity. For this reason, fragility curves should not be treated as definitive indicators of failure.

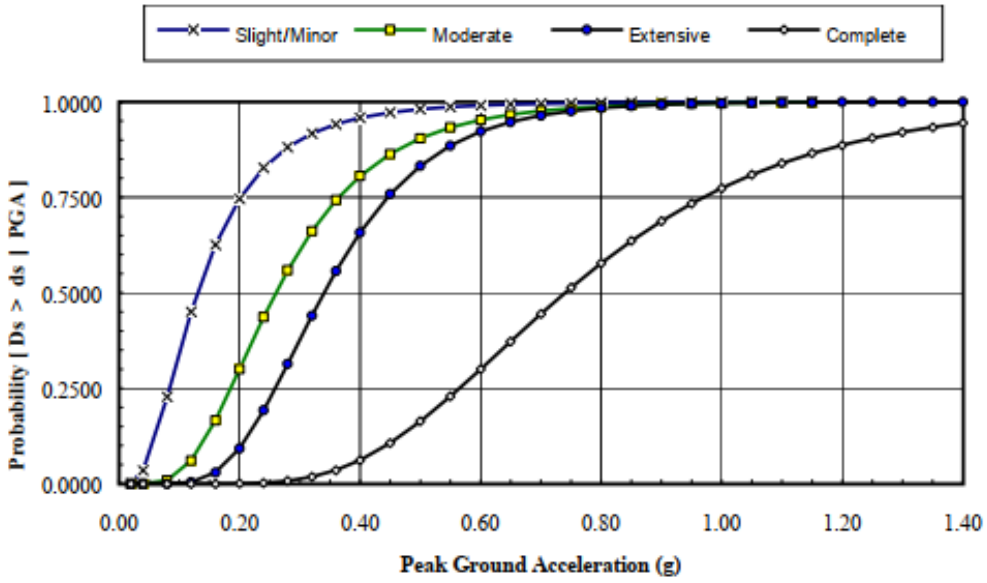


Figure 3.1: Example fragility function for 34.5kV-150kV substation using standard components, from Hazus. [5]

The plotted lines of fragility curves serve as indications of probability of failure. In 3.1, each line represents the probability of that damage state occurring. For example, at 0.4 PGA there is a 10% probability of complete substation failure, a 67% probability of at least extensive substation failure, a 80% probability of at least moderate failure, and a 95% probability of at least slight failure.

The x-axis of the plot shows PGA, representing greater earthquake intensity as it increases. The y-axis is the probability of failure, here represented in performance-based earthquake engineering (PBEE) notation. For the results and comparative plots in this thesis, failure will be viewed as a simple probability on a scale from 0 to 1.

3.2 Restoration Curves

After a disaster, it is important for local government, emergency services, and utility operators to have some idea of when a given structure or network will be back online. To this end, restoration curves are useful as a general indicator of functionality after a given amount of time has elapsed. However, they are entirely dependent on data from manufacturers and construction interests [5], and represent a real-world approximation of how soon the node of interest will be back to full functionality.

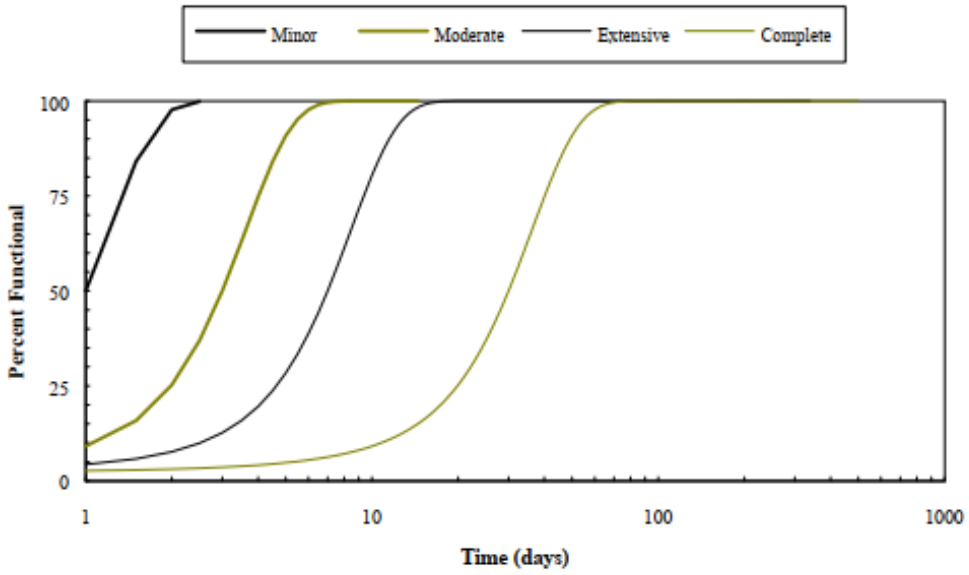


Figure 3.2: Restoration curves for electric substations, for various damage levels [5].

This thesis aims to improve the accuracy of restoration curves by providing more concrete data at the failure level. Current substation models do not provide

the component-level analysis necessary to differentiate between a disconnect switch and a transformer. The code included here not only provides an equivalent list of failures to the Hazus data, but can be made to identify the most likely components to fail and allow local utilities to shorten their restoration time by preparing substitutes and replacements for vulnerable components.

3.2.1 Substation Layouts

The Earthquake Resilience of the Western Power Grid project lays out a number of parameters that govern how substation configuration should be determined. This is taken from general best practices in protection, and governs how configurations are assigned in the code.

Four basic protection configurations exist for this project: Single Bus Single Breaker (SBSB), Ring Bus (RB), Breaker-and-a-Half (BAH), and Dual Bus Dual Breaker (DBDB). The chosen protection scheme is selected along the following guidelines:

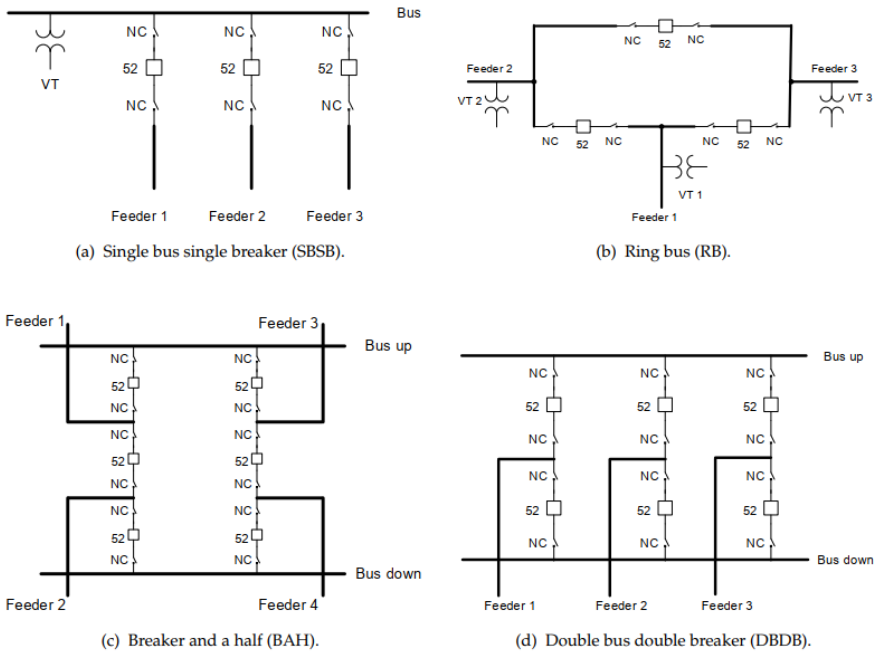


Figure 3.3: Archetypal substation layouts. [6]

- Buses with 2 feeders, no generators: SBSB
- Buses with 3-4 feeders, no generators: RB
- Buses with 5+ feeders, no generators: BAH
- Buses with generators, regardless of feeders: DBDB

A general setup for a line connecting a bus to a feeder is to have, in series, a disconnect switch, a circuit breaker, and another disconnect switch. This can be considered an industry standard for a generic line, and serves well for the generalized substations that are being modeled.

For matters of voltage switching, the default is to assume a single high-side feeder, designated in the code as feeder 1. This feeder also contains a power transformer.

3.2.2 Code Algorithm

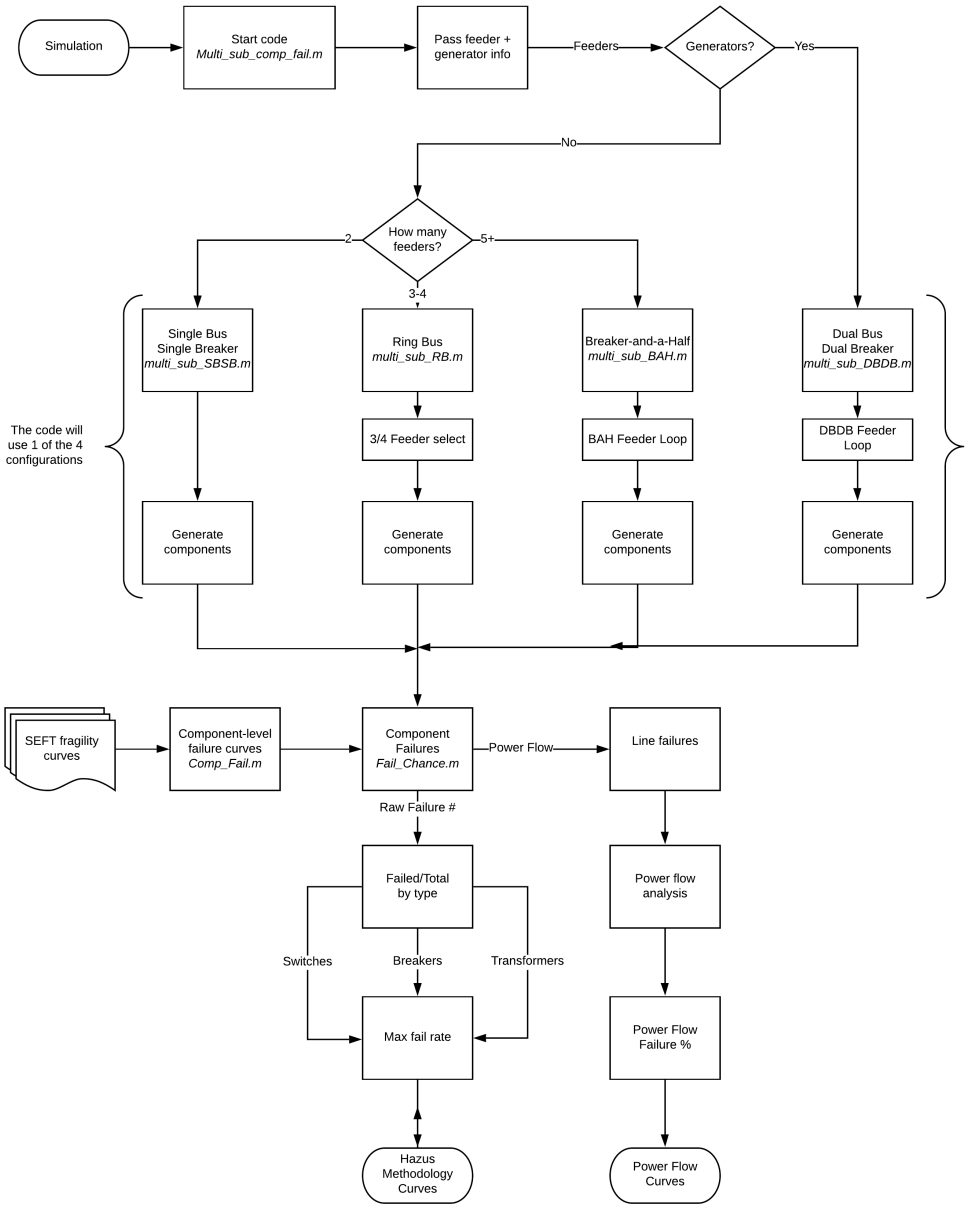


Figure 3.4: Code Algorithm Flowchart.

The major branch of the code for this thesis represents a PGA sweep for a variable substation. This means that, for a custom substation determined by user input, failure probability curves will be generated for the range of 0 to 1.4 PGA.

As seen in Figure 3.4, the code in its current state requires the user to input only the number of feeders and whether or not a generator is present. From this information, a substation configuration is selected. The SEFT data is then accessed, and failure states are generated for each component individually.

Once individual failure states have been generated, two forms of evaluation occur.

- In the power capacity failure analysis, the components are arranged into the structure provided by the substation configuration, and failure is determined based on the number of feeders that are able to connect to each other.

- In the component failure analysis, which is the same methodology used by the Hazus curves, the number of failed components is examined as a fraction of a whole. If it reaches certain thresholds, failure states are reached as stated in the Hazus analysis above.

The power capacity failure analysis is the most relevant for looking at the system during a seismic event, and attempting to predict how it will behave under stress. However, the component failure analysis is more useful for analyzing repair times, costs, and difficulty. Thus, both will be found and inspected.

3.2.3 Response Spectra

SEFT data recognizes three distinct response spectra, A, B, and C [2]. Simply put, these are bedrock and soil qualities that attenuate the earthquake's vibrational frequencies, amplifying some while suppressing others. These spectra vary widely based on region, and do not generally have a significant impact on the fragility of the components this project examines. For this thesis, soil type B is assumed. Appendix A contains a brief investigation on the extent to which different soil types can affect results. The results found there indicate negligible differences at all damage states except complete, where there is a maximum difference of 0.1 in complete failure probability.

Chapter 4: Simulation Results and Analysis

4.1 Results Terminology and Description

In the following sections, various types of outputs and conditions for failure will be discussed. The terms used will be as follows:

- **Power Capacity Failure:** A type of failure directly related to the substation's ability to carry power. Dependent on line failure, not number of components.
- **Component Failure:** A type of failure directly related to the number of components no longer functioning. Described by $\frac{Components\ Failed}{Total\ Components}$. This is the methodology used by Hazus.

In the plotted results, two types of curves will be seen.

- **High-Level:** Curves taken directly from Hazus reporting. Represented with dashed lines.
- **Low-Level:** Curves built from individual component-level failures. Represented by heavy solid lines.

In all cases, colors indicate paired failure states. Blue lines are $P(slight)$, green is $P(moderate)$, magenta is $P(extensive)$, and red is $P(complete)$.

- **Slight:** 5% of components failed or 5% reduction in power capacity. Represented by blue lines.
- **Moderate:** 40% of components failed or 40% reduction in power capacity. Represented by green lines.
- **Extensive:** 70% of components failed or 70% reduction in power capacity. Represented with magenta lines.
- **Complete:** All component failed or zero power capacity. Represented by red lines.

For the purposes of producing mathematically consistent and representative cumulative distribution functions, logistic regression curves have been generated based on the experimentally-generated data points. These points are visible as filled dots on the plots, but the curves labeled as *low-level* are generated through the method seen in Appendix B.1.

Also of note are the guide shapes present on the experimentally-derived logistic curves. These are present purely as identification, especially when multiple lines overlap. They are not indicative of data points, and are there only to assist the reader.

4.2 Results Overview

This code can be run either for an individual substation, or as part of a larger simulation. One branch is focused on examining the failure for any given substation

and comparing against Hazus data, while the other is more focused on providing a mechanism to directly improve ongoing research.

The Hazus branch consists of a standalone MATLAB script which provides fragility information for a theoretical substation. This creates a set of fragility curves that can serve to replace the Hazus curves, allowing for component-level data to be used in place of Hazus' native failure curves.

It is worth noting that all power capacity data is made using the SEFT control house failure curve unless otherwise stated. This is a relatively fragile component whose failure immediately causes *extensive* state damage, and so it tends to dominate all power capacity analysis. This will be examined further below, but is worth noting as a reason for a sharp increase to *extensive* damage just below 0.2 PGA.

Hazus is a comprehensive tool, and one that has numerous applications outside of simply computing substation failure. The comparative failure curves, referred to in this report as *component failure* can be cross-applied to other resources such as restoration curves. Power capacity is important for keeping the grid from collapsing entirely during and after an event. However, in order to bring infrastructure back online after, it is vital that the actual number of damaged components is known. These data also serve as a general test of Hazus' accuracy with regards to this project's component-level approach.

4.3 Results Plots

4.3.1 Single Bus Single Breaker

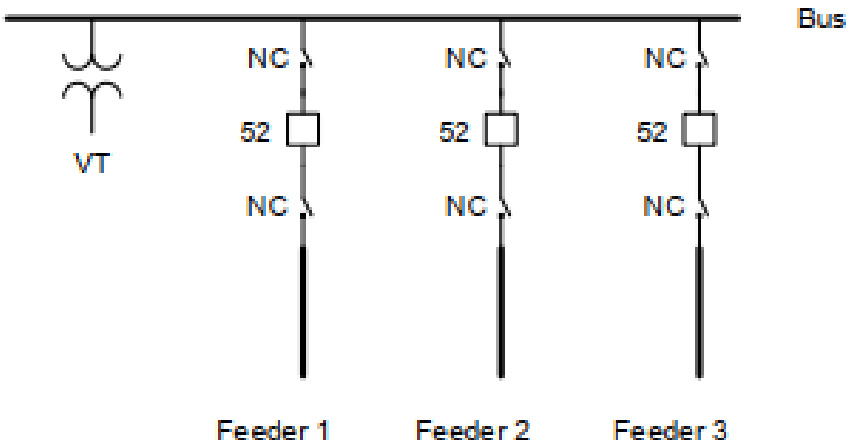
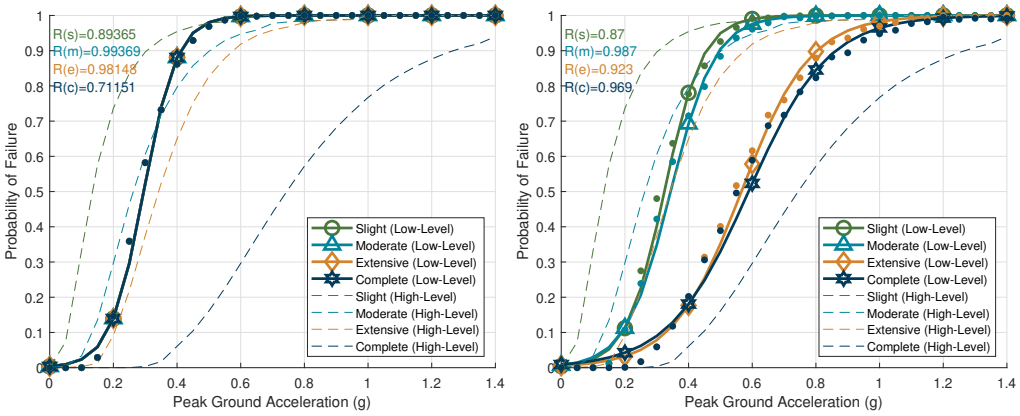


Figure 4.1: Single Bus Single Breaker Substation Archetype



(a) Power Capacity Failure (b) Component Failure

Figure 4.2: Single Bus Single Breaker Fragility Curves (2 feeders)

The only case in which the Single Bus Single Breaker configuration will appear is when there are 2 feeders and no generators. It is the least protected from a power flow standpoint. Because SBSB configurations are designed with only a single path for the entire substation, any component's failure zeroes power capacity and triggers a *complete* failure state.

When examining component failure, the curves approximate Hazus' data with a correlation coefficient of 0.9. The most notable aspect is the tightening of the curves, with the *slight*, *moderate*, and *extensive* failure states appearing at greater PGA values while complete failure occurs at a lower value. This is, however, consistent with the limitations of the singe bus single breaker design, which prioritizes economy over reliability.

