Power Grid Asset Restoration Sequencing and Importance Metric for Improved Resilience

Author: Parveer Banwait

Student Number: 1007016499

Supervised by: Professor Zeb Tate

April 2025

ESC499: Thesis

Final Report

Abstract

This is the abstract of the paper. It summarizes the main objectives, methods, results, and conclusions of your work. The abstract should be concise and typically between 150-250 words.

Acknowledgements

I would like to express my gratitude to my advisor, Dr. XYZ, for their guidance and support. Special thanks to my colleagues and family for their encouragement throughout this journey.

Contents

List of Figures

1 Introduction

1.1 Project Introduction

Climate change is expected to increase the number of extreme weather events [1], increasing the risk of infrastructure damage. If power grid infrastructure fails, extended power outages may result while they are being repaired, disrupting services which have become critical to society. Increasing the grid's ability to react, recover from, and mitigate the effect of these high impact low probability events, which comprise the idea of grid resilience, has thus become an increasingly important field of research. A grid's resilience can be improved in multiple ways, including having measures in place to repair grid assets in order to restore power optimally, as well as strengthening critical grid infrastructure [2].

The field of grid resilience is interdisciplinary. Weather models are required for determining components or sets of components that are vulnerable to extreme weather conditions. Structural analysis is then required to determine whether a component hit by extreme weather is likely to be damaged. Finally, network models are required to determine the extent of the subsequent power outage, and how it pervades after network components are restored [2].

This thesis will present two separate measures which will improve the resilience of power grids. First, i various methods for sequencing the repair of grid assets such as Keepn will be introduced, which balance computation time and efficacy. This attempts to decrease the extent of the blackouts caused by extreme weather events in the short term. For long-term resilience, Monte Carlo simulations are conducted using generated semi-realistic tornado events and structural models to obtain various scenarios of damaged grid elements. A network model will then assess the impact of these scenarios, using the results to calculate an 'importance' value for each grid element.

1.2 Literature Review

One of the main metrics for quantifying the resilience of a power grid in response to an extreme weather event is the resilience triangle discussed in [3] and presented in Figure 1. The triangle demonstrates the states of the grid in with respect to extreme weather. After an "attack" on the grid occurs, oftentimes by an extreme weather event, the grid

enters a degraded state, with some loss in functionality. Restoration then begins, with the grid functionality slowly increasing over time as grid elements are repaired and returned to service. This continues until the grid returns to full service. The area of this triangle is one of the main methods used for quantifying grid resilience and represents the total impact of the event [3].

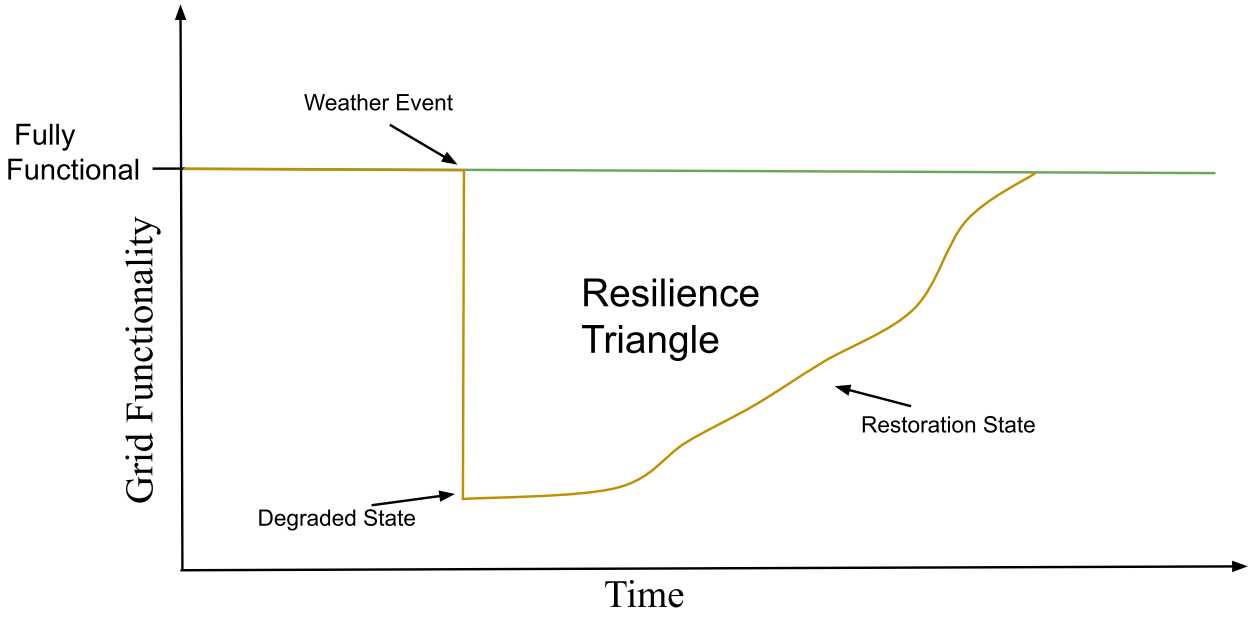

Figure 1: A visual representation of the resilience triangle

It is apparent that finding the optimal repair sequence to minimize the area of this resilience triangle, and thus the total impact of the event, is a critical field of study. However, it is extremely important that determining this sequence, helps instead of hindering the restoration of the grid [4], and thus it is essential that it does not take a large amount of time. For larger outages, with n grid elements removed, evaluating all $n!$ possible restoration sequences is unfeasible. The majority of the literature attempts to solve this problem for operators in the control room [5] assuming all of the grid components have already been repaired, such as in [6][7]. However, there have been various methods proposed which attempt the "Restoration Ordering Problem" as coined in [8]. [9] proposes injecting precedence constraints while using in a large neighbourhood search in a mixed integer program, including vehicle routing for repair units. [5] takes the repair crews further, adding in repair crew capacities and trips to the depot in their model. They also include a novel weather dependent repair time for substations as well as ant inspired pheromone trails, speeding up their MIP solution. However, [10] assumes that crew travel times are small compared to actual repair times after discussion with an industry expert. They use an island by island approach in

their solution, prioritizing islands using a soft precedence graph to solve a linear program. However, these MIP based solutions can be difficult to implement and take an uncertain amount of time to complete.

In [11], the grid is simplified to one source and one sink, without losing generality, and treats it as a graph problem, determining which arc to restore next by dividing the amount of residual capacity arc by the time it takes to restore that given arc. This approach takes its inspiration from Smith's weighted processing time rule [12]. Unfortunately, this solution does not appear to schedule a repair sequence on an asset by asset basis.

A randomized adaptive decomposition (RAD) method of solving the problem is proposed by [8], where they start with an initial sequence based on basic characteristics of the grid asset (dubbed a utilization heuristic), partition it randomly, and solve the problem for each subsequence individually. They then partition again and repeat, before a stopping condition is reached. They guarantee that their sequence does not worsen after every iteration.

Another solution for efficiently decreasing the impact of severe weather events is by selectively hardening critical grid infrastructure elements structurally to reduce the resilience triangle's for potential future events. In [13] this was attempted by simulating earthquakes in Portland and using a component model to determine when grid assets are damaged. They then search for the grid elements which tend to be damaged when a lot of load is lost. Mu et al. [14] uses weather and component models to find the probability of failure for transmission lines. They then use that along with the amount of load connected with each line to determine the importance of each line. Braik et al. [15] generate tornado events, assuming an EF1 or EF2 magnitude, a set of starting locations and a 45 degree northeastward angle for all tornadoes. They then find the amount of downstream load, assuming a radial network, to determine which lines should be hardened. A machine learning model is used by [16] to evaluate the effectiveness of resilience improvement strategies such as vegetation management, undergrounding, and structural hardening, in minimizing future outages by utilizing historical outage data for training. These solutions in literature do not utilize the true metric for grid resilience, the area of the resilience triangle, and do not use general weather events which will fully represent the space of possible future hazards. This paper will present a framework for determining the important assets to improve within an entire power network, in responding to a large range of potential tornado events.

2 Weather Event Creation

2.1 Asset Importance Metric

In order to prioritize the hardening of specific assets, a metric is needed to identify the assets of highest importance. In this paper, the grid assets considered are grid buses and transmission lines. This "importance" metric must be a function of both the assets likelihood of getting damaged by a tornado (or any other extreme weather event), as well as the effect that the asset's damage on an ensuing outage. There are other key criteria this metric must meet. For instance, it should identify and account for cases where an asset and nearby ones could trigger large outages if they fail simultaneously, as well as the likelihood of such failures occurring together. Additionally, larger outages, caused by more devastating events, are more critical to prevent in order to enhance future resilience. Therefore, these events must be given greater weight. Given the above criteria, the metric chosen is the following. For each extreme event, and for each asset damaged by that event, the difference in the event's total impact is found in one case where the asset was damaged, and another assuming the asset was not (as if it had been perfectly hardened). This difference is then summed over each extreme weather event. The metric is presented in equation (1) below.

$$I_i = \sum_{j=1}^{n} A_j - A_j^i \tag{1}$$

Here, I_i is the importance of asset i, n is the number of simulated extreme weather events, A_j is the extent of outage j, and A_j^i is the extent of outage j assuming asset i was never damaged. In this paper, the extent of the outage is measured as the area of the *Reslience Triangle* in Figure 1.

The metric is calculated for each asset by completing a Monte Carlo simulation to simulate outages. This entails creating semi-realistic severe tornado events, before determining whether or not grid assets within the tornado's extent are damaged. After all the assets damaged by the outage are removed, the extent of the event is calculated by finding the area of the resilience triangle. This will be completed via grid network analysis as well as asset repair sequencing, which will be discussed in the following chapter. Then, for each damaged asset, the extent of the outage caused by the tornado will be calculated as if it had never been damaged.

This metric was chosen as it is quite simple and yet achieves all of the necessary criteria. It accounts for the likelihood of the asset being hit by an extreme weather event via it having more nonzero elements in the summation of equation (1) if it is damaged more during the Monte Carlo simulation. If it is very important for the grids operation, the cost difference in each individual event it is damaged in is likely to be high. As well, if it and another close by asset being damaged simultaneously cause a large outage, this metric will give both assets a large importance if weather events are likely to damage both at the same time. Lastly, events which cause larger outages are more heavily weighted since a simple cost difference is taken instead of averaging the importance over many events.

2.2 Tornado Model

In creating a tornado model, the TorMC model from [17] was followed with a few modifications. This model generates a tornado starting location either randomly or from a heatmap, within a pre-determined study region. In this paper, a heatmap provided from NOAA [18] for EF2 or higher tornadoes (Figure 2) was used to determine the starting location . Anthropogenic climate change is likely to change this heatmap as the climate system changes. However, the exact effect of this change on the locations of extreme weather is still uncertain [1]. After discussions with an atmospheric scientist who specializes in extreme weather events it was concluded that attempting to determine the change at a grid asset level would be impractical. Consequently, historical data was deemed the most reliable predictor for future weather events. Thus, for other tornado metrics NOAA's dataset of past tornadoes was used. The NOAA heatmap is structured by having probability of tornado value for various values of longitude and latitude. However, the heatmap is only 45x70, which is not a high enough resolution when simulating weather events at a substation level. To resolve this, SciPy's interpolate.griddata function was used to refine it to a discrete PDF with a 4000x4000 grid. A 2-D cubic spline interpolation method was specified here. The PDF is then cut to match the study region.

When plotting using heatmaps with longitude and latitude, results end up being pole skewed as longitude lines draw closer the further poleward one goes [17]. Hence, the probability values were normalized by multiplying by the cosine of the latitude.

Next, a cumulative probability distribution is calculated for the longitudes. A random value between zero and one is generated, and the CDF is used to map that to a longitude, interpolating between the two nearest grid points in the discrete 4000x4000 PDF. Then, using the closest longitude grid point in the PDF, the conditional CDF is calculated for latitudes given that longitude. Another random value is generated, and a latitude is determined.

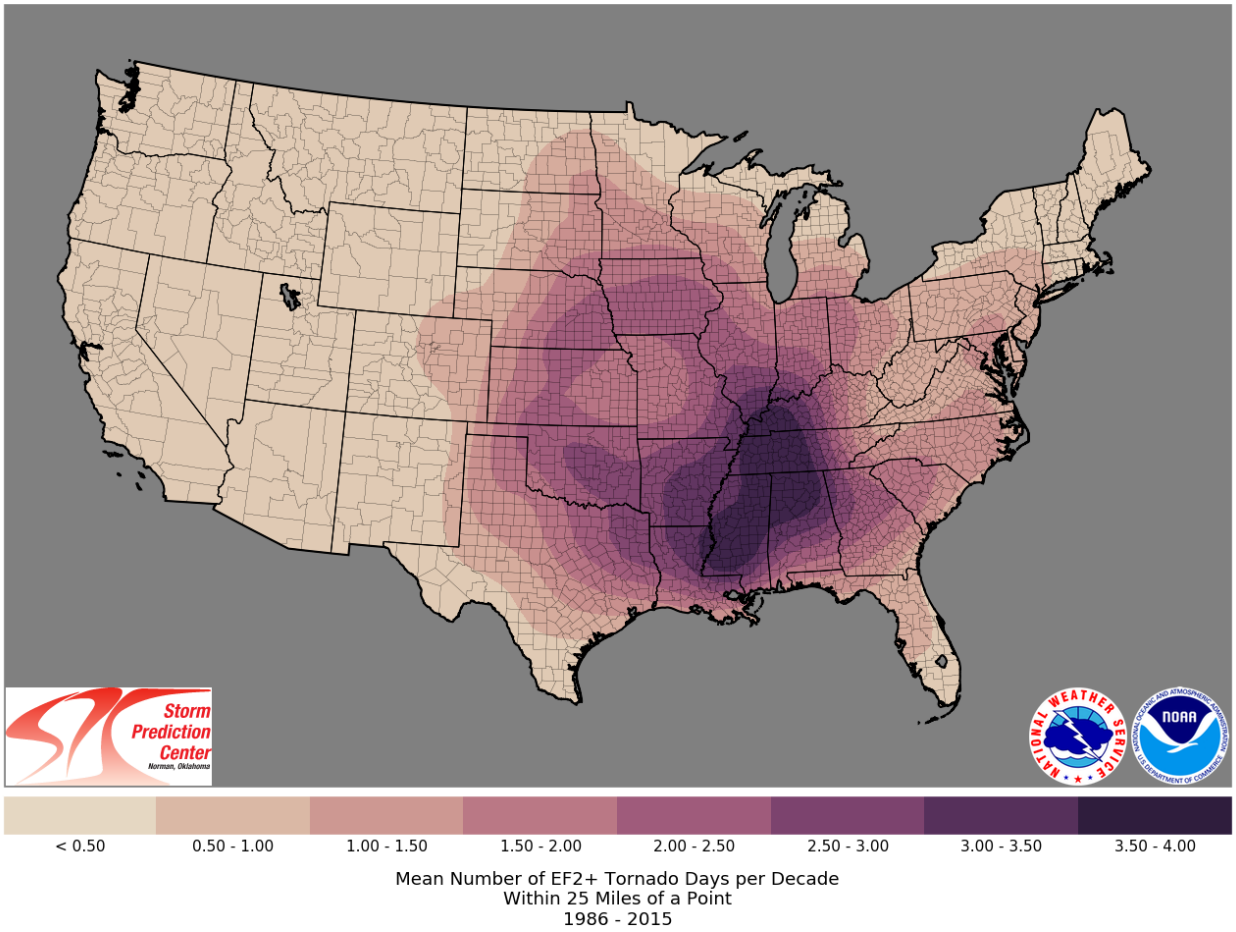

Figure 2: Heatmap of past tornadoes from NOAA's website [18].

The results of this process can be seen via the heatmap on the top left of Figure 3 below, with he study region being the entirety of the continental United States. This interpolated heatmap closely matches the heatmap provided by NOAA, with the distinct features such as the high probability tornado alley near Mississippi and Tennessee remaining, as well as the low probability region in eastern Kansas and western Missouri. The heatmap created by generated points on the top right of Figure 3 was created using SciPy's gaussian_kde function using a 200x200 grid, hence why it is blurrier. The gaussian_kde function uses kernel density estimation to estimate the PDF of a random variable [19]. The bottom figure shows that the 100000 generated points accurately reflect the 4000x4000 heatmap, with a large divergence from what is expected only appearing in locations with few tornadoes.

Next, unlike [17], given a tornado starting location, its magnitude is determined by sampling from the magnitudes of the 50 nearest EF2-or-higher tornadoes to that starting location. This approach was adopted because the heatmaps reveal significant variations in tornado magnitude across different locations. This method avoids placing a high magnitude

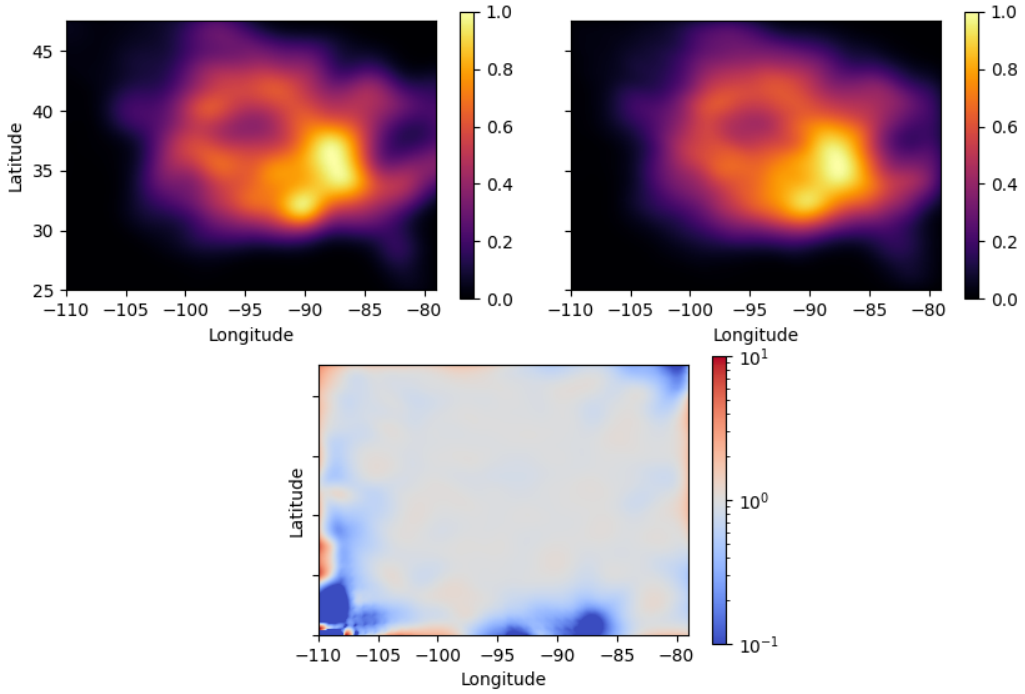

Figure 3: Top Left: Heatmap generated after interpolating the NOAA heatmap data. Top Right: Heatmap generated by the generated touchdown points from the Monte Carlo simulation. Bottom: Heatmap created by dividing the top left heatmap by the top right heatmap.

tornado in a region with no historical precedent of such events preventing the overestimation of grid damages in certain areas. The 50 nearest tornadoes were selected instead of all past tornadoes within a fixed radius to ensure a sufficiently large sample of magnitudes, even in areas with sparse tornado history.

Next, to find the angle (or "azimuth" in [17]) of the tornado, a random angle is selected from all previous tornadoes within the study region of the selected magnitude. A tornado length is then chosen in the same method. Lastly, as in [17], a Weibull distribution of tornado widths within the study region is fit for each possible magnitude. Then, a tornado width is sampled from the distribution corresponding to the randomly selected magnitude. The Weibull distribution is used here as the tornado width data from NOAA has a large number of data points at the exact same value, likely due to approximations used when measuring the width. The Weibull distribution allows for a more varied range of width values to be achieved, while fitting the right skewed width data.

To test the Weibull distributions, Q-Q plot's was created to plot the theoretical quantiles calculated from the Weibull distribution against the observed quantiles observed from the

data. The quantiles are calculated using filliben's estimate which finds the expected quantile of a given observed data point given it's rank in the ordered data set. These Q-Q plots are shown in Figure 4.