4.3.2 Ring Bus

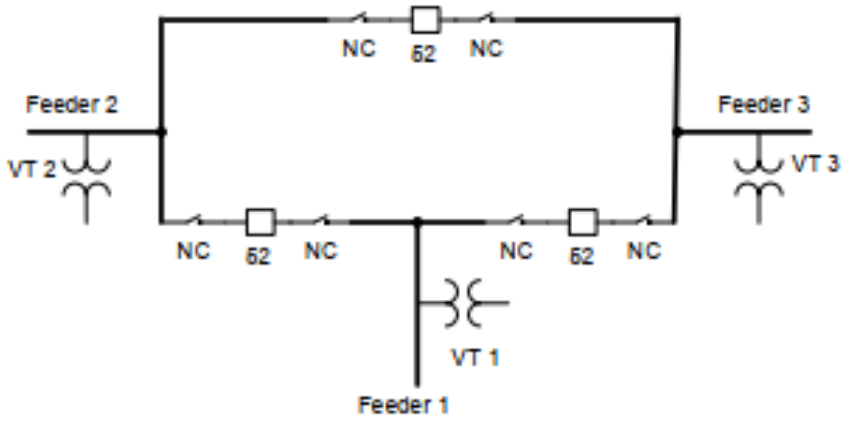
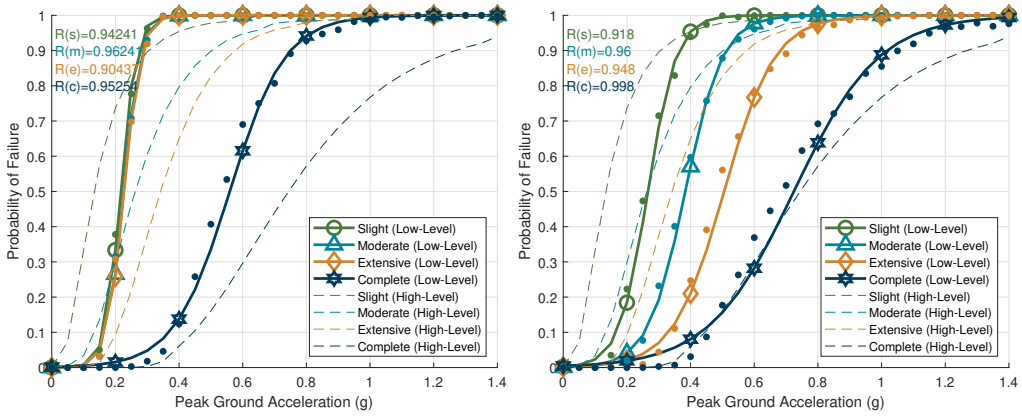


Figure 4.3: Ring Bus Substation Archetype

The Ring Bus case is selected when there are either 3 or 4 feeders, and no attached generators. This configuration is notable for the alternative pathway available should a single breaker or switch fail.

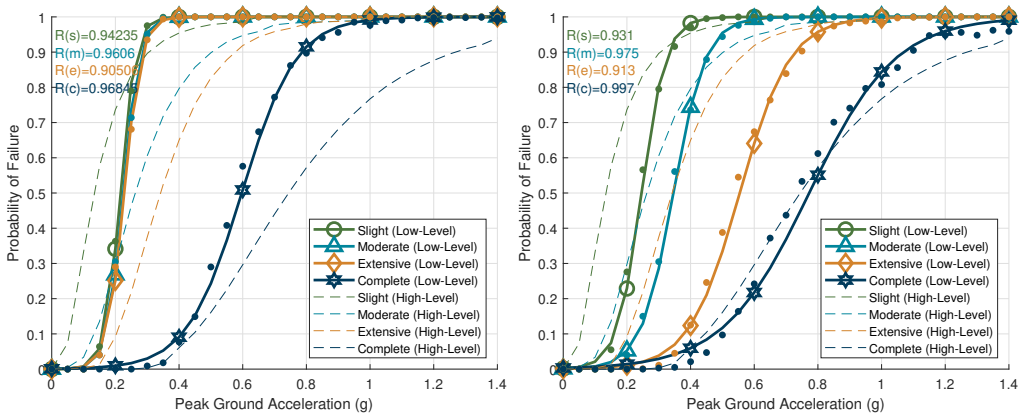


(a) Power Capacity Failure (b) Component Failure

Figure 4.4: Ring Bus Fragility Curves (3 feeders)

The notable difference from the single bus single breaker configuration as seen in Figure 4.2a is that we can see differentiation between the slight and moderate failure cases and the extensive and complete. This is reflective of the ring bus' ability to offer an alternative pathway for power flow in the case of a single line's failure.

Hazus behavior for 3 feeders is fairly similar to Figure 4.2b. It still follows the same general trends, including the compression of failure curves to within a narrower band of PGA values.



(a) Power Capacity Failure (b) Component Failure

Figure 4.5: Ring Bus Fragility Curves (4 feeders)

Differences between 3 and 4 feeder ring bus configurations as seen in Figure 4.4a are minimal, though it should be noted that complete failure occurs at higher PGA at 4 than at 3.

Of note here is the slightly steeper curve as the number of buses increases. With more components available to fail, *slight* and *moderate* states occur at slightly lower PGA. In Figure 4.4b there is $P(moderate) \approx 0.61$ while Figure 4.5b has $P(moderate) \approx 0.75$ at $PGA = 0.4$. However, 3 feeders yields $P(extensive) \approx 0.78$ while 4 has $P(extensive) \approx 0.68$ at $PGA = 0.6$.

4.3.3 Breaker-and-a-Half

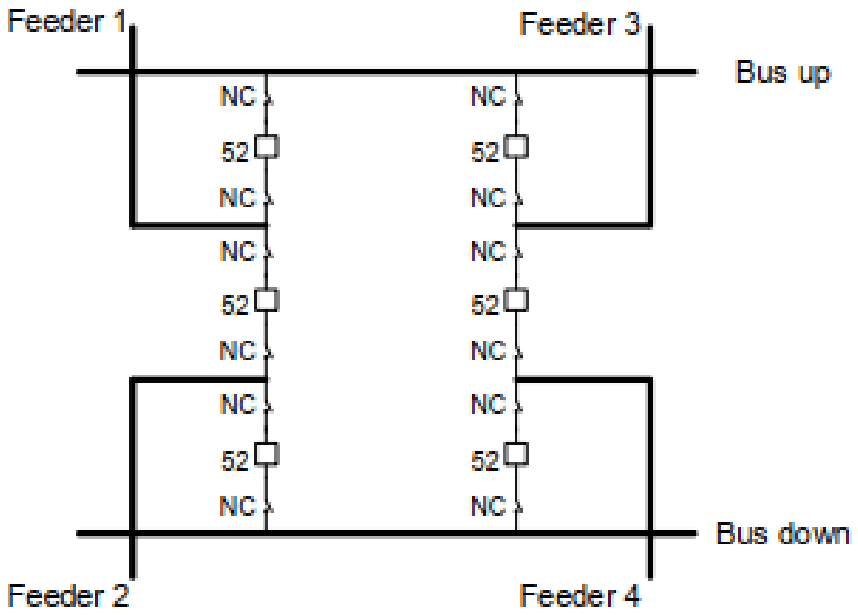
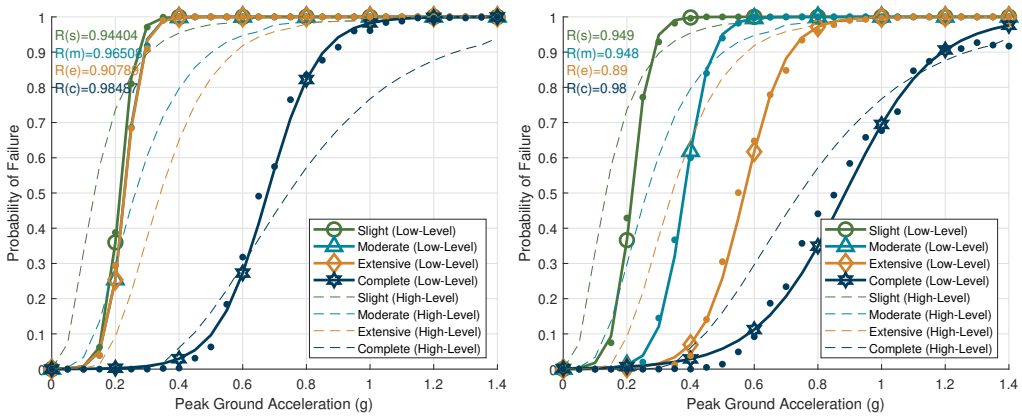


Figure 4.6: Breaker-and-a-Half Substation Archetype

Breaker-and-a-Half configuration offers a middle ground between Dual Bus Dual Breaker and the less redundant configurations. It offers significant advantages from a standard protection standpoint, but the linked feeders and indirect pathing that occurs if a single component fails can offer vulnerabilities in seismic situations.

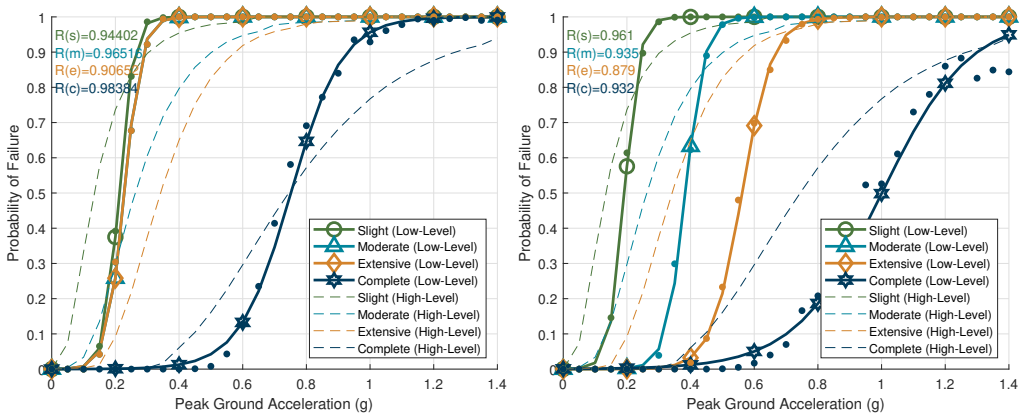


(a) Power Capacity Failure (b) Component Failure

Figure 4.7: Breaker-and-a-Half Fragility Curves (5 feeders)

As with other configurations, *slight* through *extensive* damage states are dominated by control house failure, and relatively unchanged from before. However, $P(complete) \approx 0.33$ occurs at $PGA = 0.6$, as opposed to $P(complete) \approx 0.55$ at $PGA = 0.6$ in Figure 4.5a. This trend of *complete* failure state being reached at higher PGA will continue as the number of feeders increases.

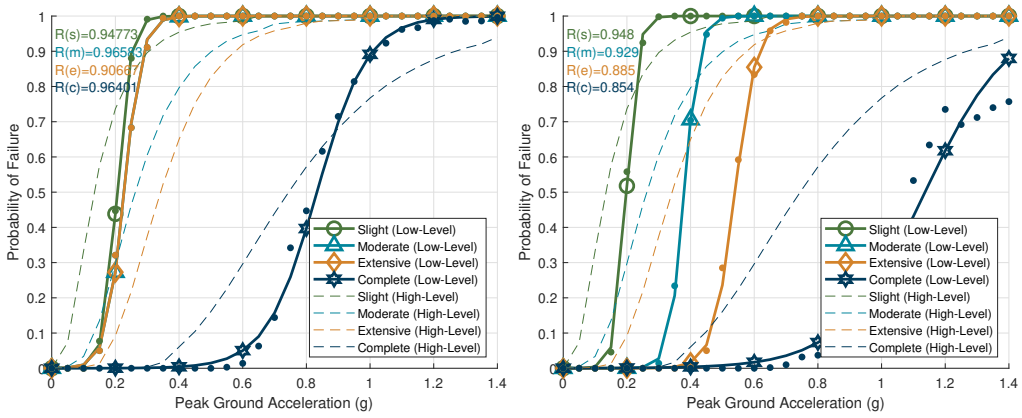
At a low number of feeders, the breaker-and-a-half protection scheme is not dissimilar to the ring bus formation. Similar trends are followed, and all behavior is in line with previous changes from the Hazus fragility curves.



(a) Power Capacity Failure (b) Component Failure

Figure 4.8: Breaker-and-a-Half Fragility Curves (10 feeders)

As stated before, increasing numbers of feeders yields higher PGA values for complete failure curves, due to the additional feeders and resilience. At $PGA = 0.6$, $P(complete) \approx 0.02$, while $P(complete) \approx 0.48$ occurs at $PGA = 0.8$.



(a) Power Capacity Failure (b) Component Failure

Figure 4.9: Breaker-and-a-Half Fragility Curves (20 feeders)

Most notable here is the *complete* failure curve falling entirely to the right of the Hazus data. This can be explained by examining the criteria for a complete

failure. In order for this condition to trigger, all of one type of component must fail. In a breaker-and-a-half configuration with 20 feeders, there are 60 disconnect switches and 30 circuit breakers. If at least 1 of both does not fail, complete failure cannot be reached.

4.3.4 Dual Bus Dual Breaker

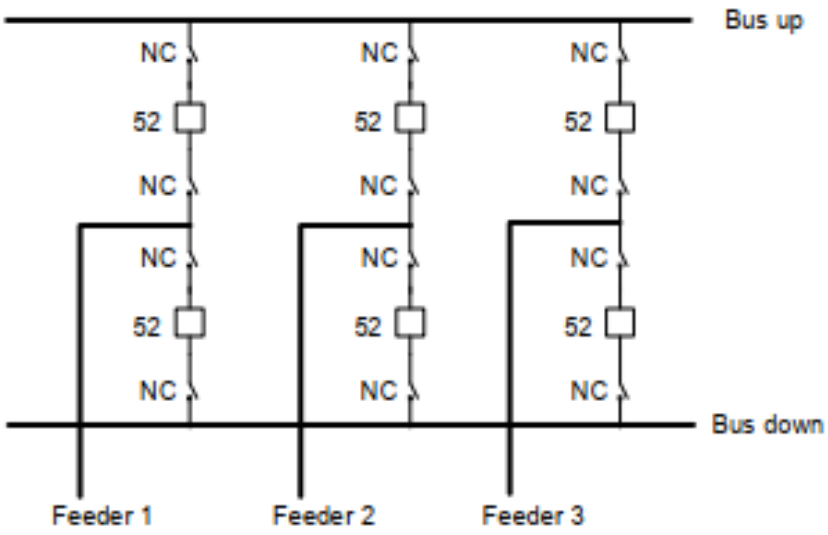
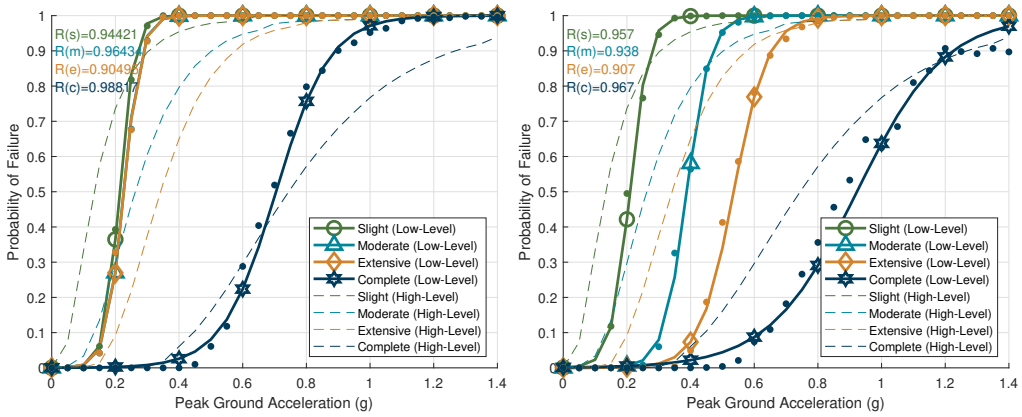


Figure 4.10: Dual Bus Dual Breaker Substation Archetype

The Dual Bus Dual Breaker configuration is selected when a generator is present. This configuration provides the most protection for individual feeders, and the parallel buses are designed to allow for complete redundancy in the case of electrical fault. However, as has been seen before, even redundant complexity increases the number of possible failure points, especially affecting the power capacity failure

results.

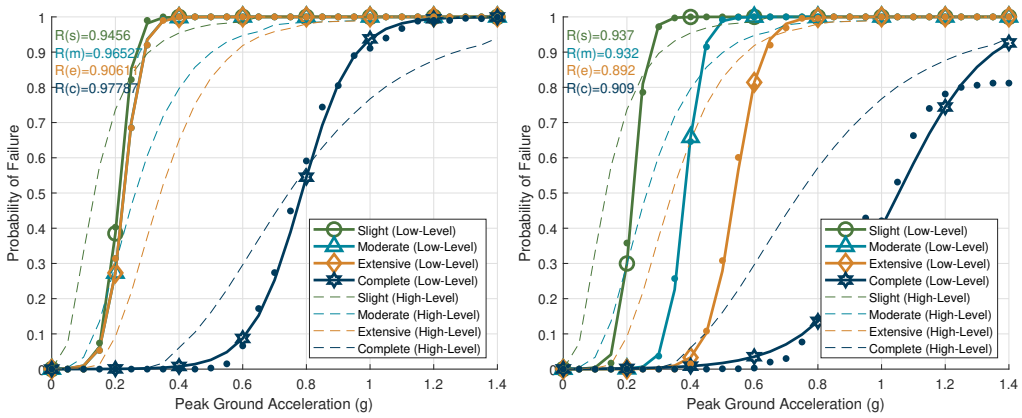


(a) Power Capacity Failure (b) Component Failure

Figure 4.11: Dual Bus Dual Breaker Fragility Curves (5 feeders)

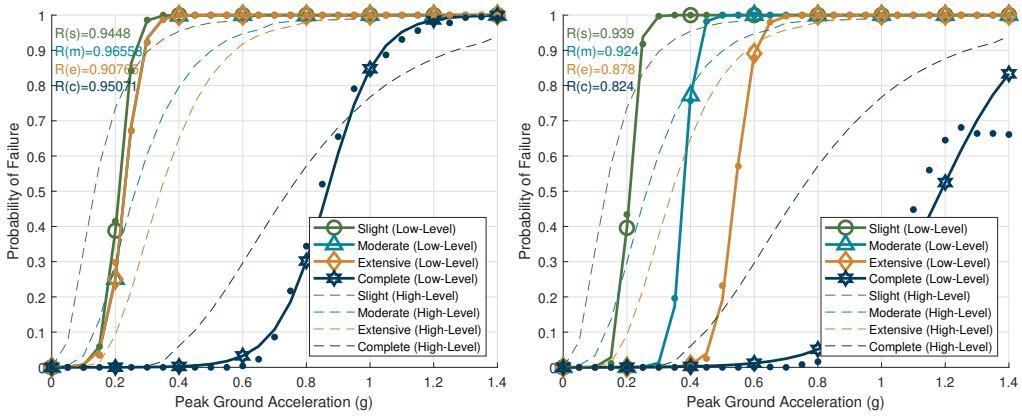
Dual Bus Dual Breaker, as the most robust configuration, should have a higher *complete* failure threshold than an equivalent BAH feeder scenario. Accordingly, $P(complete) \approx 0.25$ occurs at $PGA = 0.6$, while Figure 4.7a has $P(complete) \approx 0.33$ at $PGA = 0.6$.

Component failure methodology sees a lesser shift in reliability than does power capacity when transitioning from BAH to DBDB at low feeder numbers. Here, there are 10 circuit breakers and 20 disconnect switches, as opposed to 8 and 16 for BAH. This difference is not enough to dictate significant raw failure difference.



(a) Power Capacity Failure (b) Component Failure

Figure 4.12: Dual Bus Dual Breaker Fragility Curves (10 feeders)



(a) Power Capacity Failure (b) Component Failure

Figure 4.13: Dual Bus Dual Breaker Fragility Curves (20 feeders)

As with the BAH 20 feeder case, it can be noted that the *complete* failure case is significantly less likely than its Hazus counterpart, though to an even greater extent. This is due to the fact that, as with the BAH configuration, 20 feeders yields 40 circuit breakers and 80 disconnect switches, and none of them are a guaranteed failure.

4.4 Power Capacity Analysis

The first and most notable result of the power capacity analysis is that almost all substations go to $P(extensive) > 0.50$ at $PGA < 0.4$. This is entirely due to the presence of the Control House, which is far and away the most vulnerable component. Consulting SEFT data shows that anchorage failure for the structure as a whole was the primary cause of failure, an issue exacerbated by apparent lapses in best practice found in the outer construction of the control house.

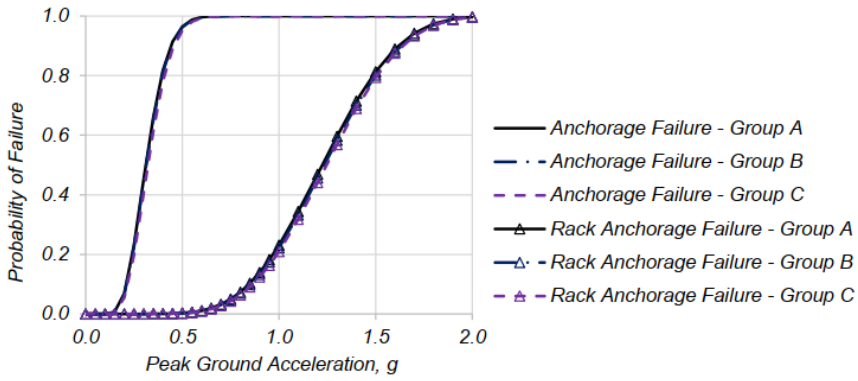


Figure 4.14: 57 kV Control House failure curves [2]

In the interest of determining how the substation would behave assuming control house failure were impossible, a simulation was run with control house failure disabled. The results show a marked difference, as the moderate and extensive failure curves see separation.

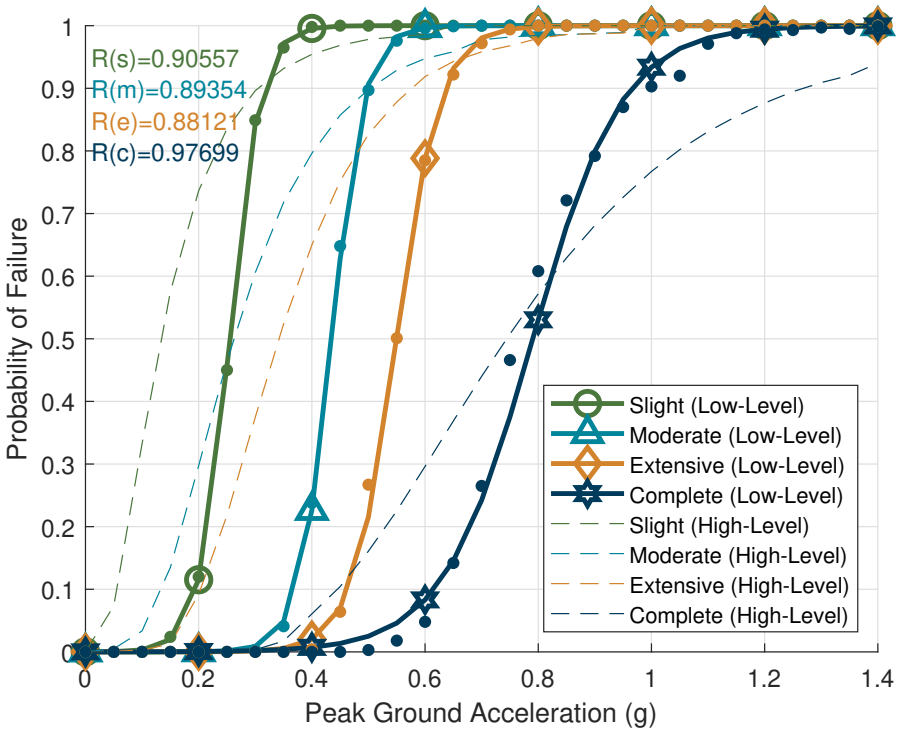


Figure 4.15: Dual Bus Dual Breaker power flow failure curves, no control house (10 feeders)

While it may be tempting to use this information to argue for a reduced emphasis on the control house in this simulation to get a better spread of data, this structure is vital to the functionality of the substation. The choice to have a control house failure move the substation to *extensive* damage rather than *complete* was made only because a control house collapse does not necessarily dictate the total destruction of all equipment within. Without a properly functioning control house, all other components of the substation are unable to properly function.

4.5 Component Failure Analysis

Hazus data does not take into account number of feeders or substation protection configuration. Therefore, various setups will have varying levels of fidelity to the Hazus data.

Generally, curves computed using the Hazus methodology tend to have their *slight*, *moderate*, and *complete* failure states at higher PGA than Hazus' original reports. The edge cases of SBSB and high feeder BAH and DBDB aside, complete failure curves follow their Hazus counterparts fairly closely.

The reasonable conclusion to draw from these curves is that the curves are largely consistent, which indicates a high level of reliability for both sets of results. Moving forward, it would not be unreasonable to consider substituting component-derived Hazus methodology curves when using Hazus resources.

4.6 Implementation with Synthetic Western Power Grid Code

The impetus for this work was creating a model that could be used to approximate substation function for the synthetic model being created of the Western Interconnection. Therefore, it is useful to evaluate the end product and see how well it works with the current synthetic models.

The main substation model used throughout this project is not designed to be inserted into the code as currently written. However, it could easily be modified to do so, and that capability has been written and commented out. Once the exact designs and needs of the synthetic model are known, the extant code could be

slotted in with minimal revision.

Currently, the synthetic model uses two implementations of the substation code, both of which are designed to be lightweight and provide failure information without the Hazus compatibility checks or probabilistic calculations. One such set is a generic line failure, consisting of two disconnect switches on either side of a circuit breaker. The other is a failure for a lone transformer, designed for points where voltage is stepped up or down. When used with the augmented bus-branch model adopted by the project, they allow for each line to be evaluated independently, rather than taking the substation as a unit[6].

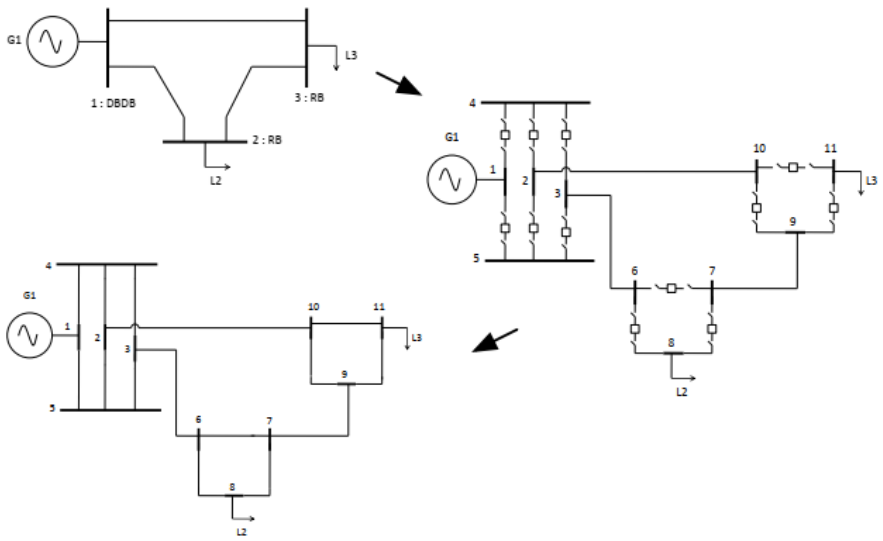


Figure 4.16: Process for the creation of the Augmented Bus-Branch model [6])

The major downside of this method is a marked increase in complexity, as it is no longer possible to view any substation as a node. However, in exchange, it is possible to view each line independently, and gain a higher level of fidelity when

evaluating power capacity failure. Ultimately, it will be up to the researcher or team working on a final implementation of the synthetic model of the Western Interconnection to decide which approach fits their criteria better.

This additional use shows the versatility of the models created for this project. While it is not always feasible to present all of the data generated at any one time, it exists nonetheless, and can be accessed as needed.

Chapter 5: Conclusion

It is premature to use this data to completely rewrite substation failure models, given the limited data we have and the vastness of the project's stated goals. However, the work done here represents a framework by which individual components may be integrated into substation-level fragility analysis, bridging the gap between the narrow focus of real-world breakage numbers and the more synthetic probabilities necessary for large-scale analysis. The code created here is highly robust for the protection schemes outlined, and can be scaled up or down as needed.

From a power capacity standpoint, the most germane take-away is that the control house, for all that it is vital, is also dangerously underprotected. Reinforcing this structure would appear to be the easiest way to shift fragility curves to higher PGAs, and in doing so would represent the most cost-effective option to seismically reinforce substations.

Much of this thesis is proof-of-concept for the integration of component data into substation fragility curves. Local utility data could be substituted for the SEFT data used, and other regional changes could easily be implemented. The results presented here will be leveraged in the larger project analysis. However, as it stands the results described above allow this code to stand on its own as a mathematically reliable alternative to traditional Hazus substation modeling.

Bibliography

[1] PNSN: Cascadia Subduction Zone. https://pnsn.org/outreach/earthquakesources/csz.

[2] SEFT Consulting Group. Power Transmission and Distribution Seismic Vulnerability Assessment, June 2018.

[3] Kevin W. Franke et al. Geotechnical Engineering Reconnaissance of the 20 November 2018 M7.0 Anchorage, Alaska Earthquake (Version 2.0), 2018.

[4] Oregon Seismic Safety Policy Advisory Committee. Oregon Resilience Plan, February 2013.

[5] FEMA Mitigation Division. Earthquake Model Hazus- MH 2.1 Technical Manual, 2012.

[6] Project Description: Earthquake Resilience of the Western Power Grid.

[7] B Wilkinson, B McElroy, and C Dummond. Broken Sheets- On the Numbers and Areas of Tectonic Plates, November 2017.

[8] Chris Goldfinger, C. Hans Nelson, Ann E. Morey, Joel E. Johnson, Jason R. Patton, Eugene B. Karabanov, Julia Gutierrez-Pastor, Andrew T. Eriksson, Eulalia Gracia, Gita Dunhill, and et al. Turbidite Event History-Methods and Implications for Holocene Paleoseismicity of the Cascadia Subduction Zone, Jan 2020.

[9] Oregon Department of Geology and Mineral Industries. Cascadia: News and information from the oregon department of geology and mineral industries, 2012.

[10] L Staisch, M. Walton, and R. Witter. Addressing Cascadia Subduction Zone great earthquake recurrence . *Eos*, July 2019.

[11] Cascadia Region Earthquake Workgroup. Cascadia Subduction Zone Earthquakes: A Magnitude 9.0 Earthquake Scenario, Update 2013, 2013. https://www.dnr.wa.gov/publications/ger$_i$c116$_c$sz$_s$cenario$_u$pdate.pdf.

[12] Liang Chang, Amr Elnashai, and Billie Spencer. Post-earthquake modelling of transportation networks. *Structure and Infrastructure Engineering*, 8:893–911, 10 2012.

[13] M. Reza Manshoori. Evaluation of Seismic Vulnerability and Failure Modes for Pipelines. *Procedia Engineering*, 14:3042–3049, 2011.

[14] M. Nazemi, M. Moeini-Aghtaie, M. Fotuhi-Firuzabad, and P. Dehghanian. Energy Storage Planning for Enhanced Resilience of Power Distribution Networks Against Earthquakes. *IEEE Transactions on Sustainable Energy*, 11(2):795–806, 2020.

[15] Hazus Homepage.

[16] Keith Porter, Ron Hamburger, and Robert Kennedy. Practical Development and Application of Fragility Functions.

APPENDICES

Appendix A: Response Spectra Investigation

In the interest of fully accounting for variance among seismic responses for components based on spectra, the most significant shift based on response spectra is in the 57 kV disconnect switch.

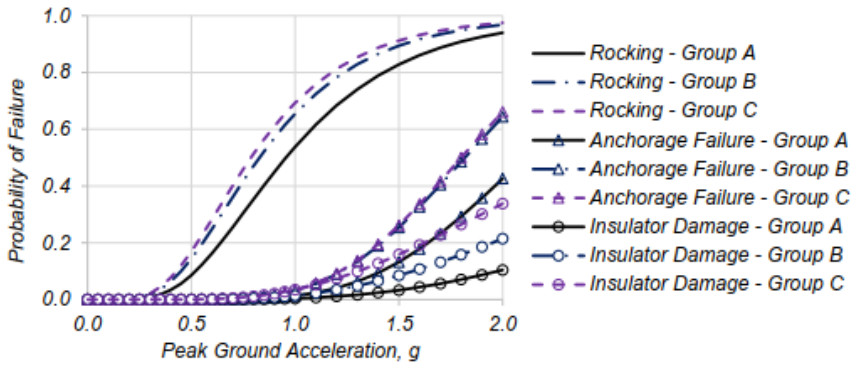


Figure A.1: 57 kV Disconnect Switch, all failure cases, all spectra. [2]

Figure A.1 shows the fragility curves for a 57 kV disconnect switch across spectra, and for all probable causes of failure.

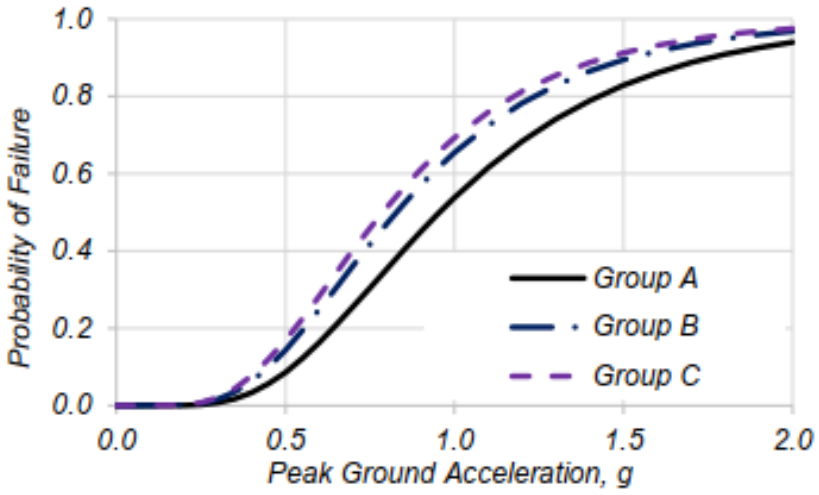


Figure A.2: 57 kV Disconnect Switch, max failure case, all spectra. [2]

From Figure A.2, the variance in failure probabilities is minimal, but should be investigated to ensure it is not a source of error. To this end, substation failure was plotted using all three response spectra, producing a curve that demonstrates the variance in fragility depending on response spectra information.

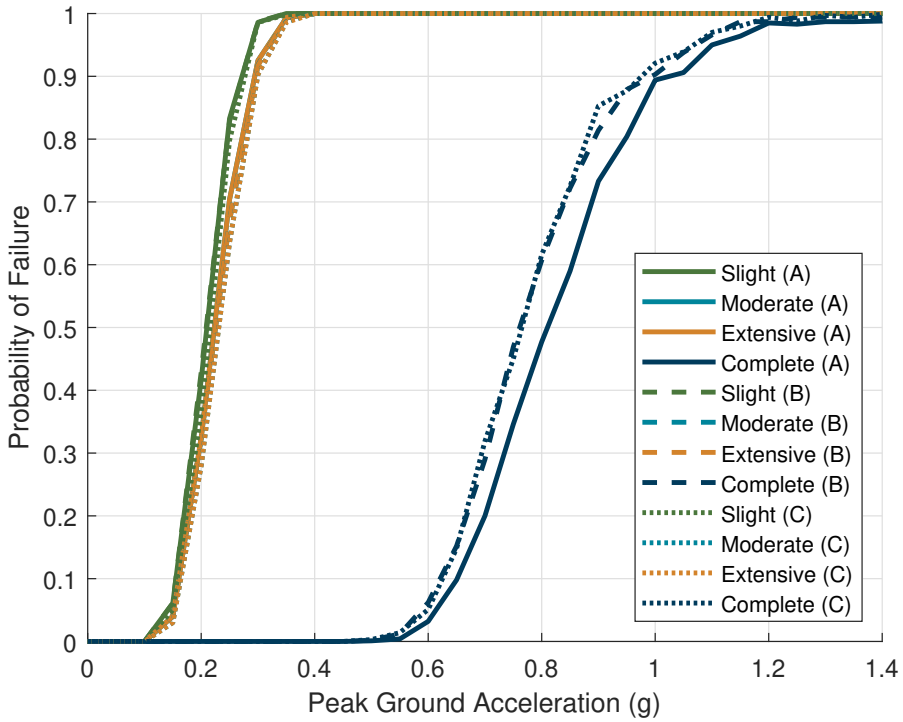


Figure A.3: Dual Bus Dual Breaker Response Spectra Comparison

Fig. A.3 clearly shows that, even at the largest possible variance in response spectra, the overall fragility of the substation is substantially similar, and complete failure is the only curve where any difference beyond the trivial can be detected.

Appendix B: Code

A NOTE ON THE CODE BELOW:

Code with a name containing *Function* below is in function form, meaning that it is intended to be called by another piece of code. Errors shown in the publishing are an artifact of that fact, and not a problem with the code's accuracy.

B.1 Main Code

NOTE: This code relies on input from the user. The lack of that input in the publishing process shows as errors below. It is not a logical error in the code.