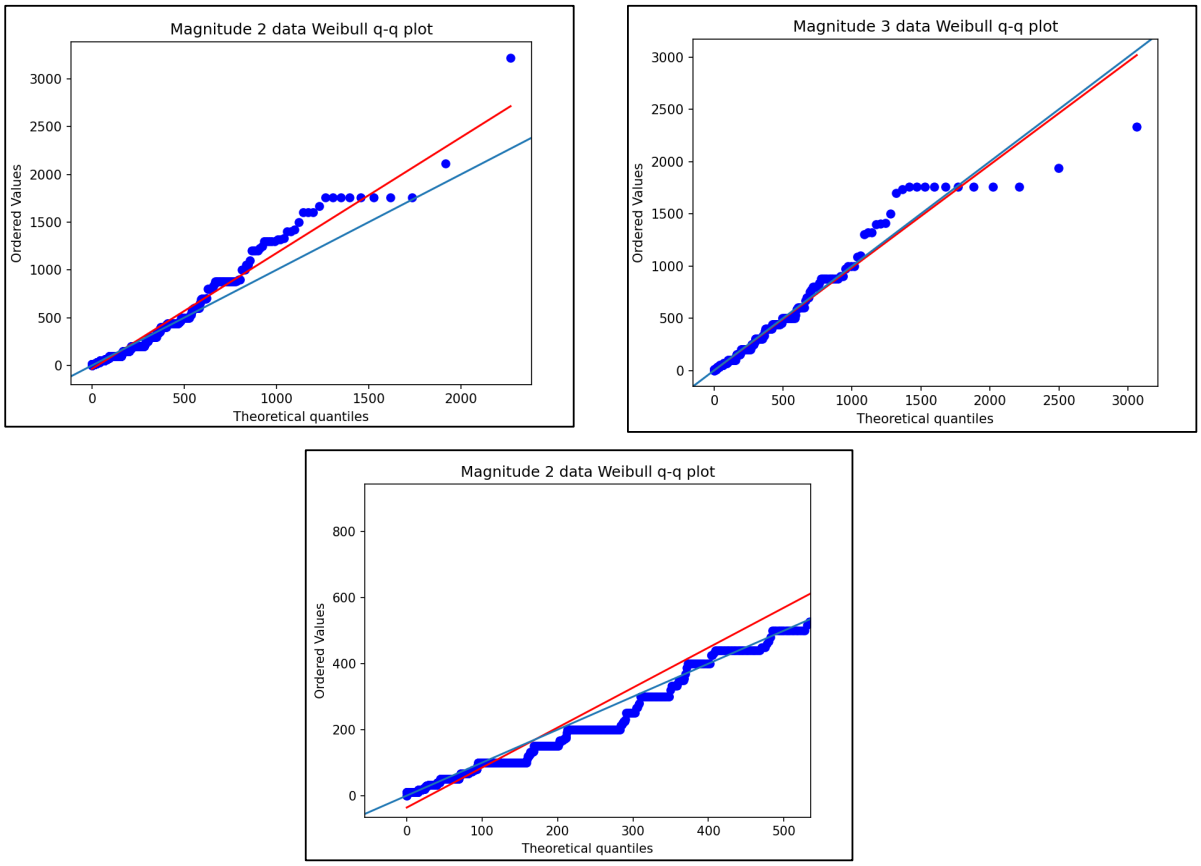

Figure 4: Top Left: Weibull fitted to magnitude 2 tornado width data. Top Right: Weibull fitted to magnitude 3 tornado width data. Bottom: Top left chart zoomed into smaller quantiles. All data is in yards.

In the figure, the red line represents a linear fit of the data, while the blue line corresponds to $x = y$, which the data would follow if the Weibull distribution perfectly fit the data. In these plots, it is clear to see that many of the observed data values are repeated, such as unlike the more continuous distribution one might expect with real world weather, justifying the use of the Weibull distribution here. For the magnitude 3 tornado data, the fit is quite strong justifying that the Weibull is a great choice. For magnitude 2 tornadoes, this is less true, as the Weibull distribution overestimates observed tornado widths at lower quantiles and underestimates them at higher quantiles. However, because the vast majority of the data falls within these lower width values, where the fit is quite strong, the Weibull distribution was deemed appropriate for this use case. This also explains why the fit worsens as width

increases, as the more prevalent lower-width data exerts a stronger influence on the fit.

Using the selected angle, length, width and the generated starting location, a tornado "box" is formed representing the spatial extent of the tornado path. The substations (and the buses that encompass them) which are within the tornadoes path are found. Additionally, all transmission lines which pass through or are completely within the borders of the tornadoes path are found, and their length within the storm is noted.

2.3 Structural Analysis

Fragility curves have been a key tool in resilience studies to describe the probability of failure of infrastructure in response to extreme weather [20]. These curves describe output the probability of failure of an item given an input on a wide domain of potential loads which could be applied to it. This load could be peak ground acceleration for earthquakes [21], wind speed for wind based events [20], or ice accretion for ice storms [22]. As such, fragility curves have been widely used to assess the vulnerability of power system infrastructure [2], with numerous studies dedicated to their development [20].

In this paper, testing the full space of outcomes of outages due to the fragility function for each tornado generated was deemed to be of a lower priority than having a large number of tornadoes generated. For larger study regions, a lower number of tornadoes would cause the heatmap to be poorly explored leading to some system assets being over/under sampled. Thus, the fragility functions were used as thresholds instead of probabilities in this study.

For transmission lines, fragility curves from [23] were used, shown in 5. They generated a large distribution of fragility curves using a machine learning model. Taking the middle curve as an average, there is a 50% chance of failure at a wind load of around 55 m/s. Using the correspondence provided from the American National Weather Service [24], this value corresponds to a EF2 tornado. Hence, as a threshold, transmission lines within the track of a EF2 or higher tornado will be removed. For substations/buses, the complexity of the station causes there to have been little research done on their fragility curves. However, these stations contain smaller towers similar to transmission towers. Hence those towers were used as a proxy. From fragility curves in [25], for an unhardened transmission tower, there is a 50% chance of failure at wind loads of around 78 m/s. This value corresponds to an EF4 tornado. However, this value is based on a purely wind based load. As discussed in [26], substation failures tend to be caused by flying debris caused by the tornado. Hence, the threshold for grid substation damage was lowered to an EF3 tornado or higher.

Pure wind (non-tornado) events were also considered in this study. However, even the highest recorded wind gusts, such as the 35 m/s maximum observed in Dallas–Fort Worth

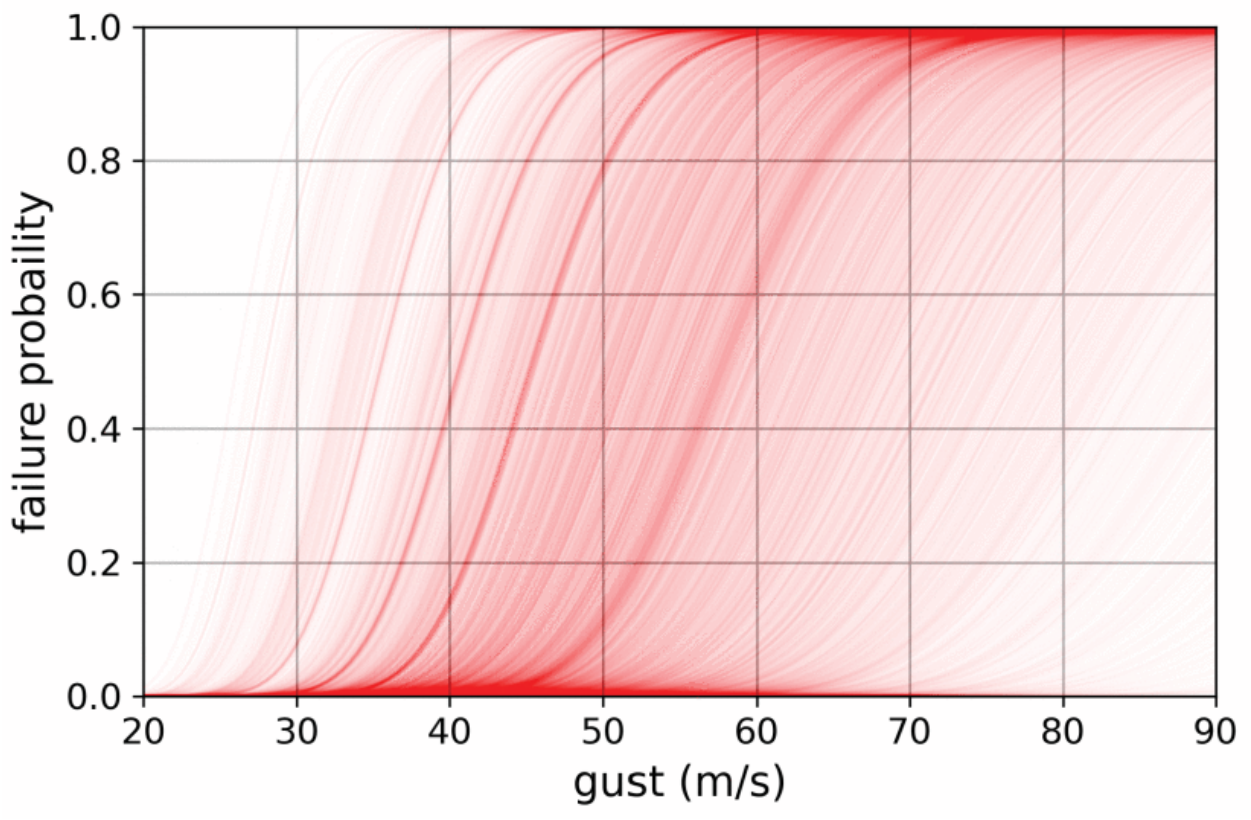

Figure 5: Transmission line fragility curves generated by [23]

over the past 20 years [24], remain well below the 55 m/s threshold associated with a 50% failure probability in the fragility curve discussed above. Consequently, wind loads were excluded from the analysis. However, climate change may alter this dynamic, as, for example, some RCP8.5 climate simulations project 50-year return level wind speeds to increase by 8–12% in northern Ontario. Further research is needed to assess how these changes might affect power grid infrastructure.

Ice accretion loads were also considered for this paper. For transmission lines, [27] presents a 2 axis fragility curve giving probability of failure with respect to wind speed and ice accretion. For transmission towers, [28] gives failure probabilities with respect to the amount of ice accretion assuming a concurrent wind load of 23 m/s. With these curves, time-series weather data which accounts for extreme values could be used for ice accretion events. Specifically, the Chaine model, which calculates ice thickness on horizontal wires as a function of depth of precipitation, wind speed, drop size and precipitation rate [22] is a viable method for modeling ice events. Ice event creation was not conducted in this thesis. However, the asset removals from these events could be easily sent to this paper's network model in future studies.

The time required to repair each grid asset is also important when calculating the area of the resilience triangle. It is difficult to determine this required time, with factors such as the situational awareness of grid operators, coordination of information sharing and the specific asset itself [2] influencing it. In this paper, for grid buses this value is assumed to be proportional to the number of transmission lines connected to it, as that scales with the size of the bus and is readily available in the network model. For transmission lines, the repair time is assumed to be proportional to the length of the line within the storm, as the longer the damaged part of the line, the longer it would likely take to repair. For this paper, it is assumed that a line which is within the storm's extent for one kilometer takes the same amount of time to repair as a grid bus connected to just one line. This is just a temporary representative value and should be refined by using utility repair records for future studies.

2.4 Synthetic Grids

To run simulations in this thesis, Texas A&M's electric grid datasets [29] were used. In particular, the 37 bus Hawaii synthetic case was used for testing bus repair sequences and the 6717 bus Texas synthetic case was used for both testing repair sequences and when testing implementation of the grid asset importance framework. Images of the two power grids are shown in Figure 6 below. These synthetic grids do not model the actual grid but do place grid elements such as generators and substations in realistic locations. This realistic model makes them ideal for this work where the geography of the region also impacts factors such as tornado placement.