Contents


```
%function Multi_sub_comp_fail(feed, gen_in)
```

Base code - William Stark

```
% Stand-alone form, for ease of runnung


%This is the base from which all functions will be executed. It will loop
%component-level failures in the selected configurations to return
%substation-level failure states.

%Current has function functionality disabled, using inputs.
```

Initialization

```
clear all
clc
```

- Test Input - Selecting from 4 basic configurations. Assumes 115 kV on high-side and 57 kV on low side, and response spectra B

Note that, per the project guidelines, we will be using a combination of configurations based on number of feeders.

```
%If there is a generator, we
% will use DBDB regardless of feeder #

prompt = 'Enter number of feeders (2 or more)\n';
feed = input(prompt);

if feed < 2
    error('Please choose a workable number of feeders')
```

```
end

%If there is a generator, we
% will use DBDB

prompt = 'Is there an attached generator? (Y/N)\n';
gen_in = input(prompt, 's');
gen = lower(gen_in);

if ~ismember(gen, ['y' 'n'])
    error('Please choose either Y or N')
end


% For now, we assume Response Spectra 'A'
%
% prompt = 'What is the Response spectra? (A, B, or C) ';
% Spec = lower(input(prompt, 's'));
%
% if ~ismember(Spec, ['a' 'b' 'c'])
%     error('Invalid Response Spectra')
% end

Spec = 'b';
```

```
Error using input
Cannot call INPUT from EVALC.

Error in Multi_sub_comp_fail (line 27)
feed = input(prompt);
```

Matrix Generation

```
PGA_mat = [0:0.05:1.4]; %Matrix Containing all measured PGA values (Length 29)

States = zeros(5,29); % matrix containing possible failure states: 0 (no failure) 0.333 (1 li
ne failed)* 0.667 (2 lines failed) 1 (All lines failed)
S_raw = zeros(5,29);
```

- Select Substation, Loop through all PGA values 0-2 ten times and build a matrix where each term has PGA and operational states

```
S_count = 1;

run = zeros(1,1000);
raw = zeros(1,1000);



 for n = 1 : length(PGA_mat)
```

```
    PGA = PGA_mat(n);

    for k = 1:1000

        if gen == 'y' % Generator present, means Dual Bus Dual Breaker
            [Fail_state, F_raw] = model_sub_DBDB(feed, PGA, Spec);
            type = 'Dual Bus, Dual Breaker';
            savename = 'DBDB';
        elseif feed == 2 % 2-feeder substation, SBSB
            [Fail_state, F_raw] = model_sub_SBSB(PGA, Spec);
            type = 'Single Bus, Single Breaker';
            savename = 'SBSB';
        elseif feed < 5 % 3 or 4-feeder, Ring Bus
            [Fail_state, F_raw] = model_sub_RING(feed, PGA, Spec);
            type = 'Ring Bus';
            savename = 'RB';
        else % 5 or more feeders, Breaker-and-a-half
            [Fail_state, F_raw] = model_sub_BAH(feed, PGA, Spec);
            type = 'Breaker-and-a-Half';
            savename = 'BAH';
        end
            run(k) = Fail_state;
            raw(k) = F_raw;

            if k == 1000

                States(1,S_count) = sum(run==0)/1000; % No damage, does not need a curve
                S_raw(1,S_count) = sum(raw==0)/1000; % No damage, does not need a curve

                States(2,S_count) = (sum(run==0.05) + sum(run==0.4) + sum(run==0.7)+ sum(run==1))/1000; % Slight damage or greater
                S_raw(2,S_count) = (sum(raw==0.05) + sum(raw==0.4) + sum(raw==0.7)+ sum(raw==1))/1000;

                States(3,S_count) = (sum(run==0.4) + sum(run==0.7)+ sum(run==1))/1000; % Moderate damage or greater
                S_raw(3,S_count) = (sum(raw==0.4) + sum(raw==0.7)+ sum(raw==1))/1000;

                States(4,S_count) = (sum(run==0.7)+ sum(run==1))/1000; % Severe damage or greater
                S_raw(4,S_count) = (sum(raw==0.7)+ sum(raw==1))/1000;

                States(5,S_count) = sum(run==1)/1000; % Complete Damage, totally nonfunctional
                S_raw(5,S_count) = sum(raw==1)/1000;

                S_count = S_count + 1;
            end

    end
 end
```

Compare with Hazus

```
HSS_LV_S_raw = [0 0.0001
0.019761551     0.001
```

```
0.042095966     0.00977537
0.051094988     0.082260763
0.060946549     0.12995576
0.07079811      0.177650757
0.079418225     0.213328981
0.088038341     0.266724896
0.096658457     0.313944001
0.104047128     0.354835499
0.112667244     0.403517533
0.121287359     0.448948614
0.129907475     0.489665814
0.139759036     0.532484383
0.150842042     0.578704701
0.161925048     0.62510924
0.17670239      0.672085372
0.191479731     0.714856939
0.210874992     0.764086347
0.228423085     0.800004476
0.260440658     0.852494432
0.296460427     0.89330779
0.320781468     0.913361662
0.347380683     0.930077825
0.374965054     0.945578779
0.403042002     0.955294249
0.427917194     0.962274004
0.457831325     0.970845213
0.489156627     0.978179895
0.518072289     0.977903982
0.544578313     0.981467856
0.573493976     0.985008737
0.595921492     0.983799114
0.623716967     1
1.4         1
];


HSS_LV_M_raw = [0 0.0001
0.018461692     0.003281419
0.043952606     0.004674498
0.071536977     0.006123456
0.099121347     0.032210694
0.124981695     0.070535008
0.145916262     0.126219123
0.162804652     0.162422199
0.175470945     0.213712829
0.187785396     0.255727668
0.201331292     0.301946159
0.214877188     0.346365789
0.227191639     0.388770742
0.240737536     0.436809766
0.255514877     0.480177341
0.271523664     0.528271374
0.292458231     0.587106259
0.313392798     0.63848828
0.33309592      0.683411109
0.351743517     0.721129991
0.3784131       0.767689225
```

```
0.404519736     0.802195753
0.430380084     0.83741393
0.453012048     0.860204175
0.483332224     0.886382247
0.515662651     0.909224685
0.537515809     0.925408631
0.564607602     0.932715974
0.597590361     0.946610871
0.620902236     0.954487945
0.644241053     0.956876329
0.674698795     0.964959073
0.700066565     0.977527843
0.727158357     0.97846332
0.75425015      0.98250789
0.781341942     0.984542779
0.808433735     0.984443862
0.835525528     0.985337963
0.863109898     0.986079593
0.887793531     0.986364215
0.917786061     0.989276964
0.942661253     0.985060439
0.969753045     1
1.4     1
];

HSS_LV_E_raw = [0 0.0001
0.012920189     0.0001
0.094195567     0.00204157
0.121697841     0.00580737
0.147393996     0.008819632
0.175470945     0.049937069
0.197636957     0.086497446
0.217340079     0.133061231
0.23458031      0.175716413
0.250589097     0.219544399
0.265366438     0.265520477
0.281375225     0.314888352
0.297384011     0.366644682
0.313392798     0.415646327
0.330633029     0.468258712
0.345410371     0.510528703
0.360187712     0.552380715
0.377427944     0.597441602
0.394668175     0.637403137
0.414371297     0.683363174
0.437631927     0.727896001
0.457608704     0.766890196
0.483332224     0.803571622
0.510424017     0.841171931
0.537515809     0.867068812
0.567070492     0.89350372
0.593177128     0.915722492
0.621254077     0.927801446
0.64834587      0.942288469
0.675437662     0.950691284
0.720481928     0.964522211
0.751807229     0.968040099
```

```
0.785542169     0.975351789
0.810896625     0.983054961
0.837988418     0.985929188
0.865819077     0.986131765
0.892172003     0.986245595
0.921726686     0.988771225
0.948079611     0.988168663
0.975910271     0.989850128
1.003002064     1
1.4         1
];

HSS_LV_C_raw = [0 0.0001
0.013659056     0.0001
0.286301005     0.0001
0.313392798     0.006759261
0.34048459      0.016076285
0.368162785     0.01960355
0.394175597     0.055775292
0.421759968     0.079078532
0.448851761     0.103113504
0.475943553     0.133177514
0.503281635     0.164746435
0.530127138     0.199984765
0.558204087     0.23571081
0.584310724     0.27364857
0.613865406     0.316160915
0.640957199     0.35574135
0.668048992     0.394671595
0.694648206     0.432684451
0.722232577     0.470758837
0.749570658     0.507062326
0.776416162     0.542235638
0.803507954     0.576378116
0.830599747     0.607683399
0.858676696     0.638855155
0.884783332     0.665328875
0.911875125     0.693442314
0.937981761     0.715731171
0.96605871      0.73973901
0.992904214     0.762410932
1.020242295     0.780893291
1.047334088     0.800534551
1.07442588      0.815663883
1.102885945     0.832514
1.128609465     0.845114735
1.157616839     0.858866299
1.182793051     0.868891197
1.212347733     0.881908537
1.238700659     0.891339005
1.266531319     0.89965596
1.292802148     0.90699084
1.320714904     0.915098162
1.347533042     0.919905869
1.374898489     0.92770317
1.396243538     0.936346027
1.4         0.94
```

```
];
```

Modify Hazus Values to match SEFT layout

```
xPGA = [0:0.05:1.4];



    HSS_LV_S = interp1(HSS_LV_S_raw(:,1), HSS_LV_S_raw(:,2), xPGA);

    HSS_LV_M = interp1(HSS_LV_M_raw(:,1), HSS_LV_M_raw(:,2), xPGA);

    HSS_LV_E = interp1(HSS_LV_E_raw(:,1), HSS_LV_E_raw(:,2), xPGA);

    HSS_LV_C = interp1(HSS_LV_C_raw(:,1), HSS_LV_C_raw(:,2), xPGA);
```

Prepare to PLot

```
% Create power flow arrays

Flow_slight = States(2,:);
Flow_moderate = States(3,:);
Flow_extensive = States(4,:);
Flow_complete = States(5,:);

% Create Hazus Method Arrays

Haz_slight = S_raw(2,:);
Haz_moderate = S_raw(3,:);
Haz_extensive = S_raw(4,:);
Haz_complete = S_raw(5,:);

% Create Logistic curves for experimental component data

% Sourced from http://matlabdatamining.blogspot.com/2009/03/logistic-regression.html

% Additional information from
% https://www.mathworks.com/help/stats/glmfit.html and https://www.mathworks.com/help/stats/g
lmval.html

N = ones(29,1);

%----------------------------------------Power Capacity Failure

FS_b = glmfit(xPGA, [Flow_slight.' , N] , 'binomial','link','logit');
Flow_slight_curve = glmval(FS_b,xPGA,'logit');

FM_b = glmfit(xPGA, [Flow_moderate.' , N] , 'binomial','link','logit');
Flow_moderate_curve = glmval(FM_b,xPGA,'logit');

FE_b = glmfit(xPGA, [Flow_extensive.' , N] , 'binomial','link','logit');
Flow_extensive_curve = glmval(FE_b,xPGA,'logit');

FC_b = glmfit(xPGA, [Flow_complete.' , N] , 'binomial','link','logit');
```

```
Flow_complete_curve = glmval(FC_b,xPGA,'logit');
%----------------------------------------Component Failure

HS_b = glmfit(xPGA, [Haz_slight.' , N] , 'binomial','link','logit');
Haz_slight_curve = glmval(HS_b,xPGA,'logit');

HM_b = glmfit(xPGA, [Haz_moderate.' , N] , 'binomial','link','logit');
Haz_moderate_curve = glmval(HM_b,xPGA,'logit');

HE_b = glmfit(xPGA, [Haz_extensive.' , N] , 'binomial','link','logit');
Haz_extensive_curve = glmval(HE_b,xPGA,'logit');

HC_b = glmfit(xPGA, [Haz_complete.' , N] , 'binomial','link','logit');
Haz_complete_curve = glmval(HC_b,xPGA,'logit');
```

- Plot results - Power Flow

```
figure('Name', [type , ' Seismic Failure States'])
grid on
hold on
xlim([0 1.4])

plot(PGA_mat, Flow_slight_curve,'LineWidth',2, 'Color', 1/255*[74,119,60],'HandleVisibility',
'off')
plot(PGA_mat, Flow_moderate_curve,'LineWidth',2, 'Color', 1/255*[0,133,155],'HandleVisibility
','off')
plot(PGA_mat, Flow_extensive_curve,'LineWidth',2, 'Color', 1/255*[211,131,43], 'HandleVisibil
ity','off')
plot(PGA_mat, Flow_complete_curve,'LineWidth',2, 'Color', 1/255*[0,59,92], 'HandleVisibility'
,'off')

plot(PGA_mat(1:4:end), Flow_slight_curve(1:4:end), 'o','MarkerSize',10, 'Color', 1/255*[74,11
9,60]) % Gives markers for slight
plot(PGA_mat(1:4:end), Flow_moderate_curve(1:4:end), '^','MarkerSize',10, 'Color', 1/255*[0,1
33,155]) % Gives markers for moderate
plot(PGA_mat(1:4:end), Flow_extensive_curve(1:4:end), 'd','MarkerSize',10, 'Color', 1/255*[21
1,131,43]) % Gives markers for extensive
plot(PGA_mat(1:4:end), Flow_complete_curve(1:4:end), 'h','MarkerSize',10, 'Color', 1/255*[0,5
9,92]) % Gives markers for complete

plot(xPGA ,HSS_LV_S, 'Color', 1/255*[74,119,60], 'Linestyle', '--')
plot(xPGA ,HSS_LV_M, 'Color', 1/255*[0,133,155], 'Linestyle', '--')
plot(xPGA,HSS_LV_E, 'Color', 1/255*[211,131,43], 'Linestyle', '--')
plot(xPGA,HSS_LV_C, 'Color', 1/255*[0,59,92], 'Linestyle', '--')

xlabel('Peak Ground Acceleration (g)')
ylabel('Probability of Failure')

Corr_Sf = mean((HSS_LV_S-mean(HSS_LV_S)).*(Flow_slight-mean(Flow_slight)))/(std(HSS_LV_S)*std
(Flow_slight));
Corr_Mf = mean((HSS_LV_M-mean(HSS_LV_M)).*(Flow_moderate-mean(Flow_moderate)))/(std(HSS_LV_M)
*std(Flow_moderate));
Corr_Ef = mean((HSS_LV_E-mean(HSS_LV_E)).*(Flow_extensive-mean(Flow_extensive)))/(std(HSS_LV_
E)*std(Flow_extensive));
Corr_Cf = mean((HSS_LV_C-mean(HSS_LV_C)).*(Flow_complete-mean(Flow_complete)))/(std(HSS_LV_C)
*std(Flow_complete));
```

```
text(0.01, .95 , ['R(s)=' , num2str(Corr_Sf)], 'Color',1/255*[74,119,60])
text(0.01, .9 , ['R(m)=' , num2str(Corr_Mf)], 'Color',1/255*[0,133,155])
text(0.01, .85 , ['R(e)=' , num2str(Corr_Ef)], 'Color',1/255*[211,131,43])
text(0.01, .8 , ['R(c)=' , num2str(Corr_Cf)], 'Color',1/255*[0,59,92])

scatter(PGA_mat, Flow_slight, 25, 1/255*[74,119,60], 'filled' ,'HandleVisibility','off')
scatter(PGA_mat, Flow_moderate, 25,  1/255*[0,133,155], 'filled' ,'HandleVisibility','off')
scatter(PGA_mat, Flow_extensive, 25, 1/255*[211,131,43], 'filled' , 'HandleVisibility','off')
scatter(PGA_mat, Flow_complete, 25, 1/255*[0,59,92], 'filled' , 'HandleVisibility','off')

legend('Slight (Low-Level)','Moderate (Low-Level)', 'Extensive (Low-Level)', 'Complete (Low-L
evel)', ...
       'Slight (High-Level)','Moderate (High-Level)','Extensive (High-Level)','Complete (High
-Level)', 'Location', 'southeast')

% Saving Plot

feedstr = num2str(feed);

filename = [savename, '_Flow', feedstr];
saveas(gcf, [pwd, '/Plots/', filename], 'epsc')
```

Plot Results - Raw Failure

```
figure('Name', [type , ' Seismic Failure States (Hazus Methodology)'])
grid on
hold on
xlim([0 1.4])

plot(PGA_mat, Haz_slight_curve,'LineWidth',2, 'Color', 1/255*[74,119,60],'HandleVisibility','
off')
plot(PGA_mat, Haz_moderate_curve,'LineWidth',2, 'Color', 1/255*[0,133,155],'HandleVisibility'
,'off')
plot(PGA_mat, Haz_extensive_curve,'LineWidth',2, 'Color', 1/255*[211,131,43],'HandleVisibilit
y','off')
plot(PGA_mat, Haz_complete_curve,'LineWidth',2, 'Color', 1/255*[0,59,92],'HandleVisibility','
off')

plot(PGA_mat(1:4:end), Haz_slight_curve(1:4:end), 'o','MarkerSize',10, 'Color', 1/255*[74,119
,60]) % Gives markers for slight
plot(PGA_mat(1:4:end), Haz_moderate_curve(1:4:end), '^','MarkerSize',10, 'Color', 1/255*[0,13
3,155]) % Gives markers for moderate
plot(PGA_mat(1:4:end), Haz_extensive_curve(1:4:end), 'd','MarkerSize',10, 'Color', 1/255*[211
,131,43]) % Gives markers for extensive
plot(PGA_mat(1:4:end), Haz_complete_curve(1:4:end), 'h','MarkerSize',10, 'Color', 1/255*[0,59
,92]) % Gives markers for complete

plot(xPGA,HSS_LV_S, 'Color', 1/255*[74,119,60], 'Linestyle', '--')
plot(xPGA,HSS_LV_M, 'Color', 1/255*[0,133,155], 'Linestyle', '--')
plot(xPGA,HSS_LV_E, 'Color', 1/255*[211,131,43], 'Linestyle', '--')
plot(xPGA,HSS_LV_C, 'Color', 1/255*[0,59,92], 'Linestyle', '--')

xlabel('Peak Ground Acceleration (g)')
ylabel('Probability of Failure')
```

```
Corr_Sh = round(mean((HSS_LV_S-mean(HSS_LV_S)).*(Haz_slight-mean(Haz_slight)))/(std(HSS_LV_S)
*std(Haz_slight)),3);
Corr_Mh = round(mean((HSS_LV_M-mean(HSS_LV_M)).*(Haz_moderate-mean(Haz_moderate)))/(std(HSS_L
V_M)*std(Haz_moderate)),3);
Corr_Eh = round(mean((HSS_LV_E-mean(HSS_LV_E)).*(Haz_extensive-mean(Haz_extensive)))/(std(HSS
_LV_E)*std(Haz_extensive)),3);
Corr_Ch = round(mean((HSS_LV_C-mean(HSS_LV_C)).*(Haz_complete-mean(Haz_complete)))/(std(HSS_L
V_C)*std(Haz_complete)),3);

text(0.01, .95 , ['R(s)=' , num2str(Corr_Sh)], 'Color',1/255*[74,119,60])
text(0.01, .9 , ['R(m)=' , num2str(Corr_Mh)], 'Color',1/255*[0,133,155])
text(0.01, .85 , ['R(e)=' , num2str(Corr_Eh)], 'Color',1/255*[211,131,43])
text(0.01, .8 , ['R(c)=' , num2str(Corr_Ch)], 'Color',1/255*[0,59,92])

scatter(PGA_mat, Haz_slight, 25, 1/255*[74,119,60], 'filled' ,'HandleVisibility','off')
scatter(PGA_mat, Haz_moderate, 25,  1/255*[0,133,155], 'filled' ,'HandleVisibility','off')
scatter(PGA_mat, Haz_extensive, 25, 1/255*[211,131,43], 'filled' , 'HandleVisibility','off')
scatter(PGA_mat, Haz_complete, 25, 1/255*[0,59,92], 'filled' , 'HandleVisibility','off')

legend('Slight (Low-Level)','Moderate (Low-Level)', 'Extensive (Low-Level)', 'Complete (Low-L
evel)', ...
       'Slight (High-Level)','Moderate (High-Level)','Extensive (High-Level)','Complete (High
-Level)', 'Location', 'southeast')




% Saving Plot


filename2 = [savename, '_Hazus', feedstr];
saveas(gcf, [pwd, '/Plots/', filename2], 'epsc')

% end
```

Published with MATLAB® R2018b

B.2 SBSB Function

Contents


```
function [Fail, Fraw] = model_sub_SBSB(PGA, Spec)
```

```
%This function tests component-level failures on a model substation

%From Notes:

% High-side to low-side feeder:
%
% High voltage SD -> High voltage CB -> High voltage SD -> transformer -> Low voltage bus ->
Low voltage SD -> Low voltage CB -> Low voltage SD
%
% Any additional feeders branch off after transformer.
%
% Consider adding secondary systems (PTs, CTs, as data allows)
%
% So for the Hazus comparison, do a single high-side with 10 branch feeders.

% Will not have relaying PTs and CTs, but will add control house
% rated at 115 kV.

%We will be treating each component as a flat pass/fail, then getting
%values to link viability for each of 10 feeders.
```

Initialization

```
Fail = 0; % Saves a headache later
```

Get Component Data

```
    [Fail_SD_57, Fail_CS_57, Fail_IT_57, Fail_GCB_57, Fail_OCB_57, Fail_LTCPT_57, Fail_CH_57,
 ...
    Fail_SD_115, Fail_CS_115, Fail_IT_115, Fail_GCB_115, Fail_OCB_115, Fail_LTCPT_115, Fail_N
LTCPT_115, Fail_CH_115, ...
    Fail_SD_230, Fail_IT_230, Fail_GCB_230, Fail_OCB_230, Fail_LTCPT_230, Fail_LTCPT2_230, Fa
il_CH_230] = Comp_Fail(PGA, Spec);
%Feeds PGA, and Ground Response Spectra, returns failure of component type
```

```
Not enough input arguments.
```

```
Error in model_sub_SBSB (line 36)
    Fail_SD_230, Fail_IT_230, Fail_GCB_230, Fail_OCB_230, Fail_LTCPT_230, Fail_LTCPT2_230, Fa
il_CH_230] = Comp_Fail(PGA, Spec);
```