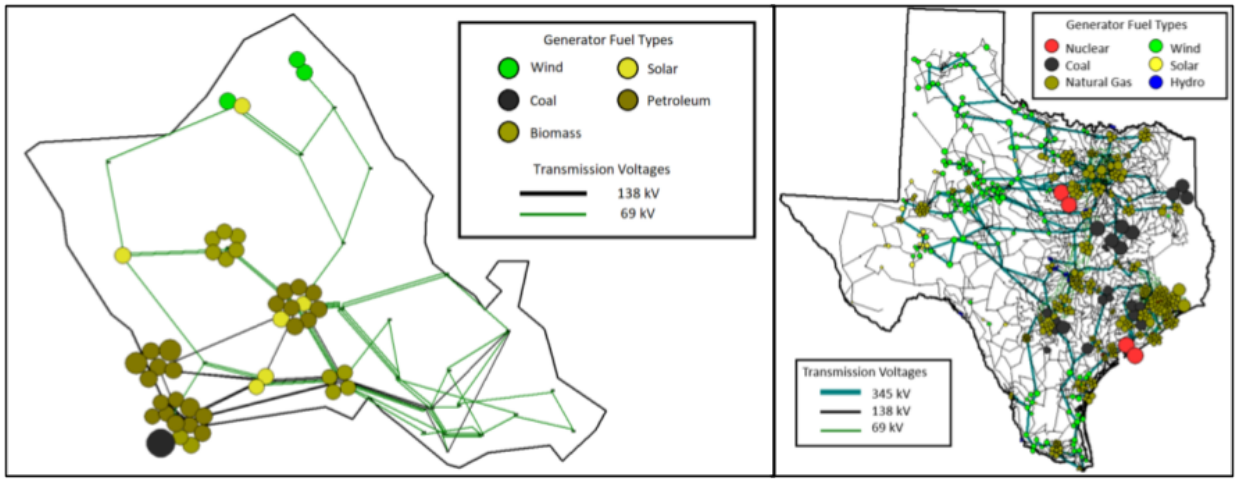

Figure 6: Texas A&M's Hawaii (left) and Texas (right) mock power grids [30]

3 Outage Repair Sequencing Method

3.1 Optimal Power Flow

Network analysis entails running an optimal power flow (OPF) simulation to determine the manner in which a power system must be dispatched to provide the optimal value for a given variable. Generally, this variable is monetary cost as networks attempt to economically dispatch the available generators. However, after extreme weather events, with potential large scale outages, it becomes critical that the system is dispatched in a manner which maximizes the amount of load delivered. Put the other way, the load shed of the grid must be minimized, defined in equation (2).

$$LS = \sum_{k \in B} P_{D,k} - P_{Del,k} \tag{2}$$

In equation (2), B is the set of buses, $P_{D,k}$ is the power demand at $busk$, and $P_{Del,k}$ is the power delivered to bus k, which is calculated by the OPF.

When solving the OPF problem, the AC power flow (AC-PF) equations are generally used to obtain exact solutions. At a given grid bus, these equations are [31]:

$$P_k = V_k \sum_{n \in \mathcal{N}_k} V_n \left[G_{kn} \cos(\delta_k - \delta_n) + B_{kn} \sin(\delta_k - \delta_n) \right] \tag{3}$$

$$Q_k = V_k \sum_{n \in \mathcal{N}_k} V_n \left[G_{kn} \sin(\delta_k - \delta_n) + B_{kn} \cos(\delta_k - \delta_n) \right] \tag{4}$$

In these equations, $\mathcal{N}_k$ is the set of buses directly connected to bus k, G_{kn} is the conductance of the line between buses k and n and B_{kn} is the admittance. P_k represents the net real power injection at bus k, which is the difference between generated and delivered power. If no generator or load is present, $P_k = 0$.

Unfortunately, the non-linearity of the AC-PF equations causes them to be extremely computationally expensive to solve. Since this paper involves running a large number of optimal power flow simulations in order to fully explore the space of possible tornadoes, this paper uses the DC optimal power flow (DC-OPF) formulation when running it's network analysis. DC-OPF involves using the DC power flow (DC-PF) equations which linearize the

AC-PF formulas. This linearization ignores the reactive component of the power flow and works under the following conditions [31]:

1. The voltage at each bus is close to one per unit. This allows the V_k and V_n terms in equation (3) to be set to one.

2. The transmission line resistances are low relative to the reactance. This allows for the neglect of the G_{kn} term in (3).

3. The phase angles of buses directly connected by a transmission line are close. The small angle approximation $\sin(\delta_k - \delta_n) = \delta_k - \delta_n$ can then be used.

After applying these assumptions, equation (3) becomes the following:

$$P_k = \sum_{n \in \mathcal{N}_k} \frac{\delta_k - \delta_n}{X_{nk}} \tag{5}$$

Where X_nk is the reactance of the line concecting buses n and k. This DC-PF formulation allows for the power flow to be solved in one iteration, vastly improving the time required to compute the power flow. It should be noted however, that these assumptions are not always very realistic. For instance, the reactance to resistance ratio being large may not always be true. Additionally, the per unit bus voltages are likely somewhat variable [31]. Nonetheless, the DC-PF remains a popular tool in power systems studies [31] and is used in this paper given the high number of simulations run.

Using the DC-PF formulas, an optimal power flow formulation can be formed to minimize the amount of load shed.

$$\min \quad \sum_{k \in B} LS_k \tag{6}$$

$$\text{s.t.} \quad P_{D,k} - LS_k - P_{G,k} + \sum_{n \in \mathcal{N}_k} \frac{\delta_k - \delta_n}{X_{nk}} = 0 \quad \forall k \in B \tag{7}$$

$$\frac{|\delta_{l^+} - \delta_{l^-}|}{X_l} \leq P_l^{max} \quad \forall l \in \mathcal{L}, \tag{8}$$

$$P_{G,k}^{min} \leq P_{G,k} \leq P_{G,k}^{max} \quad \forall k \in B \tag{9}$$

$$\delta_s = 0 \tag{10}$$

In this formulation, LS_k and $P_{G,k}$ are the load shed and generation at bus k respectively. $\mathcal{L}$ is the set of transmission lines, with δ_{l^+} and δ_{l^-} being the phase at the from and to ends of the line. P_l^{max} is the power flow limit through line l. $P_{G,k}^{min}$ and $P_{G,k}^{max}$ are the generator limits for the generator at bus k. If bus k is not connected to a generator, both of these are

zero. δ_s is the phase of the *slack bus*, which serves as the angle reference (hence $\delta_s = 0$) and provides for the ability to balance the power flow throughout the system.

In this paper, MATPOWER [32]., a MATLAB library for power system simulations, was used to solve this optimization problem. MATPOWER uses an interior point solver to conduct the DC-OPF [32]. The Hawaii and Texas synthetic power grid cases both have casefiles which are importable into MATPOWER.

3.2 Grid Asset Removal Logic

To assess the impact of outages given a set of damaged grid assets, a mechanism is required to systematically remove and return these assets within the system model. This section outlines the methodology for implementing this process.

Given a list of damaged transmission lines and buses, all items in the list were removed from the case file and stored. Additionally, transmission lines connected to damaged buses at either end would no longer be able to carry power and were stored separately. Since they did not require repairs, they could be reinstated as soon as whichever damaged bus (or buses) that was connected to it was repaired and returned to service.

Once lines and buses are removed from the system, there is the possibility of *islands* forming, or buses in the system which are disconnected from the main grid. MATPOWER cannot handle these cases as it expects a fully connected grid. To resolve this, a "test_islanding" function was formed to detect all formed islands, and assign the grid elements. Pseudocode for that function can be found in Appendix A. Afterwards, separate cases are made for each island using its grid elements. If an island does not have a generator, a generator with a power output of zero is added to its case, connected to an arbitrary bus. That bus is also set as the slack bus. If an island does have a generator, but no slack bus, an arbitrary bus with a generator is designated as that islands slack bus.

This grid asset removal formulation can lead to two main cases in which the DCOPF can be unsolvable. For one, if the total power demand in an island $\mathcal{I}_i$ is lower than the combined minimum generation of all its generators (i.e equation (11) does not hold), the excess power cannot be used, resulting in no feasible solution. This is solved by turning off an arbitrary generator on that island (i.e setting its minimum and maximum power generation to zero) until equation (11) holds for that island.

$$\sum_{k \in \mathcal{I}_i} P_{G,k}^{min} \leq \sum_{k \in \mathcal{I}_i} P_{D,k} \tag{11}$$

Another possible failure is for this formulation is as follows. It is possible for the minimum

generation of a given bus to be unable to flow through the network due to line power flow constraints, as seen in figure 7. This was resolved by adding a decision variable (*PMINreduction*) to the OPF which is akin to having a generator which produces negative load next to each existing generator. This requires adding the variable on the left hand side of Equation (7), and settings its maximum to the maximum generation of the generator as well as its minimum to zero. This decision variable is assigned an extremely high cost so that it is only non-zero if that doing so is absolutely necessary to obtain a feasible solution. If any value in *PMINreduction* is nonzero after running an OPF, the corresponding generators are turned off and the OPF is rerun. This continues until all of *PMINreduction* is zero.

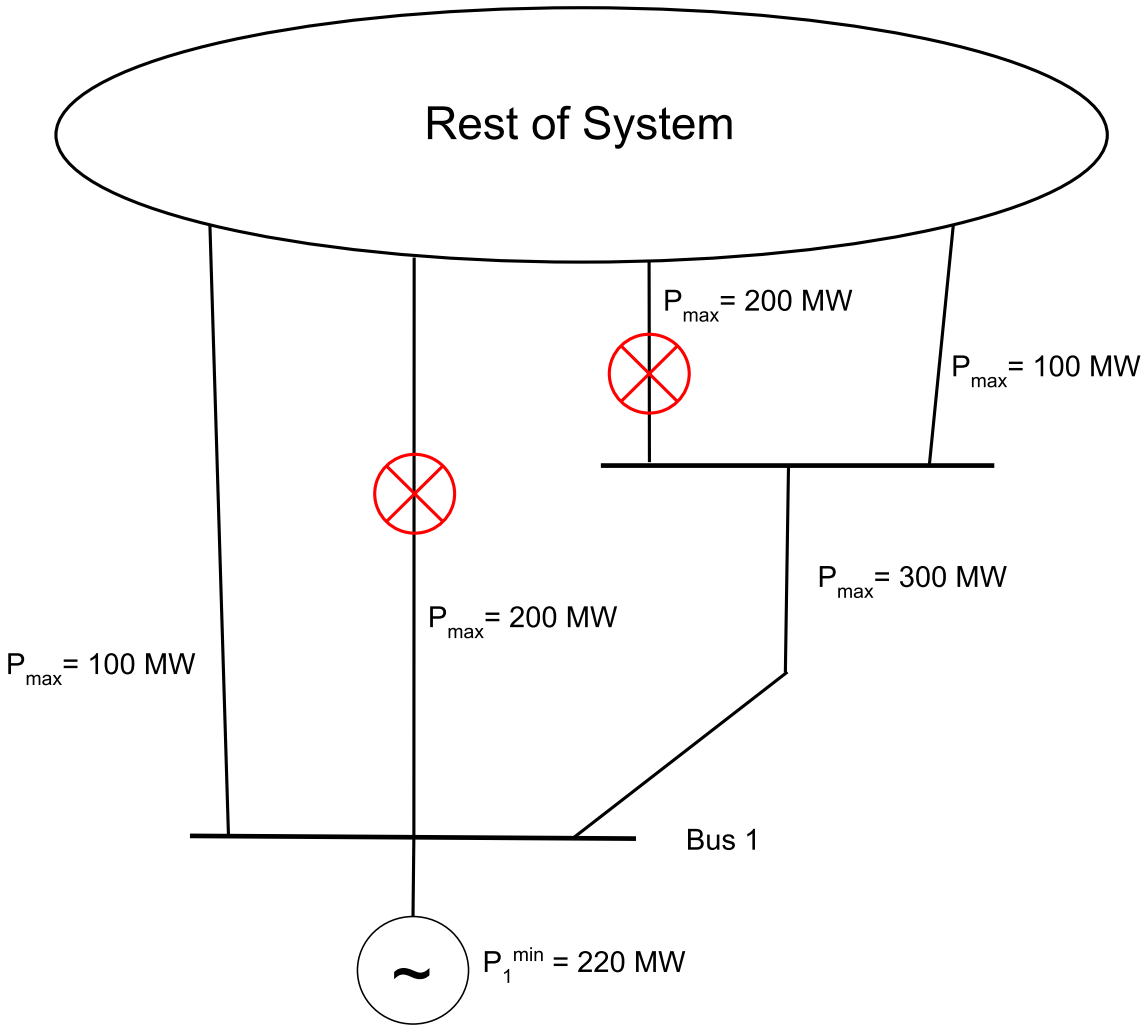

Figure 7: A potential failure scenario for bus removal programs. The lines with red x's indicate removed transmission lines. The maximum capacity of each line is indicated next to each line. The load demand on the top right bus is zero in this scenario.