Ring Bus Failure State - Components

```
%calculate each component's failure

CH_115_1 = Fail_Chance(Fail_CH_115); % Failure chance for the control house (automatic bump t
o severe)

% Branch split

% HV Branch A

SD_115_1a = Fail_Chance(Fail_SD_115);

CB_115_a = Fail_Chance(Fail_GCB_115);

SD_115_2a = Fail_Chance(Fail_SD_115);

LTCPT_115_a = Fail_Chance(Fail_LTCPT_115);

% Low voltage branch

SD_57_1a = Fail_Chance(Fail_SD_57);

CB_57_a = Fail_Chance(Fail_GCB_57);

SD_57_2a = Fail_Chance(Fail_SD_57);
```

Branch and Feeder Failure State

```
% 1 means failure!

% High Side - 1 of 2 branches must remain intact

%High-Side Disconnect

if CH_115_1 == 0
    Fail_CH = 0;
else
    Fail_CH = 1;
end

% Branch A
if SD_115_1a == 0 && SD_115_2a == 0 && CB_115_a ==0 && LTCPT_115_a == 0
    Fail_Ha = 0;
else
    Fail_Ha = 1;
end

% % Branch B
% if SD_115_1b == 0 && SD_115_2b == 0 && CB_115_b ==0 && IT_115_b == 0
```

```
%     Fail_Hb = 0;
% else
%     Fail_Hb = 1;
% end

% Feeders on low-voltage side

% Feeder A
if SD_57_1a == 0 && SD_57_2a == 0 && CB_57_a ==0
    Fail_La = 0;
else
    Fail_La = 1;
end
```

Create substation failure percentage from branch failures

```
% To compare with Hazus - Slight = 0.1, Moderate = 0.4, Severe = 0.7,
% Complete = 1.0

% Total Failure condition - High-side SD
if Fail_CH == 1
    Fail = 0.7;
end
     if Fail_Ha == 1  % Failure of the branch -> Total
         Fail = 1;
     else
         Fail = 0;
     end

    %                                Low-voltage side: each feeder's failure
    %                                increases Fail by 0.1, can not go
    %                                beow High Branch Value


      Fail_feed = Fail_La;

    if Fail_feed > Fail
        Fail = Fail_feed;
    end
%end

% - Simplify Fail to 0, 0.4, 0.7, or 1

if Fail < 1 && Fail >= 0.7
        Fail = 0.7;
elseif Fail < 0.7 && Fail >= 0.4
        Fail = 0.4;
elseif Fail < 0.4 && Fail >= 0.05
        Fail = 0.05;
elseif Fail < 0.05 && Fail >= 0
        Fail = 0;
end


% Outputs failed components as portion of total
```

```
%Fraw = (SD_115_1a + CB_115_1+ IT_115_1 + SD_115_1b + Fail_CH + sum(Fail_mat, 'all'))/(5 + fe
ed*6);

Fraw_CB =  (CB_115_a + CB_57_a)/(2);

Fraw_SD =  (SD_115_1a + SD_115_2a + SD_57_1a + SD_57_2a)/(4);

%Fraw_IT =  (IT_115_1)/(1 + feed *0); % Only one instrument transformer here

%Fraw_CH = Fail_CH; % This data isn't designed for partial damage to control house. This may
not be a useful data point.

Fraw = max([Fraw_CB, Fraw_SD]); % Not using IT or Ch here, too few data points

% - Simplify Fraw to 0, 0.05, 0.4, 0.7, or 1

if Fraw < 1 && Fraw >= 0.7
        Fraw = 0.7;
elseif Fraw < 0.7 && Fraw >= 0.4
        Fraw = 0.4;
elseif Fraw < 0.4 && Fraw >= 0.05
        Fraw = 0.05;
elseif Fraw < 0.05 && Fraw >= 0
        Fraw = 0;
end
```

```
end
```

Published with MATLAB® R2018b

B.3 RB Function

Contents


```
function [Fail, Fraw] = model_sub_RING(feed, PGA, Spec)
```

```
%This function tests component-level failures on a ring bus for use in a

%We are using a simple ring bus here, with two circuit breakers linking each bus, and a trans
former linking the line to the bus
%
%-------[Xformer]----------/----[52]----/--------[Xformer]------------
%                  |                       |
%                  |                       |
%                  /                       /
%                  |                       |
%                [52]                    [52]
%                  |                       |
%                  /                       /
%                  |                       |
%                  |                       |
%                  -----------------------------[Xformer]--------

%We will be treating each component as a flat pass/fail, then getting
%values to link a, b, and c viability.
```

Initialization

Get Component Data

```
    [Fail_SD_57, Fail_CS_57, Fail_IT_57, Fail_GCB_57, Fail_OCB_57, Fail_LTCPT_57, Fail_CH_57,
 ...
    Fail_SD_115, Fail_CS_115, Fail_IT_115, Fail_GCB_115, Fail_OCB_115, Fail_LTCPT_115, Fail_N
LTCPT_115, Fail_CH_115, ...
    Fail_SD_230, Fail_IT_230, Fail_GCB_230, Fail_OCB_230, Fail_LTCPT_230, Fail_LTCPT2_230, Fa
il_CH_230] = Comp_Fail(PGA, Spec);
%Feeds PGA, and Ground Response Spectra, returns failure of component type

Fail = 0;
```

```
Not enough input arguments.

Error in model_sub_RING (line 35)
    Fail_SD_230, Fail_IT_230, Fail_GCB_230, Fail_OCB_230, Fail_LTCPT_230, Fail_LTCPT2_230, Fa
il_CH_230] = Comp_Fail(PGA, Spec);
```

Ring Bus Failure State - Components

```
%calculate each component's failure

CH_115_1 = Fail_Chance(Fail_CH_115); % Failure chance for the control house (automatic bump t
o severe)

%High-voltage, feeder 1

SD_115_1a = Fail_Chance(Fail_SD_115);

CB_115_1 = Fail_Chance(Fail_GCB_115);

LTCPT_115_1 = Fail_Chance(Fail_LTCPT_115);

SD_115_1b = Fail_Chance(Fail_SD_115);

% Bus interconnects (Different ones will be used based on if there are 3 or
% 4 feeders)

% 1 - 2 (All)

SD_57_12a = Fail_Chance(Fail_SD_57); % Failure chance for the first switch connecting buses 1
 and 2

CB_57_12 = Fail_Chance(Fail_GCB_57);

SD_57_12b = Fail_Chance(Fail_SD_57);

% 2 - 3 (All)

SD_57_23a = Fail_Chance(Fail_SD_57);

CB_57_23 = Fail_Chance(Fail_GCB_57);

SD_57_23b = Fail_Chance(Fail_SD_57);

% 3 - 4 (4 feeders)
if feed == 4
    SD_57_34a = Fail_Chance(Fail_SD_57);

    CB_57_34 = Fail_Chance(Fail_GCB_57);

    SD_57_34b = Fail_Chance(Fail_SD_57);
end

% End - 1 (All)

SD_57_E1a = Fail_Chance(Fail_SD_57);

CB_57_E1 = Fail_Chance(Fail_GCB_57);

SD_57_E1b = Fail_Chance(Fail_SD_57);
```

Line Failure State - Substation connections

```
% 1 means failure!

if CH_115_1 == 0
    Fail_CH = 0;
else
    Fail_CH = 1;
end

if SD_115_1a == 0 && CB_115_1 == 0 && LTCPT_115_1 == 0 && SD_115_1b == 0
    Fail_HS = 0;
else
    Fail_HS = 1;
end

if feed == 3  % 3-Feeder scenario

    % Failure on line 1-2
    if (SD_57_12a == 0 && CB_57_12 == 0 && SD_57_12b == 0) || (SD_57_23a == 0 && CB_57_23 ==
0 && SD_57_23b == 0 ...
            && SD_57_E1a == 0 && CB_57_E1 == 0 && SD_57_E1b == 0)
        Fail_12 = 0;
    else
        Fail_12 = 1;
    end

    % Failure on line 2-3
    if (SD_57_23a == 0 && CB_57_23 == 0 && SD_57_23b == 0) || (SD_57_12a == 0 && CB_57_12 ==
0 && SD_57_12b == 0 ...
            && SD_57_E1a == 0 && CB_57_E1 == 0 && SD_57_E1b == 0)
        Fail_23 = 0;
    else
        Fail_23 = 1;
    end

    % Failure on line 3-1
    if (SD_57_E1a == 0 && CB_57_E1 == 0 && SD_57_E1b == 0) || (SD_57_12a == 0 && CB_57_12 ==
0 && SD_57_12b == 0 ...
            && SD_57_23a == 0 && CB_57_23 == 0 && SD_57_23b == 0)
        Fail_E1 = 0;
    else
        Fail_E1 = 1;
    end

elseif feed == 4  % 4-feeder scenario

        % Failure on line 1-2
    if (SD_57_12a == 0 && CB_57_12 == 0 && SD_57_12b == 0) || (SD_57_23a == 0 && CB_57_23 ==
0 && SD_57_23b == 0 ...
            && SD_57_34a == 0 && CB_57_34 == 0 && SD_57_34b == 0 && SD_57_E1a == 0 && CB_57_E
1 == 0 && SD_57_E1b == 0)
        Fail_12 = 0;
    else
        Fail_12 = 1;
    end
```

```
    % Failure on line 2-3
    if (SD_57_23a == 0 && CB_57_23 == 0 && SD_57_23b == 0) || (SD_57_12a == 0 && CB_57_12 ==
0 && SD_57_12b == 0 ...
            && SD_57_34a == 0 && CB_57_34 == 0 && SD_57_34b == 0 && SD_57_E1a == 0 && CB_57_E
1 == 0 && SD_57_E1b == 0)
        Fail_23 = 0;
    else
        Fail_23 = 1;
    end

        % Failure on line 3-4
    if (SD_57_34a == 0 && CB_57_34 == 0 && SD_57_34b == 0) || (SD_57_12a == 0 && CB_57_12 ==
0 && SD_57_12b == 0 ...
            && SD_57_23a == 0 && CB_57_23 == 0 && SD_57_23b == 0 && SD_57_E1a == 0 && CB_57_E
1 == 0 && SD_57_E1b == 0)
        Fail_34 = 0;
    else
        Fail_34 = 1;
    end

    % Failure on line 4-1
    if (SD_57_E1a == 0 && CB_57_E1 == 0 && SD_57_E1b == 0) || (SD_57_12a == 0 && CB_57_12 ==
0 && SD_57_12a == 0 ...
            && SD_57_23a == 0 && CB_57_23 == 0 && SD_57_23a == 0 && SD_57_34a == 0 && CB_57_3
4 == 0 && SD_57_34b == 0)
        Fail_E1 = 0;
    else
        Fail_E1 = 1;
    end

end
```

Create substation failure percentage

```
% To compare with Hazus - Slight = 0.1, Moderate = 0.4, Severe = 0.7,
% Complete = 1.0

if Fail_HS == 1
    Fail = 0.2;
end

if Fail_CH == 1
    Fail = 0.7;
end

if feed == 3
    Feed_mat = [Fail_12, Fail_23, Fail_E1]; % Feeder matrix

    Fail_feed = sum(Feed_mat==1)/3;

    if Fail_feed > Fail
        Fail = Fail_feed;
    end
end

if feed == 4
```

```
    Feed_mat = [Fail_12, Fail_23, Fail_34, Fail_E1]; % Feeder matrix

    Fail_feed = sum(Feed_mat==1)/4;

    if Fail_feed > Fail
        Fail = Fail_feed;
    end
end

% - Simplify Fail to 0.05, 0.4, 0.7, or 1

if Fail < 1 && Fail >= 0.7
        Fail = 0.7;
elseif Fail < 0.7 && Fail >= 0.4
        Fail = 0.4;
elseif Fail < 0.4 && Fail >= 0.05
        Fail = 0.05;
elseif Fail < 0.05 && Fail >= 0
        Fail = 0;
end

% Outputs failed components as portion of total
%Fraw = (SD_115_1a + CB_115_1+ IT_115_1 + SD_115_1b + Fail_CH + sum(Fail_mat, 'all'))/(5 + fe
ed*6);
if feed == 3
Fraw_CB =  (CB_115_1 + CB_57_12 + CB_57_23 + CB_57_E1)/(4);

Fraw_SD =  (SD_115_1a + SD_115_1b + SD_57_12a + SD_57_12b + SD_57_23a + SD_57_23b + SD_57_E1a
 + SD_57_E1b)/(8);

elseif feed == 4
Fraw_CB =  (CB_115_1 + CB_57_12 + CB_57_23 + CB_57_34 + CB_57_E1)/(5);

Fraw_SD =  (SD_115_1a + SD_115_1b + SD_57_12a + SD_57_12b + SD_57_23a + SD_57_23b + SD_57_34a
 + SD_57_34b + SD_57_E1a + SD_57_E1b)/(10);
end
%Fraw_IT =  (IT_115_1)/(1 + feed *0); % Only one instrument transformer here

%Fraw_CH = Fail_CH; % This data isn't designed for partial damage to control house. This may
not be a useful data point.

Fraw = max([Fraw_CB, Fraw_SD]); % Not using IT or Ch here, too few data points

% - Simplify Fraw to 0, 0.05, 0.4, 0.7, or 1

if Fraw < 1 && Fraw >= 0.7
        Fraw = 0.7;
elseif Fraw < 0.7 && Fraw >= 0.4
        Fraw = 0.4;
elseif Fraw < 0.4 && Fraw >= 0.05
        Fraw = 0.05;
elseif Fraw < 0.05 && Fraw >= 0
        Fraw = 0;
end
```

```
end
```

Published with MATLAB® R2018b

B.4 BAH Function

Contents


```
function [Fail, Fraw] = model_sub_BAH(feed, PGA, Spec)
```

```
%We are using a breaker-and-a-half configuration, with two feeders and
%three breakers between each bus. An odd number of feeders will result in
%breakers on either side.
%


%We will be treating each component as a flat pass/fail, then getting
%values to link a, b, and c viability.
```

Initialization

Get Component Data

```
    [Fail_SD_57, Fail_CS_57, Fail_IT_57, Fail_GCB_57, Fail_OCB_57, Fail_LTCPT_57, Fail_CH_57,
 ...
    Fail_SD_115, Fail_CS_115, Fail_IT_115, Fail_GCB_115, Fail_OCB_115, Fail_LTCPT_115, Fail_N
LTCPT_115, Fail_CH_115, ...
    Fail_SD_230, Fail_IT_230, Fail_GCB_230, Fail_OCB_230, Fail_LTCPT_230, Fail_LTCPT2_230, Fa
il_CH_230] = Comp_Fail(PGA, Spec);
%Feeds PGA, and Ground Response Spectra, returns failure of component type

Fail = 0;
```

```
Not enough input arguments.

Error in model_sub_BAH (line 25)
    Fail_SD_230, Fail_IT_230, Fail_GCB_230, Fail_OCB_230, Fail_LTCPT_230, Fail_LTCPT2_230, Fa
il_CH_230] = Comp_Fail(PGA, Spec);
```

Ring Bus Failure State - Components

```
%High-voltage, feeder 1 - Always

SD_115_1a = Fail_Chance(Fail_SD_115);

CB_115_1 = Fail_Chance(Fail_GCB_115);
```

```
LTCPT_115_1 = Fail_Chance(Fail_LTCPT_115);

SD_115_1b = Fail_Chance(Fail_SD_115);



CH_115_1 = Fail_Chance(Fail_CH_115); % Failure chance for the control house (automatic bump t
o severe)

% Loop to make A-bus and B-bus

Fail_mat = [3,feed];

% Create middle breakers
if mod(feed,2)==1 % Feed is an odd number
    Fail_mat_mid = [3,round(feed/2)+1];
else              % Feed is even
    Fail_mat_mid = [3,feed/2];
end

mid_count = 1; %Used to step through middle breaker matrix

for k = 1:feed

        %For each feeder, 1 sbs line

    Fail_mat(1,k) = Fail_Chance(Fail_SD_57); % First SD, Bus A

    Fail_mat(2,k) = Fail_Chance(Fail_GCB_57); % CB, Bus A

    Fail_mat(3,k) = Fail_Chance(Fail_SD_57); % Second SD, Bus A



    if mod(k,2) == 1 % Every odd step will create a secondary breaker, ensures that 1 is crea
ted every 2, plus an extra if odd

        Fail_mat_mid(1,mid_count) = Fail_Chance(Fail_SD_57); % First SD, Middle Breaker

        Fail_mat_mid(2,mid_count) = Fail_Chance(Fail_GCB_57); % CB, Middle Breaker

        Fail_mat_mid(3,mid_count) = Fail_Chance(Fail_SD_57); % Second SD, Middle Breaker

        mid_count = mid_count + 1;
    end

end
```

Line Failure State - Substation connections

```
% 1 means failure!

% if CH_115_1 == 0    % Control House Failure
%     Fail_CH = 0;
% else
%     Fail_CH = 1;
```

```
% end
Fail_CH = CH_115_1;

if SD_115_1a == 0 && CB_115_1 == 0 && LTCPT_115_1 == 0 && SD_115_1b == 0 % High-side line fai
lure
    Fail_HS = 0;
else
    Fail_HS = 1;
end

% Create the matrices that will hold 0s and 1s to indicate failure

Fail2_mat = [1 feed]; % Contains the bus-side breakers for each feeder
Fail2_mat_mid = [1 length(Fail_mat_mid)]; % Contains the breaker between the feeders


for i = 1:feed    %Computed failure for the breaker linking each feeder to the bus. Odd feede
rs to bus 'a' and even to bus 'b'

    if Fail_mat(1, i) == 0 && Fail_mat(2,i) == 0 && Fail_mat(3,i) == 0
        Fail2_mat(1,i) = 0;
    else
        Fail2_mat(1,i) = 1;
    end

end

for i = 1:length(Fail_mat_mid)
    if Fail_mat_mid(1, i) == 0 && Fail_mat_mid(2,i) == 0 && Fail_mat_mid(3,i) == 0
        Fail2_mat_mid(i) = 0;
    else
        Fail2_mat_mid(i) = 1;
    end
end
```