3.3 Asset Sequencing Methods

As discussed above, it is critical to find the optimal sequence for repairing damaged grid elements which minimizes the area of the resilience triangle. This area, also known as the blackout value or *cost* can be expressed succinctly as the following:

$$A = \sum_{i=1}^{n} P_i t_i \tag{12}$$

where A is the area of the resilience triangle (or cost), n is the number of damaged assets, P_i is the load outage before the i'th asset in the repair sequence is repaired, and t_i is the time required to repair asset i. Note that the load outage includes both load that is shed as well as load which is connected to a bus which is currently out of service. This summation can be seen visually in Figure 8 as the total area of the red boxes. Visually the figure shows that after starting with a large amount of out of service load, repairing grid assets slowly restores some load back into service until the grid is fully operational. It should be noted that in general repairing a grid asset does not necessarily return any additional load to service.

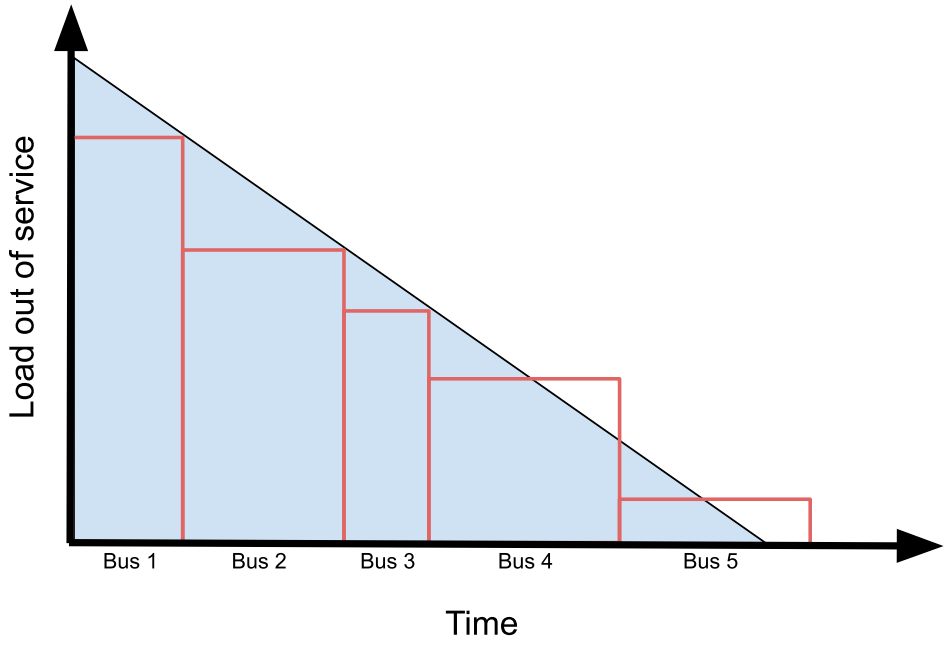

Figure 8: In this event, there are 5 damaged grid buses which have been sequenced for repair. On the time axis, the label bus x means that the xth bus in the sequence is currently being repaired at that time. A triangle has been overlaid to show that the cost area somewhat resembles a triangle.

To find the repair sequence which minimizes the cost, an initial first try may be to

attempt an exhaustive search of all possible repair sequences. However, to evaluate the cost formula for all sequences of n damaged assets, the number of simulations one would have to run would scale as $n!$.

Instead, this process can be run as if one is branching out into all possible sequences of repair. Given n damaged grid elements, attempt scenarios where each of the n elements is repaired first. Then, depending on the outcome of that repair, some scenarios (or in-progress sequences) can be dismissed. Then for each remaining sequence, every remaining grid element is added to that sequence to create a set of repair sequences of length 2. Some of those sequences are then dismissed. This repeats until the sequences reach length n, where the sequence with the lowest cost is chosen as the best order to repair the damaged grid assets. Pseudocode for this methodology can be seen below in Algorithm 1, with code on github at https://github.com/Parveer-B/Get-weather-data. This fashion of dismissing in-progress subsequences is akin to the "pruning a tree" [33].

To use the above formulation, a methodology must be devised to prune the in-progress sequences. Unfortunately, a truncated cost formula is not ideal for evaluating these sequences as it is only able to include the amount of load shed after that asset is repaired if the time it takes to repair the next asset is known. For an in progress sequence that next asset has not yet been determined. To resolve this, two different mock costs, termed a *restore based* and *load based* mock cost were developed which could assess these in-progress sequences.

The load based mock cost (LMC) is similar to the cost except instead of multiplying the time it takes to repair a given asset by the load outage before it is repaired, it is multiplied by the load outage **after** it is returned to service. It is termed "load based" because the amount of load outage is used in the formula. This formulation was chosen as an option because it includes the effect of restoring the grid asset in its formula. In-progress sequences are considered better if its load based mock cost is lower. Specifically, for an in-progress sequence of length k, which contains buses labelled 1,2,....k in order, the LMC is given as in the equation below. Its value is also shown visually in Figure 9 as the area shaded in green:

$$LMC = \sum_{i=1}^{k} P_{i+1} t_i \tag{13}$$

The restore based mock cost (RMC) on the other hand explicitly uses the amount of load that repairing a specific grid element brings back into service. This formula is similar Smith's weighted processing time rule [12] which has been used incurrent industry practice [10] with a higher value being better. In this case, the total time was used in the denominator, since using the time it takes to repair the latest asset would often result in cases where in-progress sequences which include the same assets have identical mock costs, regardless of the order

Algorithm 1 Repair Sequencing General Form

Input:
  $n \leftarrow$ number of buses and lines removed
  $x \leftarrow$ number of sequences to store
Output: Sequence with lowest cost
Initialize:
  $curseq \leftarrow []$ ▷ Store sequences with current blackout value, time, and criterion
  $newseq \leftarrow []$ ▷ Store newly generated sequences
for $k = 1$ to $n - 1$ **do**
  if $k = 1$ **then**
    for each element (i.e., bus/line) j from 1 to n **do**
      $toaddseq \leftarrow j$ ▷ Generate sequence with initial repair of element j
      Compute $getoutagedload(toaddseq)$ ▷ Get load outaged with buses and lines in toaddseq back in the system case
      Append $toaddseq$ to $newseq$ with updated blackout value, and restoration time
    end for
    $curseq \leftarrow$ a subset of $newseq$ selected based on some pruning criterion
    $newseq \leftarrow []$ ▷ Reset $newseq$ for next iteration
    continue
  end if
  for each sequence seq in $curseq$ **do**
    for each element (i.e., bus/line) j from 1 to n **do**
      if bus j is not in $seq.sequence$ **then**
        $toaddseq \leftarrow seq.sequence + j$ ▷ Append element j to $seq.sequence$
        Compute $getoutagedload(toaddseq)$
        Append $toaddseq$ to $newseq$ with updated cost and restoration time
      end if
    end for
  end for
  if $k \neq (n - 1)$ **then**
    $curseq \leftarrow$ a subset of $newseq$ selected based on some pruning criterion
    $newseq \leftarrow []$ ▷ Reset $newseq$ for next iteration
  else
    $curseq \leftarrow newseq$ ▷ Keep all sequences
  end if
end for
for each sequence seq in $curseq$ **do**
  Add the remaining grid element to $seq.sequence$
  Update cost (assuming load outage after restoration of all damaged buses and lines is 0)
end for
Sort $curseq$ by cost (ascending order)
return the sequence with the lowest cost

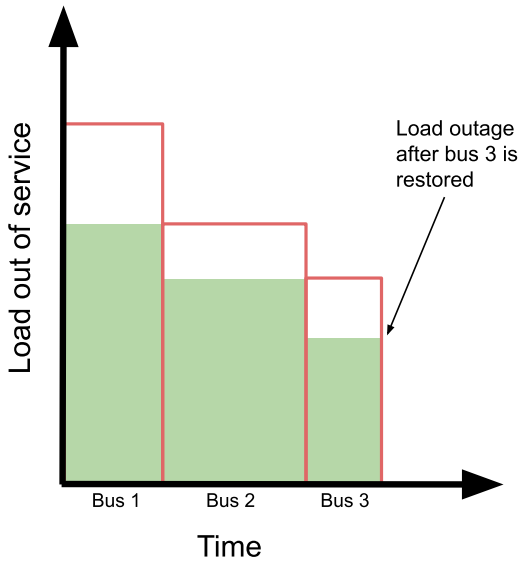

Figure 9: The load-based mock cost value is highlighted in green for an ongoing repair sequence that currently includes three buses. The upper edge of the red rectangles represents the current load that is out of service at a given time.

in which they were returned to service. This is despite the cost (or blackout value) of those in-progress sequences being different. The restore based mock cost for a sequence of length k is given in Equation (14) and shown visually in Figure 10. In cases where two in-progress sequences may have the same RMC, for example if repairing the assets in the either sequence has not yet returned any load to service, the time it takes to repair the assets in the sequences is used as a tiebreaker.

$$RMC = \sum_{i=1}^{n} \frac{P_i - P_{i+1}}{\sum_{j=1}^{i} t_j} \tag{14}$$

The mock costs are used to evaluate in-progress sequences in Algorithm 1 lines 15 and 29, and will be compared in section 3.4.2. However, to decide the number of in-progress sequences kept (i.e saved in *curseq*) in a given iteration, various pruning methods are used. These will be described in the upcoming sections.