Create substation failure percentage

```
% To compare with Hazus - Slight = 0.1, Moderate = 0.4, Extensive = 0.7,
% Complete = 1.0

if Fail_HS == 1
    Fail = 0.2;
end

if Fail_CH == 1
    Fail = 0.7;
end




% For even, every one will have near- and far-side. Near-side will depend
% only on that side's connection, while Far will rely on other 2. Each
% line's failure will add 0.1, with a 1 if both sides are downed

Fail_count = 0; % Counts individual line failures
```

```
qm = 1; % Will be used to iterate the middle breaker array


for q = 1:2:feed % q and q+1 are feeder numbers being checked. Both will be checked together

    if q ~= feed % Analysis for all steps except last odd (q == feed only exists when feed is
 odd, see below)
            % XXX
        if Fail2_mat(q) == 1 && Fail2_mat(q+1) == 1 && Fail2_mat_mid(qm) == 1 % All three sbs
 lines fail, both feeders isolated
            Fail_count = Fail_count + 2;
            % --X
        elseif (Fail2_mat(q) == 1 || Fail2_mat(q+1) == 1) && ~(Fail2_mat(q) == 1 && Fail2_mat
(q+1) == 1) && Fail2_mat_mid(qm) == 0
            % One feeder-bus sbs failed, other two still connected, all feeders
            % connected
            Fail_count = Fail_count + 0.1;
            % -X-
        elseif Fail2_mat(q) == 0 && Fail2_mat(q+1) == 0 && Fail2_mat_mid(qm) == 1 % Only midd
le sbs failed. All feeders connected
            Fail_count = Fail_count + 0.1;
            % X-X
        elseif Fail2_mat(q) == 1 && Fail2_mat(q+1) == 1 && Fail2_mat_mid(qm) == 0 % Both oute
r lines fail, middle sbs connected, feeder q & q+1 only connected
            Fail_count = Fail_count + 1;
            % XX-
        elseif (Fail2_mat(q) == 1 || Fail2_mat(q+1) == 1) && ~(Fail2_mat(q) == 1 && Fail2_mat
(q+1) == 1) && Fail2_mat_mid(qm) == 1
            % One outer sbs and middle failed, one outer connected. One
            % feeder completely out
            Fail_count = Fail_count + 1.1;
        end

    end

    if q == feed %If there are an odd number of feeders, this will kick in at the last feeder
. Analysis here is identical to DBDB
        if Fail2_mat(1,q) == 1 && Fail2_mat_mid(qm) == 1
        Fail_count = Fail_count + 1;
        elseif (Fail2_mat(1,q) == 1 || Fail2_mat_mid(qm) == 1) && ~(Fail2_mat(1,q) == 1 && Fa
il2_mat_mid(qm) == 1)
        Fail_count = Fail_count + 0.1;
        end

    end
    qm = qm +1;

end



Fail_count = Fail_count/feed;

if Fail_count > Fail
    Fail = Fail_count;
end
```

```
if Fail > 1
    Fail = 1;
end

% - Simplify Fail to 0, 0.4, 0.7, or 1

if Fail < 1 && Fail >= 0.7
        Fail = 0.7;
elseif Fail < 0.7 && Fail >= 0.4
        Fail = 0.4;
elseif Fail < 0.4 && Fail >= 0.05
        Fail = 0.05;
elseif Fail < 0.05 && Fail >= 0
        Fail = 0;
end


% Outputs failed components as portion of total
%Fraw = (SD_115_1a + CB_115_1+ IT_115_1 + SD_115_1b + Fail_CH + sum(Fail_mat, 'all'))/(5 + fe
ed*6);

Fraw_CB =  (CB_115_1 + sum(Fail_mat(2,:), 'all') + sum(Fail_mat_mid(2,:), 'all'))/(1 + feed +
 length(Fail_mat_mid));

Fraw_SD =  (SD_115_1a + SD_115_1b + sum(Fail_mat([1,3],:), 'all') + sum(Fail_mat_mid([1,3],:)
, 'all'))/(2 + 2*feed + 2*length(Fail_mat_mid));

%Fraw_IT =  (IT_115_1)/(1 + feed *0); % Only one instrument transformer here

%Fraw_CH = Fail_CH; % This data isn't designed for partial damage to control house. This may
not be a useful data point.

Fraw = max([Fraw_CB, Fraw_SD]); % Not using IT or Ch here, too few data points

% - Simplify Fraw to 0, 0.05 0.4, 0.7, or 1

if Fraw < 1 && Fraw >= 0.7
        Fraw = 0.7;
elseif Fraw < 0.7 && Fraw >= 0.4
        Fraw = 0.4;
elseif Fraw < 0.4 && Fraw >= 0.05
        Fraw = 0.05;
elseif Fraw < 0.05 && Fraw >= 0
        Fraw = 0;
end
```

```
end
```

Published with MATLAB® R2018b

B.5 DBDB Function

Contents


```
function [Fail, Fraw] = model_sub_DBDB(feed, PGA, Spec)
```

```
%This function tests component-level failures on a ring bus for use in a

%We are using a brekaer-and-a-half configuration, with two feeders and
%three breakers between each bus. An odd number of feeders will result in
%breakers on either side.
%


%We will be treating each component as a flat pass/fail, then getting
%values to link a, b, and c viability.
```

Initialization

Get Component Data

```
    [Fail_SD_57, Fail_CS_57, Fail_IT_57, Fail_GCB_57, Fail_OCB_57, Fail_LTCPT_57, Fail_CH_57,
 ...
    Fail_SD_115, Fail_CS_115, Fail_IT_115, Fail_GCB_115, Fail_OCB_115, Fail_LTCPT_115, Fail_N
LTCPT_115, Fail_CH_115, ...
    Fail_SD_230, Fail_IT_230, Fail_GCB_230, Fail_OCB_230, Fail_LTCPT_230, Fail_LTCPT2_230, Fa
il_CH_230] = Comp_Fail(PGA, Spec);
%Feeds PGA, and Ground Response Spectra, returns failure of component type

Fail = 0;
```

```
Not enough input arguments.

Error in model_sub_DBDB (line 27)
    Fail_SD_230, Fail_IT_230, Fail_GCB_230, Fail_OCB_230, Fail_LTCPT_230, Fail_LTCPT2_230, Fa
il_CH_230] = Comp_Fail(PGA, Spec);
```

Ring Bus Failure State - Components

```
%High-voltage, feeder 1 - Always

SD_115_1a = Fail_Chance(Fail_SD_115);
```

```
CB_115_1 = Fail_Chance(Fail_GCB_115);

LTCPT_115_1 = Fail_Chance(Fail_LTCPT_115);

SD_115_1b = Fail_Chance(Fail_SD_115);



CH_115_1 = Fail_Chance(Fail_CH_115); % Failure chance for the control house (automatic bump t
o severe)

% Loop to make A-bus and B-bus

Fail_mat = [6,feed];

for k = 1:feed

        %Bus A

    Fail_mat(1,k) = Fail_Chance(Fail_SD_57); % First SD, Bus A

    Fail_mat(2,k) = Fail_Chance(Fail_GCB_57); % CB, Bus A

    Fail_mat(3,k) = Fail_Chance(Fail_SD_57); % Second SD, Bus A

        %Bus B

    Fail_mat(4,k) = Fail_Chance(Fail_SD_57); % First SD, Bus B

    Fail_mat(5,k) = Fail_Chance(Fail_GCB_57); % CB, Bus B

    Fail_mat(6,k) = Fail_Chance(Fail_SD_57); % Second SD, Bus B

end
```

Line Failure State - Substation connections

```
% 1 means failure!

% if CH_115_1 == 0    % Control House Failure
%     Fail_CH = 0;
% else
%     Fail_CH = 1;
% end
Fail_CH = CH_115_1;

if SD_115_1a == 0 && CB_115_1 == 0 && LTCPT_115_1 == 0 && SD_115_1b == 0 % High-side line fai
lure
    Fail_HS = 0;
else
    Fail_HS = 1;
end



Fail2_mat = [2 feed];
```

```
for i = 1:feed

    if Fail_mat(1, i) == 0 && Fail_mat(2,i) == 0 && Fail_mat(3,i) == 0
        Fail2_mat(1,i) = 0;
    else
        Fail2_mat(1,i) = 1;
    end

    if Fail_mat(4, i) == 0 && Fail_mat(5,i) == 0 && Fail_mat(6,i) == 0
        Fail2_mat(2,i) = 0;
    else
        Fail2_mat(2,i) = 1;
    end

end
```

Create substation failure percentage

```
% To compare with Hazus - Slight = 0.1, Moderate = 0.4, Extensive = 0.7,
% Complete = 1.0

if Fail_HS == 1
    Fail = 0.2;
end

if Fail_CH == 1
    Fail = 0.7;
end

Fail_count = 0;

for q = 1:feed

    if Fail2_mat(1,q) == 1 && Fail2_mat(2,q) == 1
        Fail_count = Fail_count + 1;
    elseif (Fail2_mat(1,q) == 1 || Fail2_mat(2,q) == 1) && ~(Fail2_mat(1,q) == 1 && Fail2_mat
(2,q) == 1)
        Fail_count = Fail_count + 0.1;
    end

end

Fail_count = Fail_count/feed;

if Fail_count > Fail
    Fail = Fail_count;
end

if Fail > 1
    Fail = 1;
end

% - Simplify Fail to 0, 0.4, 0.7, or 1

if Fail < 1 && Fail >= 0.7
```

```
        Fail = 0.7;
elseif Fail < 0.7 && Fail >= 0.4
        Fail = 0.4;
elseif Fail < 0.4 && Fail >= 0.05
        Fail = 0.05;
elseif Fail < 0.05 && Fail >= 0
        Fail = 0;
end


% Outputs failed components as portion of total
%Fraw = (SD_115_1a + CB_115_1+ IT_115_1 + SD_115_1b + Fail_CH + sum(Fail_mat, 'all'))/(5 + fe
ed*6);

Fraw_CB =  (CB_115_1 + sum(Fail_mat([2,5],:), 'all'))/(1 + feed *2);

Fraw_SD =  (SD_115_1a + SD_115_1b + sum(Fail_mat([1,3,4,6],:), 'all') )/(2 + feed *4);

%Fraw_IT =  (IT_115_1)/(1 + feed *0); % Only one instrument transformer here

%Fraw_CH = Fail_CH; % This data isn't designed for partial damage to control house. This may
not be a useful data point.

Fraw = max([Fraw_CB, Fraw_SD]); % Not using IT or Ch here, too few data points

% - Simplify Fraw to 0, 0.05 0.4, 0.7, or 1

if Fraw < 1 && Fraw >= 0.7
        Fraw = 0.7;
elseif Fraw < 0.7 && Fraw >= 0.4
        Fraw = 0.4;
elseif Fraw < 0.4 && Fraw >= 0.05
        Fraw = 0.05;
elseif Fraw < 0.05 && Fraw >= 0
        Fraw = 0;
end
```

```
end
```

Published with MATLAB® R2018b

B.6 SEFT Failure Function

Contents


```
function [Fail_SD_57, Fail_CS_57, Fail_IT_57, Fail_GCB_57, Fail_OCB_57, Fail_LTCPT_57, Fail_C
H_57, ...
    Fail_SD_115, Fail_CS_115, Fail_IT_115, Fail_GCB_115, Fail_OCB_115, Fail_LTCPT_115, Fail_N
LTCPT_115, Fail_CH_115, ...
    Fail_SD_230, Fail_IT_230, Fail_GCB_230, Fail_OCB_230, Fail_LTCPT_230, Fail_LTCPT2_230, Fa
il_CH_230] = Comp_Fail(PGA, Spec)
```

```
% NOTE: Currently modified to only return gas circuit breaker failure to
% test functionality.
```

Curve Table Generation

SEFT - SEFT Data

```
% 57 - 57 kV components,
% 115 - 115 kV components,
% 230 - 230 kV components

%SD - Switch, Disconnect (Rocking @ 57/115 kV, Anchorage Failure @ 230 kV)
%CS - Circuit Switcher (Insulating Column Damage @ 57 kV, Rocking/Anchorage Failure @ 115 kV,
 N/A @ 230 kV)
%IT - Transformer, Instrument (Anchorage Failure @ 57/115 kV, Rocking @ 230 kV)
%GCB - Circuit Breaker, Gas (Rocking @ 57 kV, Sliding @ 115 kV, Rocking/Anchorage Failure @ 2
30 kV)
%OCB - Circuit Breaker, Oil (Anchorage Failure @ 57 kV, Sliding @ 115/230 kV)
%LTCPT - Transformer, LTC Power (Overturning @ 57 kV, Sliding @ 115/230 kV)
%LTCPT2 - Transformer, LTC Power, type 2 (N/A  @ 57/115 kV, Anchorage Failure @ 230 kV)
%NLTCPT - Transformer, non-LTC Power (N/A @ 57 kV, Anchorage Failure @ 115 kV, N/A @ 230 kV)
%CH - Control House (Anchorage Failure @ 57/115/230 kV)

%Response spectra are in 3 groups, A, B, and C

%POF - Probability of Failure

% Matrix columns structured as:
% [ PGA     |    Spectra A POF  |    Spectra B POF  |    Spectra C POF  ]

% Only have failure and non-failure states

%generic x-axis matrix

xax = [
```

```
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0.35
0.40
0.45
0.50
0.55
0.60
0.65
0.70
0.75
0.80
0.85
0.90
0.95
1.00
1.10
1.20
1.30
1.40
1.50
1.60
1.70
1.80
1.90
2.00
];

% 57 kV Matrices ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~~

SEFT_57_SD = [
0.00    0.00    0.00
0.00    0.00    0.00
0.00    0.00    0.00
0.00    0.00    0.00
0.00    0.00    0.00
0.00    0.01    0.01
0.01    0.02    0.02
0.02    0.03    0.04
0.03    0.06    0.08
0.06    0.10    0.12
0.09    0.14    0.17
0.12    0.19    0.22
0.16    0.25    0.28
0.21    0.30    0.34
0.26    0.36    0.40
0.30    0.42    0.46
0.35    0.47    0.51
0.40    0.52    0.56
0.45    0.57    0.61
0.49    0.61    0.65
```

```
0.54    0.65    0.69
0.62    0.72    0.76
0.68    0.78    0.81
0.74    0.83    0.85
0.79    0.86    0.89
0.83    0.89    0.91
0.86    0.92    0.93
0.89    0.93    0.95
0.91    0.95    0.96
0.93    0.96    0.97
0.94    0.97    0.97
];

SEFT_57_CS = [
0.00    0.00    0.00
0.00    0.00    0.00
0.07    0.07    0.06
0.23    0.22    0.21
0.41    0.41    0.39
0.57    0.56    0.55
0.69    0.69    0.67
0.78    0.78    0.76
0.85    0.84    0.83
0.89    0.89    0.88
0.92    0.92    0.91
0.94    0.94    0.94
0.96    0.96    0.95
0.97    0.97    0.97
0.98    0.98    0.98
0.98    0.98    0.98
0.99    0.99    0.99
0.99    0.99    0.99
0.99    0.99    0.99
1.00    0.99    0.99
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
];

SEFT_57_IT = [
0.00    0.00    0.00
0.00    0.00    0.00
0.03    0.03    0.02
0.25    0.25    0.22
0.54    0.54    0.51
0.77    0.76    0.74
0.90    0.89    0.88
0.96    0.96    0.95
0.98    0.98    0.98
```

```
0.99    0.99    0.99
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00

];

SEFT_57_GCB = [
0.00    0.00    0.00
0.00    0.00    0.00
0.00    0.00    0.00
0.02    0.01    0.01
0.06    0.06    0.05
0.14    0.13    0.12
0.24    0.23    0.22
0.35    0.34    0.32
0.46    0.45    0.43
0.56    0.55    0.53
0.64    0.64    0.62
0.71    0.71    0.69
0.77    0.77    0.75
0.82    0.82    0.80
0.86    0.85    0.84
0.89    0.89    0.87
0.91    0.91    0.90
0.93    0.93    0.92
0.95    0.94    0.94
0.96    0.96    0.95
0.97    0.96    0.96
0.98    0.98    0.97
0.99    0.99    0.98
0.99    0.99    0.99
0.99    0.99    0.99
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
```

```
];

SEFT_57_OCB = [
0.00    0.00    0.00
0.00    0.00    0.00
0.06    0.06    0.05
0.48    0.47    0.52
0.90    0.89    0.87
0.99    0.99    0.98
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00


];

SEFT_57_LTCPT = [
0.00    0.00    0.00
0.00    0.00    0.00
0.00    0.00    0.00
0.02    0.02    0.02
0.07    0.07    0.07
0.15    0.15    0.15
0.25    0.25    0.25
0.35    0.35    0.35
0.45    0.45    0.45
0.55    0.55    0.55
0.63    0.63    0.63
0.70    0.70    0.70
0.75    0.75    0.75
0.80    0.80    0.80
0.84    0.84    0.84
0.87    0.87    0.87
0.90    0.90    0.90
```

```
0.92    0.92    0.92
0.93    0.93    0.93
0.94    0.94    0.94
0.96    0.96    0.96
0.97    0.97    0.97
0.98    0.98    0.98
0.99    0.99    0.99
0.99    0.99    0.99
0.99    0.99    0.99
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00


];

SEFT_57_CH = [
0.00    0.00    0.00
0.00    0.00    0.00
0.00    0.00    0.00
0.01    0.01    0.00
0.07    0.07    0.05
0.23    0.22    0.20
0.45    0.44    0.41
0.66    0.65    0.62
0.82    0.81    0.78
0.91    0.91    0.89
0.96    0.96    0.95
0.99    0.99    0.98
1.00    1.00    0.99
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00

];


% 115 kV Matrices ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~~~
```

```
SEFT_115_SD = [
0.00    0.00    0.00
0.00    0.00    0.00
0.00    0.00    0.00
0.00    0.00    0.00
0.00    0.00    0.00
0.01    0.01    0.01
0.03    0.03    0.02
0.05    0.05    0.05
0.09    0.09    0.08
0.14    0.14    0.12
0.20    0.19    0.18
0.26    0.25    0.23
0.32    0.31    0.29
0.38    0.37    0.35
0.44    0.43    0.41
0.50    0.49    0.47
0.55    0.55    0.52
0.60    0.60    0.57
0.65    0.64    0.62
0.69    0.68    0.66
0.73    0.72    0.70
0.79    0.78    0.77
0.84    0.83    0.82
0.88    0.87    0.86
0.91    0.90    0.89
0.93    0.92    0.92
0.94    0.94    0.94
0.96    0.96    0.95
0.97    0.97    0.96
0.97    0.97    0.97
0.98    0.98    0.98

];

SEFT_115_CS = [
0.00    0.00    0.00
0.00    0.00    0.00
0.00    0.00    0.00
0.00    0.00    0.00
0.00    0.00    0.00
0.01    0.01    0.01
0.01    0.01    0.02
0.03    0.03    0.03
0.06    0.06    0.06
0.09    0.09    0.09
0.13    0.13    0.14
0.18    0.18    0.19
0.24    0.23    0.24
0.29    0.29    0.29
0.35    0.34    0.35
0.40    0.40    0.41
0.45    0.45    0.46
0.51    0.50    0.51
0.55    0.55    0.56
0.60    0.59    0.60
```

```
0.64    0.64    0.64
0.71    0.71    0.71
0.77    0.77    0.77
0.82    0.82    0.82
0.86    0.85    0.86
0.89    0.88    0.89
0.91    0.91    0.91
0.94    0.94    0.93
0.96    0.96    0.96
0.98    0.98    0.97
0.99    0.99    0.98

];

SEFT_115_IT = [
0.00    0.00    0.00
0.00    0.00    0.00
0.02    0.03    0.03
0.13    0.17    0.16
0.33    0.37    0.35
0.53    0.56    0.54
0.68    0.70    0.68
0.79    0.80    0.79
0.86    0.87    0.86
0.91    0.92    0.91
0.94    0.95    0.94
0.96    0.97    0.96
0.98    0.98    0.98
0.99    0.99    0.99
0.99    0.99    0.99
1.00    1.00    0.99
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00

];

SEFT_115_GCB = [
0.00    0.00    0.00
0.00    0.00    0.00
0.00    0.00    0.00
0.01    0.01    0.01
0.06    0.05    0.05
0.13    0.13    0.12
0.23    0.23    0.21
```

```
0.34    0.34    0.32
0.45    0.44    0.42
0.55    0.54    0.52
0.64    0.63    0.61
0.71    0.70    0.68
0.77    0.76    0.75
0.82    0.81    0.80
0.85    0.85    0.84
0.89    0.88    0.87
0.91    0.91    0.90
0.93    0.93    0.92
0.94    0.94    0.93
0.96    0.95    0.95
0.96    0.96    0.96
0.98    0.98    0.97
0.99    0.99    0.98
0.99    0.99    0.99
0.99    0.99    0.99
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00

];

SEFT_115_OCB = [
0.00    0.00    0.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
```

```
1.00    1.00    1.00
1.00    1.00    1.00

];

SEFT_115_LTCPT = [
0.00    0.00    0.00
0.00    0.00    0.00
0.00    0.00    0.00
0.01    0.01    0.01
0.05    0.05    0.05
0.12    0.12    0.12
0.20    0.20    0.20
0.30    0.30    0.30
0.40    0.40    0.40
0.49    0.49    0.49
0.57    0.57    0.57
0.65    0.65    0.65
0.71    0.71    0.71
0.76    0.76    0.76
0.80    0.80    0.80
0.84    0.84    0.84
0.87    0.87    0.87
0.89    0.89    0.89
0.91    0.91    0.91
0.93    0.93    0.93
0.94    0.94    0.94
0.96    0.96    0.96
0.97    0.97    0.97
0.98    0.98    0.98
0.99    0.99    0.99
0.99    0.99    0.99
0.99    0.99    0.99
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00

];

SEFT_115_NLTCPT = [
0.00    0.00    0.00
0.00    0.00    0.00
0.04    0.04    0.04
0.18    0.18    0.18
0.98    0.98    0.98
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
```

```
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00

];

SEFT_115_CH = [
0.00    0.00    0.00
0.00    0.00    0.00
0.00    0.00    0.00
0.04    0.04    0.03
0.32    0.31    0.27
0.71    0.69    0.65
0.93    0.92    0.90
0.99    0.99    0.98
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00

];


% 230 kV Matrices ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~~~
```

```
SEFT_230_SD = [
0.00    0.00    0.00
0.00    0.00    0.00
0.00    0.01    0.02
0.01    0.05    0.11
0.06    0.15    0.28
0.14    0.29    0.45
0.25    0.43    0.60
0.37    0.56    0.72
0.48    0.67    0.80
0.57    0.75    0.87
0.66    0.81    0.91
0.73    0.86    0.94
0.78    0.90    0.96
0.83    0.93    0.97
0.87    0.95    0.98
0.89    0.96    0.99
0.92    0.98    0.99
0.94    0.98    1.00
0.95    0.99    1.00
0.97    0.99    1.00
0.97    0.99    1.00
0.99    1.00    1.00
0.99    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00

];

SEFT_230_IT = [
0.00    0.00    0.00
0.00    0.00    0.00
0.00    0.00    0.00
0.00    0.00    0.00
0.00    0.01    0.01
0.01    0.02    0.02
0.03    0.05    0.04
0.06    0.09    0.08
0.10    0.15    0.14
0.15    0.21    0.20
0.21    0.28    0.27
0.27    0.35    0.34
0.33    0.42    0.40
0.39    0.49    0.47
0.46    0.55    0.53
0.51    0.61    0.59
0.57    0.66    0.64
0.62    0.70    0.69
0.66    0.74    0.73
0.70    0.78    0.77
```

```
0.74    0.81    0.80
0.80    0.86    0.85
0.85    0.90    0.89
0.88    0.92    0.92
0.91    0.94    0.94
0.93    0.96    0.95
0.95    0.97    0.97
0.96    0.98    0.97
0.97    0.98    0.98
0.98    0.99    0.99
0.98    0.99    0.99

];

SEFT_230_GCB = [
0.00    0.00    0.00
0.00    0.00    0.00
0.00    0.00    0.00
0.01    0.01    0.01
0.04    0.04    0.04
0.11    0.11    0.10
0.20    0.19    0.18
0.30    0.29    0.28
0.41    0.40    0.38
0.50    0.50    0.47
0.59    0.58    0.56
0.67    0.66    0.64
0.73    0.72    0.71
0.78    0.78    0.76
0.83    0.82    0.81
0.86    0.86    0.84
0.89    0.89    0.87
0.91    0.91    0.90
0.93    0.93    0.92
0.96    0.95    0.93
0.98    0.98    0.97
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00

];

SEFT_230_OCB = [
0.00    0.00    0.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
```

```
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00

];

SEFT_230_LTCPT = [
0.00    0.00    0.00
0.00    0.00    0.00
0.00    0.00    0.00
0.01    0.01    0.01
0.05    0.05    0.05
0.11    0.11    0.11
0.20    0.20    0.20
0.29    0.29    0.29
0.39    0.39    0.39
0.48    0.48    0.48
0.57    0.57    0.57
0.64    0.64    0.64
0.70    0.70    0.70
0.75    0.75    0.75
0.80    0.80    0.80
0.83    0.83    0.83
0.86    0.86    0.86
0.89    0.89    0.89
0.91    0.91    0.91
0.92    0.92    0.92
0.94    0.94    0.94
0.96    0.96    0.96
0.97    0.97    0.97
0.98    0.98    0.98
0.99    0.99    0.99
0.99    0.99    0.99
0.99    0.99    0.99
1.00    1.00    1.00
1.00    1.00    1.00
```

```
1.00    1.00    1.00
1.00    1.00    1.00

];

SEFT_230_LTCPT2 = [
0.00    0.00    0.00
0.00    0.00    0.00
0.05    0.05    0.05
0.20    0.20    0.20
0.41    0.41    0.41
0.59    0.59    0.59
0.73    0.73    0.73
0.83    0.83    0.83
0.89    0.89    0.89
0.93    0.93    0.93
0.95    0.95    0.95
0.97    0.97    0.97
0.98    0.98    0.98
0.99    0.99    0.99
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00

];

SEFT_230_CH = [
0.00    0.00    0.00
0.00    0.00    0.00
0.00    0.00    0.00
0.02    0.02    0.02
0.08    0.08    0.07
0.18    0.17    0.16
0.30    0.29    0.27
0.42    0.41    0.39
0.53    0.52    0.50
0.62    0.62    0.60
0.70    0.70    0.68
0.77    0.76    0.75
0.82    0.82    0.80
0.86    0.86    0.84
0.89    0.89    0.88
0.92    0.91    0.91
```

```
0.94    0.93    0.93
0.95    0.95    0.94
0.96    0.96    0.96
0.97    0.97    0.96
0.98    0.98    0.97
0.99    0.99    0.98
0.99    0.99    0.99
0.99    0.99    0.99
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00
1.00    1.00    1.00

];
```