3.3.1 NxGreedy

Given a list of of n damaged grid assets, the NxGreedy method involves finding all permutations of these removed grid assets of length x. After generating these permutations, the in-progress sequence with the best mock cost is chosen to 'move forward' (i.e remain un-

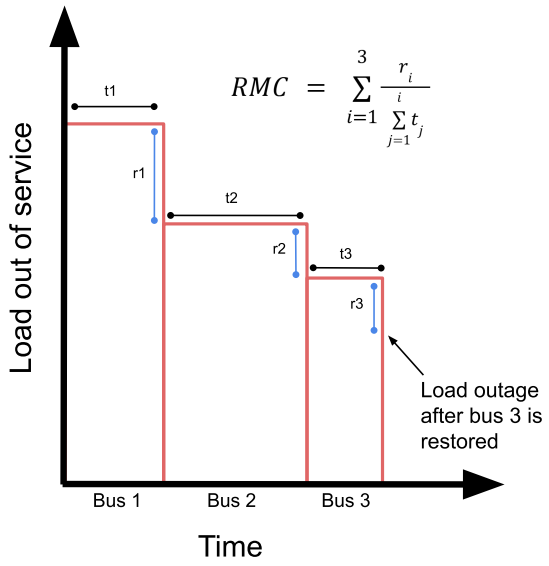

Figure 10: The restore-based mock cost value is given by the formula for an ongoing repair sequence that currently includes three buses. The upper edge of the red rectangles represents the current load that is out of service at a given time.

pruned). Afterwards each remaining asset is appended as the next elementin the sequence. The best subsequence is chosen, and the program proceeds greedily until all buses are added. In Algorithm 1 lines 15 and 29, this corresponds to setting $curseq$ equal to $newseq$ if $k < x$, and having curseq only store the best in-progress sequence in newseq if $kgeqx$. The goal of this method is to capture sets of buses of size x or less which only return large amounts of load back into service if they are all repaired. In a purely "greedy" method, where only the best in-progress sequence is kept at each iteration, this could potentially be missed.

If x is one, the method becomes purely greedy. Conversely, if x is equal to the number of assets removed (n), this method becomes an exhaustive search.

The number of power grid simulations (not counting additional islands which may form) run in this method is:

$$\sum_{i=1}^{x} \frac{n!}{(n-i)!} + \sum_{i=1}^{n-x} i \tag{15}$$

where n is the total number of assets removed. Note that the final iteration is not simulated as it is assumed that there will be no load shed after all assets are repaired. Instead, all subsequences from the $n-1$th iteration are kept. This is true for all methods.

3.3.2 Keepx

The Keepx algorithm adds every remaining grid asset to every existing sequence, before sorting them by their mock cost. It then keeps the top x sequences, proceeds to the next iteration, and repeats the process until all grid assets are back in service. In Algorithm 1 lines 15 and 29, this corresponds to letting $curseq$ equal to the best x sequences in $newseq$ by mockcost.

With this approach, the total number of simulations is given by:

$$n + \sum_{i=2}^{n} x * (n - i + 1) \tag{16}$$

If $x = 1$, this method becomes greedy.

3.3.3 Closecostx%

Closecostx% works almost exactly the same as Keepx. However instead of keeping the best x sequences by mock cost, all sequences in $newseq$ within x% of the best mock cost are stored in $curseq$. This approach does not have a set number of simulations, and thus the time it takes to complete is more variable than the others.

3.3.4 PerxGreedy

PerxGreedy is very similar to NxGreedy. However, after greedily picking the best sequence at iteration $k = x$, the method once again adds all permuations of length x of the remaining grid assets to the chosen subsequence. This continues every x iterations. This is akin to setting $curseq$ equal to $newseq$ in Algorithm 1 lines 15 and 29 if $k\%n \neq 0$, and storing only the best sequence by mockcost in $curseq$ if $k\%n \equiv 0$. The goal of this method is similar to that of NxGreedy. However instead of just finding sets of buses which restore a lot of load when repaired together at the *start* of the sequence, this method can find these sets at any point in the sequence at the cost of additional computation time.

3.4 Method Evaluation

3.4.1 Storm Travel Line

To evaluate mock costs and pruning methods, asset damaging fake disasters must be generated. In this section, it was assumed that only grid buses were damaged. To remove buses

a "storm travel line" was used as a mechanism to mock a path that astorm might travel. This storm travel line entails removing a bus at random. Then usingthe lines connected to that bus, an adjacent bus is chosen to remove. Using the set of busesadjacent to both buses another bus was selected, repeating the process until a predetermined number of buses (n) is removed. Additionally, in this chapter, the load demand on the Texas and Hawaii cases was scaled so that the maximum allowable power generated by the system's generators were close to the load demand. This was done to prevent "wasted" cases where there was minimal load outage. For the Texas case, this load scaling was set at 1.25x, while for the Hawaii case this was 1.2x. These values were chosen because increasing the load scaling slightly beyond these values resulted in load shedding, even without any buses being removed.

3.4.2 Mock Cost Evaluation

The two mock costs were compared by generating 100 "storms" via the storm travel line method in the Hawaii mock grid case with 5 buses removed. The repair sequences were found using Algorithm 1 and both mock costs were used individually throughout the algorithms process in lines 15 and 29 for a given event. The same events were used for both mock costs. Using a purely greedy (i.e N1Greedy) method for pruning repair sequences, the LMC resulted in finding sequences with a blackout value which were 12% higher than that when using the restore based mock cost on average. Of the 100 storms, the RMC found sequences with a lower blackout value than the LMC 88 times, and found sequences with the same cost 6 times. The LMC was only better than the RMC 6 times. Using the Keepn method (i.e. Keep5 in this case), the LMC had an average cost which was 1.9% higher than the RMC. Here, the LMC was better only 3 times, while the RMC was better 41 times.

150 scenarios of the Hawaii case were then run with 3 buses removed. Here, was Keepn was not tested as all in-progress sequences are kept in the $n-1$th (i.e 2nd) iteration causing Keepn to be an exhaustive search here. Once again, the same storms were used to test both mock costs Using the greedy method, the LMC was 4.3% worse than the RMC, and it was only better in 5 of the 150 scenarios. It was worse in 63 of the 150 simulations.

These results demonstrate that the RMC is likely better than the LMC. This can be attributed to the fact that when a sequence begins, the load remaining is likely significantly larger than the load that is returned to service simply by putting one bus back in service. Thus, at the start the actual effect of repairing a bus probably has very little effect on the LMC. The simulation will then likely simply choose the bus which is the fastest to repair, regardless of how much load it restores. With the RMC, the amount of load restored is used directly. Consequently, at the start of the sequence it will prioritize repairing buses which

restore the most load as well as taking the least time to repair.

Nonetheless, there are cases where the load-based mock cost proves more effective. For instance, there can be a scenario where two buses are damaged but can be quickly restored with neither bus restoring any load individually. However, the restoration of both together may restore a significant amount of load. In this situation, the restore based mock cost with the greedy method will instead choose to restore a bus that restores some amount of load, even if that bus takes a lot longer to restore. Despite this, the restore based mock cost is the only one used for the rest if this paper as it is better in the majority of cases.

3.4.3 RAD

These tree pruning mechanisms were tested against the RAD method proposed in [8] and described in Section 1.2. In this paper, random partitioning was implemented such that each partition contained at least two elements, with a maximum partition size being half the number of buses out. This maximum limit was chosen to ensure that solving the ordering problem for the partitions remained computationally feasible. The partitioning process involved selecting a random number between two and the maximum size, forming a partition, and repeating until all buses were assigned. Both exhaustive search and a purely greedy approach were tested for solving the ordering problem for a partition. If the exhaustive search was used, the maximum partition size was set at 4 to not have an overly long computation time. The stopping condition was defined as the entire sequence maintaining the same cost for x consecutive iterations or 30 iterations total, whichever comes first.

Four ways of generating initial sequences (or utilization heuristics) were tested. In [8] they found a sequence by the sorting buses by the amount of power flow through them. In this paper, this was done by running a DC-OPF to solve the economic dispatch problem and summing the outward flow from each bus. The power flow through each bus divided by the time it takes to repair that bus was also tested. As well, [10] states that repair sequencing by sorting by the load of each bus and by the load of each bus divided by time is used in practice. Both of these were also tested.

These initial sequences were tested in three different trials, with the partition ordering problems solved optimally. The first and second trial both involved 150 "storms" generated via the storm travel line with 5 buses removed in the Hawaii case. The third trial was 150 storms with 8 buses removed in the Hawaii case. For all three of these trials, the stopping condition was set as the sequence maintaining the same cost for 3 consecutive iterations. As well, the same storms were used for each heuristic. The results of this testing are in Table 1, and the average time to run each storm for each trial is given in Table 2. The values in

the table indicate the percentage by which the average cost of each method is lower than the baseline cost. In other words, a value of 1% in the table indicates that that the average cost of that method is 1% lower than the baseline cost. The baseline was arbitrarily set to be the utilization heuristic which sorts by the amount of load.

	Sort by Load	Sort by Load / Repair Time	Sort by Power Flow	Sort by PF / Repair Time
5 buses out	0	-0.35%	-0.29%	-1.63%
5 buses out (v2)	0	-0.34%	-0.59%	-1.16%
8 buses out	0	-1.31%	-0.55%	-1.19%

Table 1: Comparison of RAD utilization heuristic costs

	Sort by Load	Sort by Load / Repair Time	Sort by Power Flow	Sort by PF / Repair Time
5 buses out	7.54	7.66	8.26	6.91
5 buses out (v2)	7.58	7.41	8.15	7.16
8 buses out	45.17	48.33	47.11	44.77

Table 2: Comparison of RAD utilization heuristic run times

From the results it is clear that simply sorting the buses by the amount of load connected to them performs the best. In terms of run time, this method is on the same order as the other ones, and is only consistently slower than starting by sorting by power flow/time (which is the worse utilization heuristic). Thus, for RAD in this paper, the initital sequence of buses is determined by sorting by the amount of load connected to each one.

3.4.4 5-Bus Pruning Method Evaluation

In a first round of testing, the N1Greedy, N2Greedy, Closecost1%, Keepn, Exhaustive Search, Closecost2% and Keepn/2 methods were tested, with n being the number of buses damaged. In Keepn/2 uses the Keepx method with $x = \lfloor \frac{n}{2} \rfloor$ These methods were first tested in the Texas case with 5 buses removed. There were 150-175 storms simulated for each method, and the same storm was **not** used for each method here. However, N1Greedy was used every time as a baseline. 'Percentage Better than N1Greedy' refers to the percentage by which the blackout value of the sequence obtained using that method is lower than the one found by N1Greedy. This percentage difference is averaged over all the storms. On the right side of Figure 11, the x-axis gives the average that percentage over the first x storms. This was used to test the convergence of the results. The average percentage better than N1Greedy over all the storms simulated for that method is shown on the left of Figure 11.