Initial - Create output variables

```
%     Fail_SD57 = 0;
%     Fail_CS_57 = 0;
%     Fail_IT_57 = 0;
%     Fail_GCB_57 = 0;
%     Fail_OCB_57 = 0;
%     Fail_LTCPT_57 = 0;
%     Fail_CH_57 = 0;
%
%     Fail_SD_115 = 0;
%     Fail_CS_115 = 0;
%     Fail_IT_115 = 0;
%     Fail_GCB_115 = 0;
%     Fail_OCB_115 = 0;
%     Fail_LTCPT_115 = 0;
%     Fail_NLTCPT_115 = 0;
%     Fail_CH_115 = 0;
%
%     Fail_SD_230 = 0;
%     Fail_IT_230 = 0;
%     Fail_GCB_230 = 0;
%     Fail_OCB_230 = 0;
%     Fail_LTCPT_230 = 0;
%     Fail_LTCPT2_230 = 0;
%     Fail_CH_230 = 0;
```

Input

```
%To be inherited from call on parent script

%Ground Acceleration (Variable PGA) Can be any positive number


if PGA < 0
    error('PGA cannot be negative')
```

```
end

%Response Spectra (Variable Spec) Can be A, B or C


if       strcmp(Spec,'a')
    SR = 1;
elseif   strcmp(Spec,'b')
    SR = 2;
elseif   strcmp(Spec,'c')
    SR = 3;
else
    error('Response Spectrum must be defined as A, B, or C')
end
```

```
Not enough input arguments.

Error in Comp_Fail (line 882)
if PGA < 0
```

Failure state generation by component

```
if PGA <= 2 % Non-maxed PGA, all voltages below
                                          % 57 Kv
    Fail57 = [
        interp1(xax, SEFT_57_SD(:,SR), PGA)
        interp1(xax, SEFT_57_CS(:,SR), PGA)
        interp1(xax, SEFT_57_IT(:,SR), PGA)
        interp1(xax, SEFT_57_GCB(:,SR), PGA)
        interp1(xax, SEFT_57_OCB(:,SR), PGA)
        interp1(xax, SEFT_57_LTCPT(:,SR), PGA)
        interp1(xax, SEFT_57_CH(:,SR), PGA)
        ];

                                          % 115 Kv
    Fail115 = [
        interp1(xax, SEFT_115_SD(:,SR), PGA)
        interp1(xax, SEFT_115_CS(:,SR), PGA)
        interp1(xax, SEFT_115_IT(:,SR), PGA)
        interp1(xax, SEFT_115_GCB(:,SR), PGA)
        interp1(xax, SEFT_115_OCB(:,SR), PGA)
        interp1(xax, SEFT_115_LTCPT(:,SR), PGA)
        interp1(xax, SEFT_115_NLTCPT(:,SR), PGA)
        interp1(xax, SEFT_115_CH(:,SR), PGA)
        ];

                                          % 230 Kv
    Fail230 = [
        interp1(xax, SEFT_230_SD(:,SR), PGA)
        interp1(xax, SEFT_230_IT(:,SR), PGA)
        interp1(xax, SEFT_230_GCB(:,SR), PGA)
        interp1(xax, SEFT_230_OCB(:,SR), PGA)
        interp1(xax, SEFT_230_LTCPT(:,SR), PGA)
```

```
        interp1(xax, SEFT_230_LTCPT2(:,SR), PGA)
        interp1(xax, SEFT_230_CH(:,SR), PGA)
        ];

else

    Fail57 = [
        SEFT_57_SD(end,SR)
        SEFT_57_CS(end,SR)
        SEFT_57_IT(end,SR)
        SEFT_57_GCB(end,SR)
        SEFT_57_OCB(end,SR)
        SEFT_57_LTCPT(end,SR)
        SEFT_57_CH(end,SR)
        ];

    Fail115 = [
        SEFT_115_SD(end,SR)
        SEFT_115_CS(end,SR)
        SEFT_115_IT(end,SR)
        SEFT_115_GCB(end,SR)
        SEFT_115_OCB(end,SR)
        SEFT_115_LTCPT(end,SR)
        SEFT_115_NLTCPT(end,SR)
        SEFT_115_CH(end,SR)
        ];

    Fail230 = [
        SEFT_230_SD(end,SR)
        SEFT_230_IT(end,SR)
        SEFT_230_GCB(end,SR)
        SEFT_230_OCB(end,SR)
        SEFT_230_LTCPT(end,SR)
        SEFT_230_LTCPT2(end,SR)
        SEFT_230_CH(end,SR)
        ];
        disp('PGA has exceeded mapped values, using failure recorded for PGA = 2.0')

end
```

Create output values to return from function

```
    Fail_SD_57 = Fail57(1);
    Fail_CS_57 = Fail57(2);
    Fail_IT_57 = Fail57(3);
    Fail_GCB_57 = Fail57(4);
    Fail_OCB_57 = Fail57(5);
    Fail_LTCPT_57 = Fail57(6);
    Fail_CH_57 = Fail57(7);

    Fail_SD_115 = Fail115(1);
    Fail_CS_115 = Fail115(2);
    Fail_IT_115 = Fail115(3);
    Fail_GCB_115 = Fail115(4);
    Fail_OCB_115 = Fail115(5);
    Fail_LTCPT_115 = Fail115(6);
```

```
    Fail_NLTCPT_115 = Fail115(7);
    Fail_CH_115 = Fail115(8);

    Fail_SD_230 = Fail230(1);
    Fail_IT_230 = Fail230(2);
    Fail_GCB_230 = Fail230(3);
    Fail_OCB_230 = Fail230(4);
    Fail_LTCPT_230 = Fail230(5);
    Fail_LTCPT2_230 = Fail230(6);
    Fail_CH_230 = Fail230(7);
```

```
end
```

Published with MATLAB® R2018b

B.7 Fail Chance Calculation Function

```
function [Fail] = Fail_Chance(chance)
% This function outputs a 1 if the component fails and a zero if it does
% not.

% Value recieveed is decimal probability of failure, ie 0.1 is 10% chance

pick = rand;
if pick >= chance %with 0.1 failure, random has a 90% chance to be higher, and not trigger fa
ilure
    Fail = 0;
else
    Fail = 1;
end
end
```

```
Not enough input arguments.

Error in Fail_Chance (line 8)
if pick >= chance %with 0.1 failure, random has a 90% chance to be higher, and not trigger fa
ilure
```

Published with MATLAB® R2018b

B.8 Plotting Code

Contents

Initialization

```
clear all
clc

% Importing Data

% Saved data is in form CONFIGURATION_DATA_FEEDER# (Ex. DBDB_Data_10

fprintf('Load saved data from matfiles folder.\n')

[Filename, Pathname] = uigetfile(fullfile(pwd,'matfiles','*.mat'), 'Select a simulation data
set');

load([Pathname , Filename]);

if loadcheck == 6.626
    fprintf(['File ' , Filename , ' loaded successfully!']);
end
```

```
Load saved data from matfiles folder.
File DBDB_Data_10.mat loaded successfully!
```

Modify Hazus Values to match SEFT layout

```
xPGA = [0:0.05:1.4];



    HSS_LV_S = interp1(HSS_LV_S_raw(:,1), HSS_LV_S_raw(:,2), xPGA);

    HSS_LV_M = interp1(HSS_LV_M_raw(:,1), HSS_LV_M_raw(:,2), xPGA);

    HSS_LV_E = interp1(HSS_LV_E_raw(:,1), HSS_LV_E_raw(:,2), xPGA);

    HSS_LV_C = interp1(HSS_LV_C_raw(:,1), HSS_LV_C_raw(:,2), xPGA);
```

Prepare to PLot

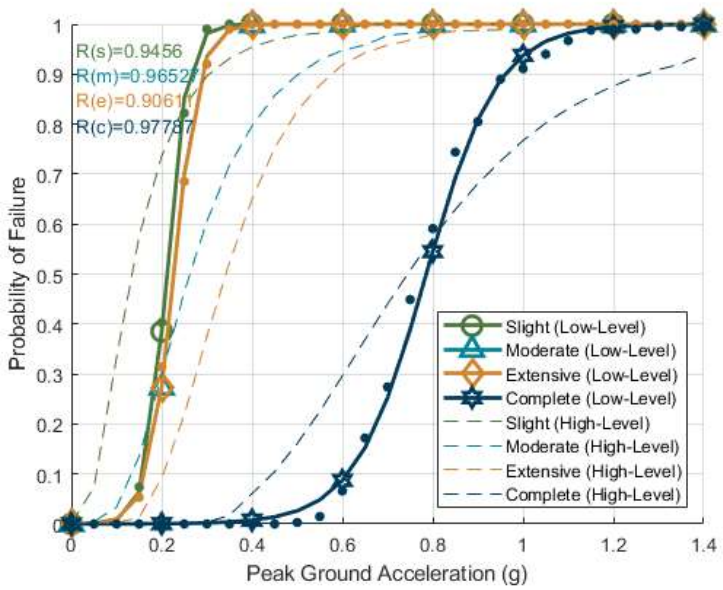
```
% Create power flow arrays
```

```
Flow_slight = States(2,:);
Flow_moderate = States(3,:);
Flow_extensive = States(4,:);
Flow_complete = States(5,:);

% Create Hazus Method Arrays

Haz_slight = S_raw(2,:);
Haz_moderate = S_raw(3,:);
Haz_extensive = S_raw(4,:);
Haz_complete = S_raw(5,:);

% Create Logistic curves for experimental component data

% Sourced from http://matlabdatamining.blogspot.com/2009/03/logistic-regression.html

% Additional information from
% https://www.mathworks.com/help/stats/glmfit.html and https://www.mathworks.com/help/stats/g
lmval.html

N = ones(29,1);

%----------------------------------------Power Capacity Failure

FS_b = glmfit(xPGA, [Flow_slight.' , N] , 'binomial','link','logit');
Flow_slight_curve = glmval(FS_b,xPGA,'logit');

FM_b = glmfit(xPGA, [Flow_moderate.' , N] , 'binomial','link','logit');
Flow_moderate_curve = glmval(FM_b,xPGA,'logit');

FE_b = glmfit(xPGA, [Flow_extensive.' , N] , 'binomial','link','logit');
Flow_extensive_curve = glmval(FE_b,xPGA,'logit');

FC_b = glmfit(xPGA, [Flow_complete.' , N] , 'binomial','link','logit');
Flow_complete_curve = glmval(FC_b,xPGA,'logit');
%----------------------------------------Component Failure

HS_b = glmfit(xPGA, [Haz_slight.' , N] , 'binomial','link','logit');
Haz_slight_curve = glmval(HS_b,xPGA,'logit');

HM_b = glmfit(xPGA, [Haz_moderate.' , N] , 'binomial','link','logit');
Haz_moderate_curve = glmval(HM_b,xPGA,'logit');

HE_b = glmfit(xPGA, [Haz_extensive.' , N] , 'binomial','link','logit');
Haz_extensive_curve = glmval(HE_b,xPGA,'logit');

HC_b = glmfit(xPGA, [Haz_complete.' , N] , 'binomial','link','logit');
Haz_complete_curve = glmval(HC_b,xPGA,'logit');
```

- Plot results - Power Flow

```
figure('Name', [type , ' Seismic Failure States'])
grid on
hold on
xlim([0 1.4])
```

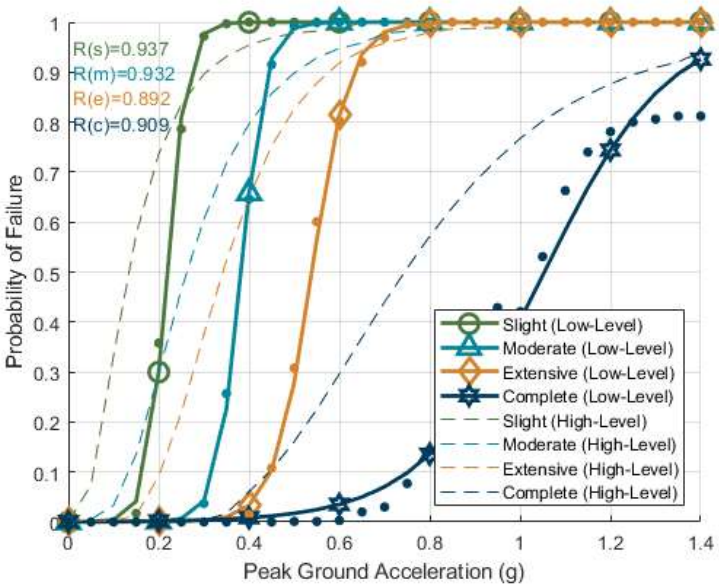
```
% Experimentally-determined logistic curves

plot(PGA_mat, Flow_slight_curve, '-o','MarkerSize',10, 'MarkerIndices', 1:4:length(PGA_mat),
'LineWidth',2, 'Color', 1/255*[74,119,60])
plot(PGA_mat, Flow_moderate_curve, '-^','MarkerSize',10, 'MarkerIndices', 1:4:length(PGA_mat)
, 'LineWidth',2, 'Color', 1/255*[0,133,155])
plot(PGA_mat, Flow_extensive_curve, '-d','MarkerSize',10, 'MarkerIndices', 1:4:length(PGA_mat
), 'LineWidth',2, 'Color', 1/255*[211,131,43])
plot(PGA_mat, Flow_complete_curve, '-h','MarkerSize',10, 'MarkerIndices', 1:4:length(PGA_mat)
, 'LineWidth',2, 'Color', 1/255*[0,59,92])

% Hazus Curves

plot(xPGA ,HSS_LV_S, 'Color', 1/255*[74,119,60], 'Linestyle', '--')
plot(xPGA ,HSS_LV_M, 'Color', 1/255*[0,133,155], 'Linestyle', '--')
plot(xPGA,HSS_LV_E, 'Color', 1/255*[211,131,43], 'Linestyle', '--')
plot(xPGA,HSS_LV_C, 'Color', 1/255*[0,59,92], 'Linestyle', '--')

xlabel('Peak Ground Acceleration (g)')
ylabel('Probability of Failure')

% Gives Correlation coefficient for each pair of curves

Corr_Sf = mean((HSS_LV_S-mean(HSS_LV_S)).*(Flow_slight-mean(Flow_slight)))/(std(HSS_LV_S,1)*s
td(Flow_slight,1));
Corr_Mf = mean((HSS_LV_M-mean(HSS_LV_M)).*(Flow_moderate-mean(Flow_moderate)))/(std(HSS_LV_M,
1)*std(Flow_moderate,1));
Corr_Ef = mean((HSS_LV_E-mean(HSS_LV_E)).*(Flow_extensive-mean(Flow_extensive)))/(std(HSS_LV_
E,1)*std(Flow_extensive,1));
Corr_Cf = mean((HSS_LV_C-mean(HSS_LV_C)).*(Flow_complete-mean(Flow_complete)))/(std(HSS_LV_C,
1)*std(Flow_complete,1));

text(0.01, .95 , ['R(s)=' , num2str(Corr_Sf)], 'Color',1/255*[74,119,60])
text(0.01, .9 , ['R(m)=' , num2str(Corr_Mf)], 'Color',1/255*[0,133,155])
text(0.01, .85 , ['R(e)=' , num2str(Corr_Ef)], 'Color',1/255*[211,131,43])
text(0.01, .8 , ['R(c)=' , num2str(Corr_Cf)], 'Color',1/255*[0,59,92])

% Raw experimental data

scatter(PGA_mat, Flow_slight, 25, 1/255*[74,119,60], 'filled' ,'HandleVisibility','off')
scatter(PGA_mat, Flow_moderate, 25,  1/255*[0,133,155], 'filled' ,'HandleVisibility','off')
scatter(PGA_mat, Flow_extensive, 25, 1/255*[211,131,43], 'filled' , 'HandleVisibility','off')
scatter(PGA_mat, Flow_complete, 25, 1/255*[0,59,92], 'filled' , 'HandleVisibility','off')

legend('Slight (Low-Level)','Moderate (Low-Level)', 'Extensive (Low-Level)', 'Complete (Low-L
evel)', ...
       'Slight (High-Level)','Moderate (High-Level)','Extensive (High-Level)','Complete (High
-Level)', 'Location', 'southeast')

% Saving Plot

feedstr = num2str(feed);

filename = [savename, '_Flow', feedstr];
saveas(gcf, [pwd, '/Plots/', filename], 'epsc')
```