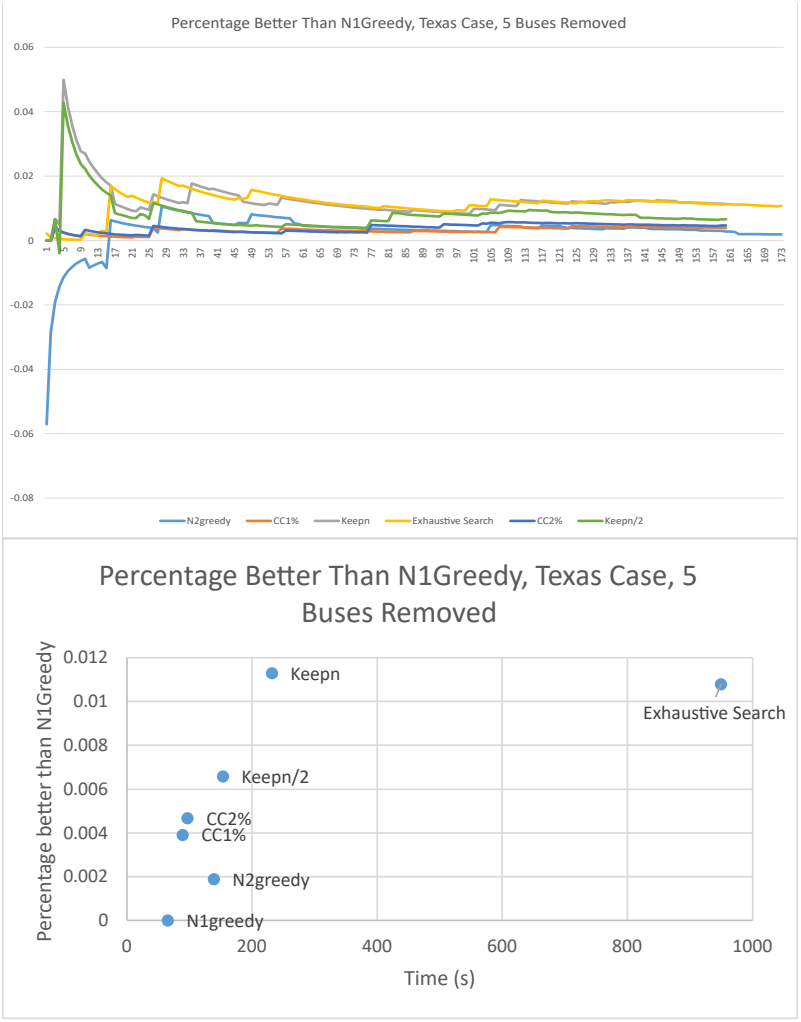

Figure 11: Various pruning methods tested for different storms in the Texas case. N1Greedy is used as a baseline.

Several observations can be made from the graph above. The first notable point is that Keepn appears to outperform the exhaustive search. This is due to the algorithms being conducted on different cases. As additional storms are simulated, Keepn will likely fall below it. Nonetheless, this result demonstrates that Keepn is consistently very close to finding the optimal restoration sequence while operating in roughly one-third of the time.

Another key observation is that N2Greedy appears to perform the worst among these methods. A discernible trend can be observed where an increase in computation time corresponds to improved accuracy. This trend can be visualized by a "line" passing through the points for Ng1reedy, Closecost1%, Closecost2%, Keepn/2, and Keepn. However, N2Greedy exceeds the processing time of Closecost2% while yielding less accurate results.

Finally, the cost results are relatively consistent across the different pruning methods.

Ngreedy1 is only about 1.2% worse than the exhaustive search, suggesting that it is an option to save time without sacrificing much efficacy.

The convergence data shows that these results were able to convergerge within the 150-175 storms demonstrating that the uncertainty on the results is likely quite low.

The same test was run on the Hawaii case. This time 500 storms were run for each method, with the same storms used for each. More storms were run on the Hawaii case since the smaller case file generates solutions faster. Results are shown in Figure 12.

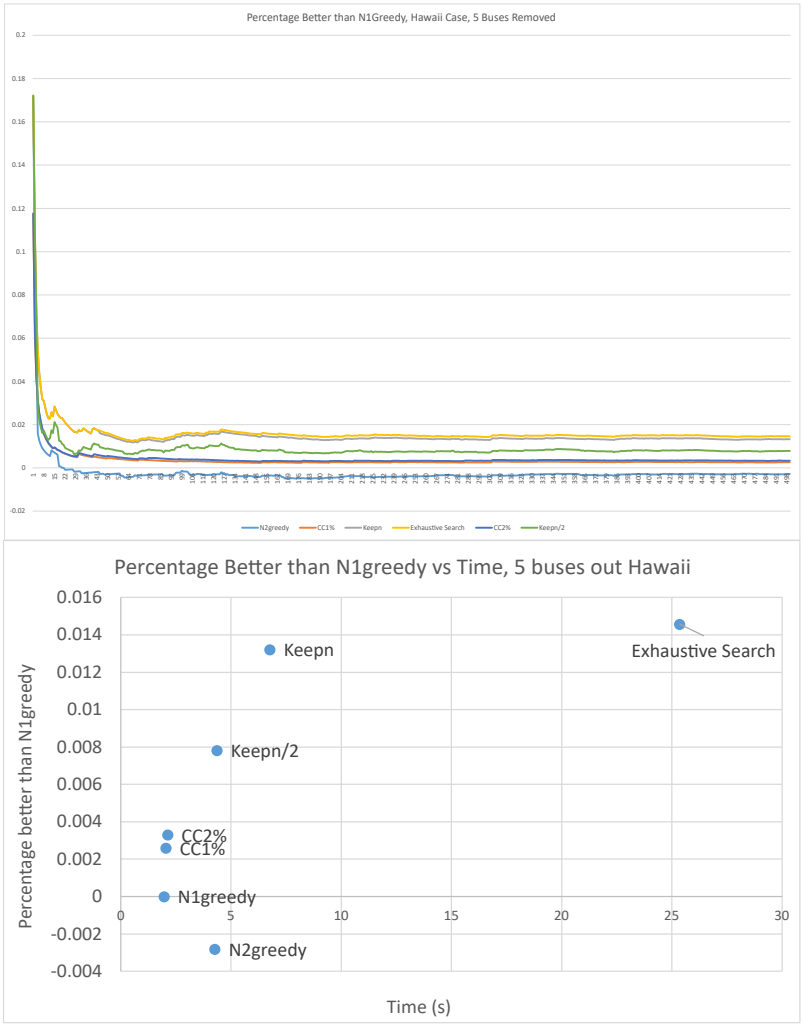

Figure 12: Various pruning methods tested for different storms in the Hawaii case. N1Greedy is used as a baseline.

In these cases, it is clear that Keepn outperforms the other options and is quite close to optimal on average despite taking a third of the time. Additionally, N2Greedy performs worse than N1Greedy despite taking more time to compute. Following these results, N1Greedy and Keepn were chosen for further testing. These were chosen since N1Greedy is a fast

and less accurate method, while Keepn takes longer computationally while being quite close to finding the optimal solution in cases with 5 buses removed. While Closecost1% and Closecost2% are slightly better than N1Greedy with very little variation in the average time to complete, the performance increase was not deemed to be high enough to compensate for the uncertain solution time of these methods. However, Keepn/2 is still a possible option to balance computational time and efficacy if needed. A summary of the performance based can be seen in Table 3.

Method	% Worse than Optimal, Hawaii	% Worse than Optimal, Texas
N1Greedy	1.43%	1.07%
N2Greedy	1.71%	0.88%
CC1%	1.18%	0.68%
CC2%	1.11%	0.61%
Keepn	0.13%	-0.05%
Keepn/2	0.67%	0.42%

Table 3: Comparison of algorithm performance relative to optimal

The efficacy of the methods seems to follow approximately the same ordering in either case, with only N1Greedy switching with N2Greedy when going from Hawaii to Texas. However, there is a larger variability in the effectiveness of the each method in the Hawaii case. For example N1Greedy is 1.43% worse than optimal in the Hawaii case, and only 1.07% worse than optimal in the Texas case. This is likely beause the smaller scale of the Hawaii system causes the removed buses to have had a larger impact on the performance of the network with the Texas case having more options to make up the lost load. Thus, there is a larger variability in the cost of different restoration solutions, leading to worse results on average relative to an exhaustive search.

3.4.5 10-Bus Pruning Method Evaluation

RAD, Per2Greedy and Keepn were then tested relative to N1Greedy. Four different variations of RAD, the optimal and greedy solving of the partition ordering problem paired with a stopping condition of three and six consecutive sequences (denoted by "same" in 13) were used. In this test, ten buses were removed on the Hawaii case, with 288 storms run. These storms were the same for each method. Ten buses were tested here to allow for a larger number of allowable partitions for the RAD algorithm based on the constraints discussed above. With five buses removed, there are only three allowed partitions with those stipulations making RAD less effective. Results are shown in 13.

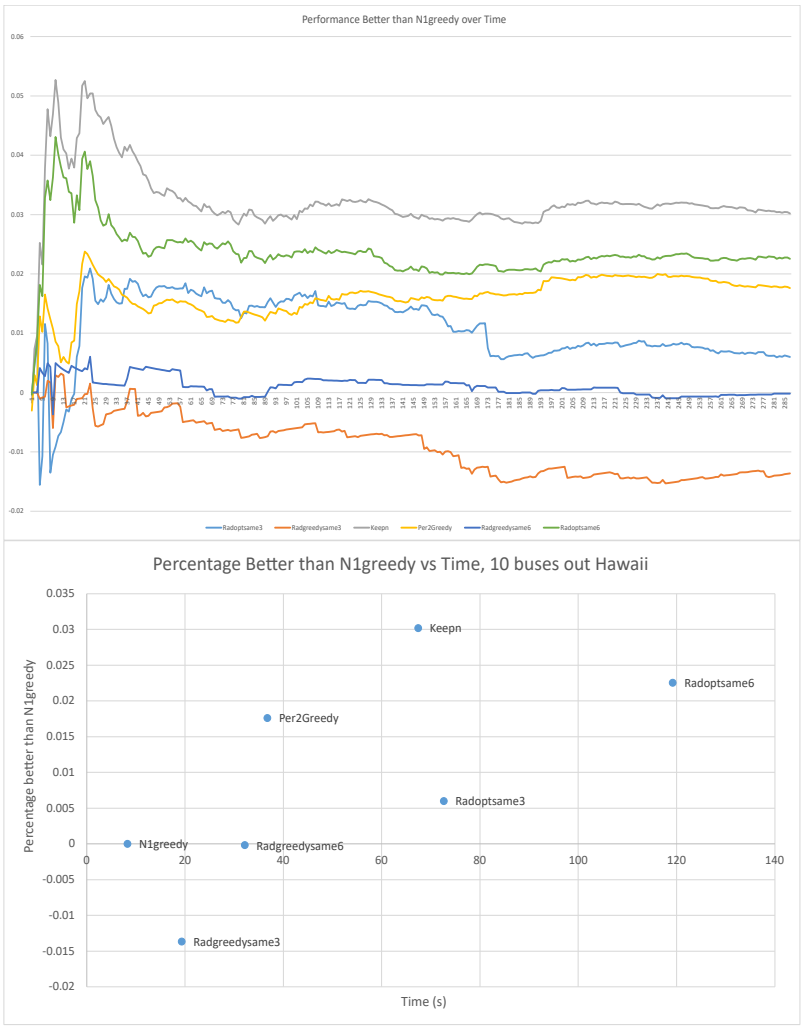

Figure 13: Various pruning methods and RAD tested for different storms in the Hawaii case. N1Greedy is used as a baseline.