Plot Results - Raw Failure

```
figure('Name', [type , ' Seismic Failure States (Hazus Methodology)'])
grid on
hold on
xlim([0 1.4])

% Experimentally-determined logistic curves

plot(PGA_mat, Haz_slight_curve, '-o','MarkerSize',10, 'MarkerIndices', 1:4:length(PGA_mat), '
LineWidth',2, 'Color', 1/255*[74,119,60])
plot(PGA_mat, Haz_moderate_curve, '-^','MarkerSize',10, 'MarkerIndices', 1:4:length(PGA_mat),
 'LineWidth',2, 'Color', 1/255*[0,133,155])
plot(PGA_mat, Haz_extensive_curve, '-d','MarkerSize',10, 'MarkerIndices', 1:4:length(PGA_mat)
, 'LineWidth',2, 'Color', 1/255*[211,131,43])
plot(PGA_mat, Haz_complete_curve, '-h','MarkerSize',10, 'MarkerIndices', 1:4:length(PGA_mat),
 'LineWidth',2, 'Color', 1/255*[0,59,92])

% Hazus Curves

plot(xPGA,HSS_LV_S, 'Color', 1/255*[74,119,60], 'Linestyle', '--')
plot(xPGA,HSS_LV_M, 'Color', 1/255*[0,133,155], 'Linestyle', '--')
plot(xPGA,HSS_LV_E, 'Color', 1/255*[211,131,43], 'Linestyle', '--')
plot(xPGA,HSS_LV_C, 'Color', 1/255*[0,59,92], 'Linestyle', '--')

xlabel('Peak Ground Acceleration (g)')
ylabel('Probability of Failure')

% Correlation Coefficients
```

```
Corr_Sh = round(mean((HSS_LV_S-mean(HSS_LV_S)).*(Haz_slight-mean(Haz_slight)))/(std(HSS_LV_S,
1)*std(Haz_slight,1)),3);
Corr_Mh = round(mean((HSS_LV_M-mean(HSS_LV_M)).*(Haz_moderate-mean(Haz_moderate)))/(std(HSS_L
V_M,1)*std(Haz_moderate,1)),3);
Corr_Eh = round(mean((HSS_LV_E-mean(HSS_LV_E)).*(Haz_extensive-mean(Haz_extensive)))/(std(HSS
_LV_E,1)*std(Haz_extensive,1)),3);
Corr_Ch = round(mean((HSS_LV_C-mean(HSS_LV_C)).*(Haz_complete-mean(Haz_complete)))/(std(HSS_L
V_C,1)*std(Haz_complete,1)),3);

text(0.01, .95 , ['R(s)=' , num2str(Corr_Sh)], 'Color',1/255*[74,119,60])
text(0.01, .9 , ['R(m)=' , num2str(Corr_Mh)], 'Color',1/255*[0,133,155])
text(0.01, .85 , ['R(e)=' , num2str(Corr_Eh)], 'Color',1/255*[211,131,43])
text(0.01, .8 , ['R(c)=' , num2str(Corr_Ch)], 'Color',1/255*[0,59,92])

scatter(PGA_mat, Haz_slight, 25, 1/255*[74,119,60], 'filled' ,'HandleVisibility','off')
scatter(PGA_mat, Haz_moderate, 25,  1/255*[0,133,155], 'filled' ,'HandleVisibility','off')
scatter(PGA_mat, Haz_extensive, 25, 1/255*[211,131,43], 'filled' , 'HandleVisibility','off')
scatter(PGA_mat, Haz_complete, 25, 1/255*[0,59,92], 'filled' , 'HandleVisibility','off')

legend('Slight (Low-Level)','Moderate (Low-Level)', 'Extensive (Low-Level)', 'Complete (Low-L
evel)', ...
       'Slight (High-Level)','Moderate (High-Level)','Extensive (High-Level)','Complete (High
-Level)', 'Location', 'southeast')




% Saving Plot


filename2 = [savename, '_Hazus', feedstr];
saveas(gcf, [pwd, '/Plots/', filename2], 'epsc')

% end
```



Published with MATLAB® R2018b

B.9 Line Failure Function

Contents


```
function [Fail] = sbs_check(PGA, v)
```

Get Component Data

```
Spec = 'a'; % Assuming response spectra 'A'

%     [Fail_SD_57, Fail_GCB_57, ...
%     Fail_SD_115, Fail_GCB_115, ...
%     Fail_SD_230, Fail_GCB_230] = Comp_Fail(PGA, Spec);

%Feeds PGA, and Ground Response Spectra, returns failure of component type
```

```
Fail = zeros(1,length(v));

for i=1:length(v)

    [Fail_SD_57, Fail_GCB_57, ...
    Fail_SD_115, Fail_GCB_115, ...
    Fail_SD_230, Fail_GCB_230] = Comp_Fail(PGA(i), Spec);

    vi = v(i);



% This is intended only to be slotted into existing code, so it will work
% exclusively as a function

if vi < 86.5 % Voltage closest to 57

    Fail_mat_1 = Fail_Chance(Fail_SD_57); % First SD

    Fail_mat_2 = Fail_Chance(Fail_GCB_57); % CB

    Fail_mat_3 = Fail_Chance(Fail_SD_57); % Second SD

elseif vi < 172.5 % Voltage closest to 172.5

    Fail_mat_1 = Fail_Chance(Fail_SD_115); % First SD

    Fail_mat_2 = Fail_Chance(Fail_GCB_115); % CB

    Fail_mat_3 = Fail_Chance(Fail_SD_115); % Second SD

else % Voltage closest to 230

    Fail_mat_1 = Fail_Chance(Fail_SD_230); % First SD

    Fail_mat_2 = Fail_Chance(Fail_GCB_230); % CB
```

```
    Fail_mat_3 = Fail_Chance(Fail_SD_230); % Second SD

end

if Fail_mat_1 == 1 || Fail_mat_2 == 1 || Fail_mat_3 == 1
    Fail(i) = 1;
else
    Fail(i) = 0;
end

end
```

```
Not enough input arguments.

Error in sbs_check (line 13)
Fail = zeros(1,length(v));
```

Published with MATLAB® R2018b

B.10 Transformer Failure Function

Contents


```
function [Fail] = xf_check(PGA, v)
```

Get Component Data

```
Spec = 'a'; % Assuming response spectra 'A'


%    [Fail_IT_57, ...
%    Fail_IT_115, ...
%    Fail_IT_230] = Comp_Fail(PGA, Spec);
%Feeds PGA, and Ground Response Spectra, returns failure of component type
```

```
Spec = 'a'; % Assuming response spectra 'A'

%     [Fail_IT_57, ...
%     Fail_IT_115, ...
%     Fail_IT_230] = Comp_Fail(PGA, Spec);

%Feeds PGA, and Ground Response Spectra, returns failure of component type
```

```
Fail = zeros(1,length(v));

for i=1:length(v)

    [Fail_IT_57, ...
    Fail_IT_115, ...
    Fail_IT_230] = Comp_Fail(PGA, Spec);

    vi = v(i);


% This is intended only to be slotted into existing code, so it will work
% exclusively as a function

if vi < 86.5 % Voltage closest to 57

    Fail(i) = Fail_Chance(Fail_IT_57); % Single Transformer

elseif vi < 172.5 % Voltage closest to 172.5

    Fail(i) = Fail_Chance(Fail_IT_115); % Single Transformer

else % Voltage closest to 230

    Fail(i) = Fail_Chance(Fail_IT_230); % Single Transformer
```

```
end



end
```

```
Not enough input arguments.

Error in xf_check (line 22)
Fail = zeros(1,length(v));
```

Published with MATLAB® R2018b

B.11 Spectra Response Test Function

Contents


```
%function Multi_sub_comp_fail(feed, gen_in)
```

Base code - William Stark

```
% Modified snippet of code to test substation at three different response
% spectra.

% No Hazus output, only looks at curves at the three spectra under both
% power flow and Hazus Methodologies
```

Initialization

```
clear all
clc
```

- Test Input -

```
% Note that, per the project guidelines, we will be using a combination of
% configurations based on number of feeders.

%If there is a generator, we
% will use DBDB regardless of feeder #

prompt = 'Enter number of feeders (2 or more)\n';
feed = input(prompt);

if feed < 2
    error('Please choose a workable number of feeders')
end

%If there is a generator, we
% will use DBDB

prompt = 'Is there an attached generator? (Y/N)\n';
gen_in = input(prompt, 's');
gen = lower(gen_in);
```

```
if ~ismember(gen, ['y' 'n'])
    error('Please choose either Y or N')
end


% For now, we assume Response Spectra 'A'
%
% prompt = 'What is the Response spectra? (A, B, or C) ';
% Spec = lower(input(prompt, 's'));
%
% if ~ismember(Spec, ['a' 'b' 'c'])
%     error('Invalid Response Spectra')
% end

Spec_mat = ['a','b','c'];
```

```
Error using input
Cannot call INPUT from EVALC.

Error in Spectra_SD_test (line 25)
feed = input(prompt);
```

Matrix Generation

```
PGA_mat = [0:0.05:2]; %Matrix Containing all possible PGA values (Length 41)

States = zeros(5,41); % matrix containing possible failure states: 0 (no failure) 0.333 (1 li
ne failed)* 0.667 (2 lines failed) 1 (All lines failed)
S_raw = zeros(5,41);
```

- Select Substation, Loop through all PGA values 0-2 ten times and build a matrix where each term has PGA and operational states

```
for sp = 1:3
```

```
    Spec = Spec_mat(sp);

S_count = 1;

run = zeros(1,1000);
raw = zeros(1,1000);



 for n = 1 : length(PGA_mat)

    PGA = PGA_mat(n);

    for k = 1:1000
```

```
        if gen == 'y' % Generator present, means Dual Bus Dual Breaker
            [Fail_state, F_raw] = model_sub_DBDB(feed, PGA, Spec);
            type = 'Dual Bus, Dual Breaker';
            savename = 'DBDB';
        elseif feed == 2 % 2-feeder substation, SBSB
            [Fail_state, F_raw] = model_sub_SBSB(PGA, Spec);
            type = 'Single Bus, Single Breaker';
            savename = 'SBSB';
        elseif feed < 5 % 3 or 4-feeder, Ring Bus
            [Fail_state, F_raw] = model_sub_RING(feed, PGA, Spec);
            type = 'Ring Bus';
            savename = 'RB';
        else % 5 or more feeders, Breaker-and-a-half
            [Fail_state, F_raw] = model_sub_BAH(feed, PGA, Spec);
            type = 'Breaker-and-a-Half';
            savename = 'BAH';
        end
            run(k) = Fail_state;
            raw(k) = F_raw;

            if k == 1000

                States(1,S_count) = sum(run==0)/1000; % No damage, does not need a curve
                S_raw(1,S_count) = sum(raw==0)/1000; % No damage, does not need a curve

                States(2,S_count) = (sum(run==0.05) + sum(run==0.4) + sum(run==0.7)+ sum(run==1))/1000; % Slight damage or greater
                S_raw(2,S_count) = (sum(raw==0.05) + sum(raw==0.4) + sum(raw==0.7)+ sum(raw==1))/1000;

                States(3,S_count) = (sum(run==0.4) + sum(run==0.7)+ sum(run==1))/1000; % Moderate damage or greater
                S_raw(3,S_count) = (sum(raw==0.4) + sum(raw==0.7)+ sum(raw==1))/1000;

                States(4,S_count) = (sum(run==0.7)+ sum(run==1))/1000; % Severe damage or greater
                S_raw(4,S_count) = (sum(raw==0.7)+ sum(raw==1))/1000;

                States(5,S_count) = sum(run==1)/1000; % Complete Damage, totally nonfunctional
                S_raw(5,S_count) = sum(raw==1)/1000;

                S_count = S_count + 1;
            end

    end
 end
```

Prepare to PLot

```
% Create power flow arrays
if sp == 1
    Flow_slight1 = States(2,:);
    Flow_moderate1 = States(3,:);
    Flow_extensive1 = States(4,:);
```

```
    Flow_complete1 = States(5,:);

    % Create Hazus Method Arrays

    Haz_slight1 = S_raw(2,:);
    Haz_moderate1 = S_raw(3,:);
    Haz_extensive1 = S_raw(4,:);
    Haz_complete1 = S_raw(5,:);
elseif sp == 2
    Flow_slight2 = States(2,:);
    Flow_moderate2 = States(3,:);
    Flow_extensive2 = States(4,:);
    Flow_complete2 = States(5,:);

    % Create Hazus Method Arrays

    Haz_slight2 = S_raw(2,:);
    Haz_moderate2 = S_raw(3,:);
    Haz_extensive2 = S_raw(4,:);
    Haz_complete2 = S_raw(5,:);
elseif sp == 3
    Flow_slight3 = States(2,:);
    Flow_moderate3 = States(3,:);
    Flow_extensive3 = States(4,:);
    Flow_complete3 = States(5,:);

    % Create Hazus Method Arrays

    Haz_slight3 = S_raw(2,:);
    Haz_moderate3 = S_raw(3,:);
    Haz_extensive3 = S_raw(4,:);
    Haz_complete3 = S_raw(5,:);
end
```

```
end
```

- Plot results - Power Flow

```
figure('Name', [type , ' Response Spectra Comparison'])
grid on
hold on

plot(PGA_mat, Flow_slight1,'LineWidth',2, 'Color', 1/255*[74,119,60])
plot(PGA_mat, Flow_moderate1,'LineWidth',2, 'Color', 1/255*[0,133,155])
plot(PGA_mat, Flow_extensive1,'LineWidth',2, 'Color', 1/255*[211,131,43])
plot(PGA_mat, Flow_complete1,'LineWidth',2, 'Color', 1/255*[0,59,92])

plot(PGA_mat, Flow_slight2,'LineWidth',2, 'Linestyle', '--', 'Color', 1/255*[74,119,60])
plot(PGA_mat, Flow_moderate2,'LineWidth',2, 'Linestyle', '--', 'Color', 1/255*[0,133,155])
plot(PGA_mat, Flow_extensive2,'LineWidth',2, 'Linestyle', '--', 'Color', 1/255*[211,131,43])
plot(PGA_mat, Flow_complete2,'LineWidth',2, 'Linestyle', '--', 'Color', 1/255*[0,59,92])

plot(PGA_mat, Flow_slight3,'LineWidth',2, 'Linestyle', ':', 'Color', 1/255*[74,119,60])
plot(PGA_mat, Flow_moderate3,'LineWidth',2, 'Linestyle', ':', 'Color', 1/255*[0,133,155])
```

```
plot(PGA_mat, Flow_extensive3,'LineWidth',2, 'Linestyle', ':', 'Color', 1/255*[211,131,43])
plot(PGA_mat, Flow_complete3,'LineWidth',2, 'Linestyle', ':', 'Color', 1/255*[0,59,92])
xlim([0 1.4])




xlabel('Peak Ground Acceleration (g)')
ylabel('Probability of Failure')

legend('Slight (A)','Moderate (A)', 'Extensive (A)', 'Complete (A)', ...
       'Slight (B)','Moderate (B)', 'Extensive (B)','Complete (B)', ...
       'Slight (C)','Moderate (C)', 'Extensive (C)','Complete (C)', 'Location', 'southeast')


% Saving Plot

feedstr = num2str(feed);

filename = [savename, '_Flow_Spec', feedstr];
saveas(gcf, filename, 'epsc')
```

Plot Results - Raw Failure

```
figure('Name', [type , ' Response Spectra Comparison'])
grid on
hold on

plot(PGA_mat, Haz_slight1,'LineWidth',2, 'Color', 1/255*[74,119,60])
plot(PGA_mat, Haz_moderate1,'LineWidth',2, 'Color', 1/255*[0,133,155])
plot(PGA_mat, Haz_extensive1,'LineWidth',2, 'Color', 1/255*[211,131,43])
plot(PGA_mat, Haz_complete1,'LineWidth',2, 'Color', 1/255*[0,59,92])

plot(PGA_mat, Haz_slight2,'LineWidth',2, 'Linestyle', '--', 'Color', 1/255*[74,119,60])
plot(PGA_mat, Haz_moderate2,'LineWidth',2, 'Linestyle', '--', 'Color', 1/255*[0,133,155])
plot(PGA_mat, Haz_extensive2,'LineWidth',2, 'Linestyle', '--', 'Color', 1/255*[211,131,43])
plot(PGA_mat, Haz_complete2,'LineWidth',2, 'Linestyle', '--', 'Color', 1/255*[0,59,92])

plot(PGA_mat, Haz_slight3,'LineWidth',2, 'Linestyle', ':', 'Color', 1/255*[74,119,60])
plot(PGA_mat, Haz_moderate3,'LineWidth',2, 'Linestyle', ':', 'Color', 1/255*[0,133,155])
plot(PGA_mat, Haz_extensive3,'LineWidth',2, 'Linestyle', ':', 'Color', 1/255*[211,131,43])
plot(PGA_mat, Haz_complete3,'LineWidth',2, 'Linestyle', ':', 'Color', 1/255*[0,59,92])
xlim([0 1.4])




xlabel('Peak Ground Acceleration (g)')
ylabel('Probability of Failure')

legend('Slight (A)','Moderate (A)', 'Extensive (A)', 'Complete (A)', ...
       'Slight (B)','Moderate (B)', 'Extensive (B)','Complete (B)', ...
       'Slight (C)','Moderate (C)', 'Extensive (C)','Complete (C)', 'Location', 'southeast')
```

```
% Saving Plot


filename2 = [savename, '_Hazus_Spec', feedstr];
saveas(gcf, filename2, 'epsc')

%end
```

Published with MATLAB® R2018b