The convergence chart shows that the solutions have converged and thus the chart results are decently accurate. The greedy sorting of partitions in RAD performs worse than greedily pruning in-progress sequeunces, while taking double and triple the time. Additionally, Radoptsame3 performs worse than Per2Greedy while taking double the time, while Radoptsame6 performs worse than Keepn while taking almost double the time. Thus, for storms with ten or less assets removed these pruning methods should be used instead of RAD. Additionally, it should be noted that Keepn's efficacy increased from 1.3% better than N1Greedy with five buses removed in the Hawaii case to 3% better. This was expected as the number of solutions scales up drastically as the number of buses removed increases, and thus it is more likely that N1Greedy prunes a critical subsequence which Keepn keeps. Additionally, with 10 buses removed, there is likely a larger variance in the cost of different

possible repair sequences, leading to a larger variance in the efficaccy of different solutions. It is possible that at ten buses removed, Keepn is now significantly worse than an exhaustive search. Unfortunately, conducting that test by finding all $10! = 3628800$ possible sequences was determined to be unfeasible due to time and storage constraints.

As a final test, Keepn, Per2Greedy and N1Greedy were tested with 10 buses removed in the Texas case. This was conducted to measure the approximate consistency of the results across different grid networks. 836 storms were run, with the same storms for each method. Results are in 14.

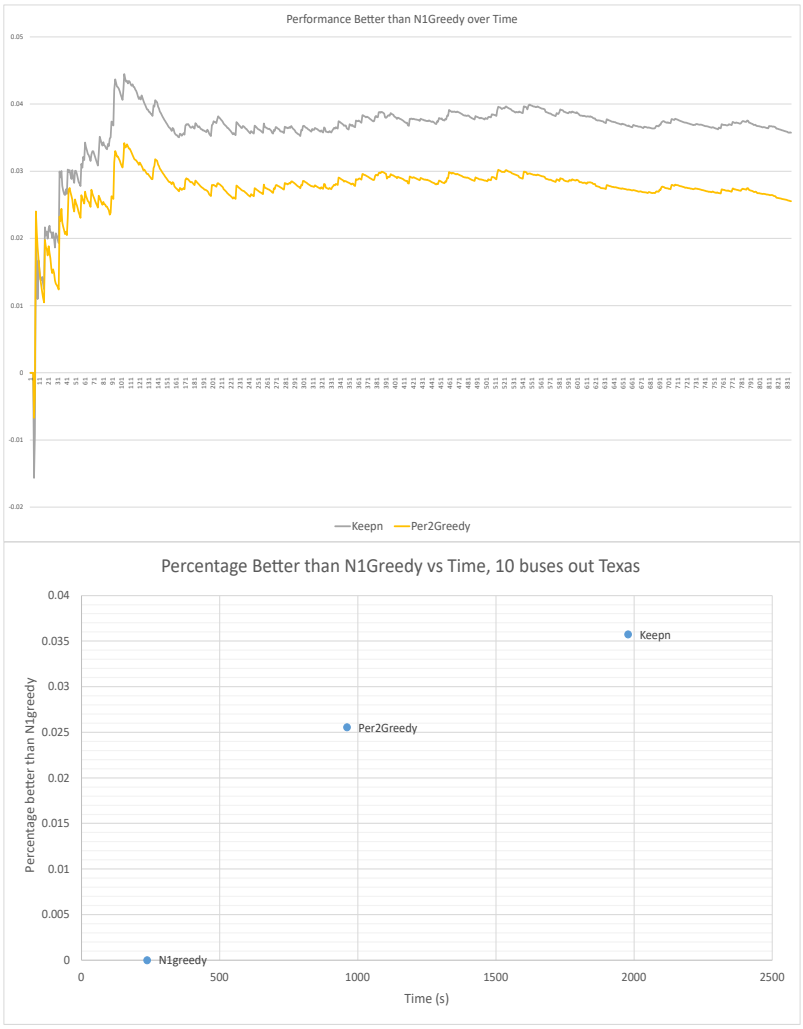

Figure 14: N1Greedy, Per2Greedy and Keepn tested for different storms in the Texas case. N1Greedy is used as a baseline.

The relative ordering of the effectiveness and time required for Keepn, Per2Greedy and N1Greedy is similar here as in 13. Interestingly the percentage difference between Keepn and Per2Greedy relative to N1greedy when switching from the Texas to the Hawaii case actually

decreases, with Keepn being about 3% better than N1Greedy in the Hawaii case and over 3.5% better in the Texas case.

3.4.6 Summary of Results

In summary, the pruning methods have been shown to be an effective way to find bus repair sequences in storms with less than 10 buses removed. With 5 buses removed, Keepn is extremely close to finding the optimal solution, and even a purely greedy method in N1Greedy is only 1 to 1.5% worse than the optimal solution. With 10 buses removed, the pruning methods outperform the RAD method from [8], although their performance relative to a full exhaustive search is unknown. In general, the Keepn method appears to have the best performance at the cost of computational time, while the N1Greedy method is a much more time-efficient method, albeit with a slight decrease in efficacy. For methods which balance computational time and efficacy, Keepn/2 and Per2Greedy have presented themselves as viable alternatives.

4 Bus Importance Simulation

4.1 Simulation Setup

This section will outline the bus importance simulation run as a proof of concept for the above's methodology. Tornadoes will be generated as discussed in Section 2.2, and buses and transmission lines will be removed as described in Section 2.3. Afterwards power system simulations will be run using methods in Section 3. After running a large number of tornadoes, an asset importance metric will be calculated.

For this simulation, the Texas system case will be used, with bus and transmission line locations being provided by the casefile. For tornado generation, the heatmap region was cut into a rectangle which encompasses the entire state of Texas, so that tornadoes would only be generated in that area. The NARVAL supercomputer, a computing cluster located at the École de technologie supérieure in Montréal [34] was used to complete these simulations. The cluster has a AMD EPYC 7532 CPU which has a 2.40 GHz clock speed.

When generating magnitude 5 tornadoes, tornado angle, width and length data will additionally be sampled from surrounding states (Louisiana, Oklahoma, Arakasas and New Mexico) in order to have a wider range of possible generated tornado shapes. Currently, there are only six recorded magnitude five tornadoes in Texas, a number that increases to 18 if surrounding states are included.

Unfortunately, simulating the DC-OPF for the Texas case takes approximately 3 seconds on the NARVAL supercomputer. To compute the asset importance metric for a tornado that removes k buses, we must determine the area of the resilience triangle (or cost) for two cases: (1) the event where k buses are removed and (2) separate scenarios where each of those k buses remains undamaged while the others are removed. This requires an additional k simulations for cases with $k-1$ buses removed. Using formula 16, which determines the number of simulations required to compute the cost of an event with n buses removed, we can calculate the total number of power grid simulations needed to determine the asset importance metric via the Keepn method for a tornado that removes k grid elements. This number of simulations can then be converted to the approximate time required to complete the power system simulations for that one tornado on the NARVAL supercomputer. This is presented in Figure 15.

From Figure 15, it is clear that the time required for the metric to be calculated becomes

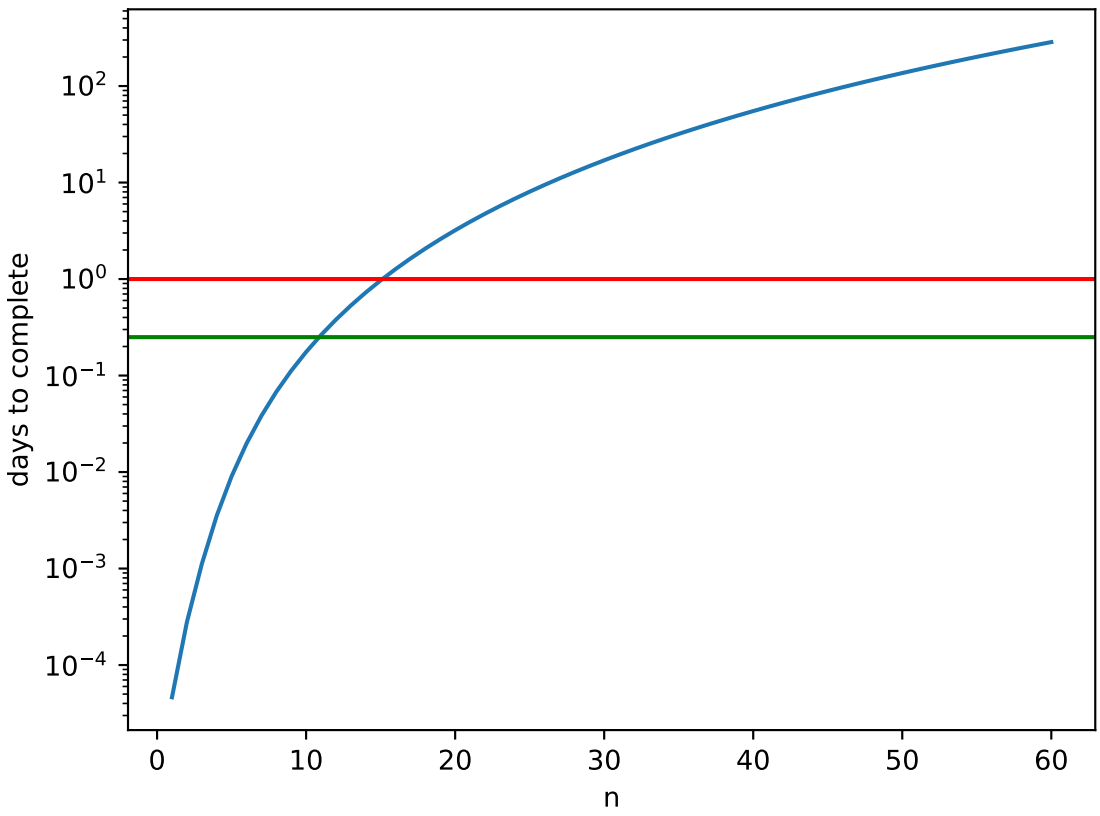

Figure 15: The time required on NARVAL to find the asset importance metric with the Keepn pruning method for a storm with n assets damaged. The red line represents over a day to complete, while the green line is 6 hours (deemed the maximum "reasonable" time to complete the event).

unreasonable extremely quickly. Passing six hours with after $n = 10$, and one day after $n = 15$. Conversely, if N1Greedy is used, the results are as follows in Figure 16.

This is a bit more reasonable with it being reasonable below $n = 22$ and only taking over a day at about $n = 35$. However some storms, partiularily those through downtown cores such as Dallas and Houston can often take out upwards of 50 grid assets. If $n = 60$, even with the purely greedy method, it would still take about five days to calculate the importance metric for that single storm.

Another possibility is to vary the method depending on how many assets are damaged. For example, if Keepn is used for $n \leq 8$, Keepn/2 is used for $9 \leq n \leq 12$, Keepn/4 (i.e Keepx with $x = \lfloor \frac{n}{4} \rfloor$) for $13 \leq n \leq 15$ and greedy otherwise, the graph becomes that in 17.This formulation is somewhat counterintuitive as methods such as Keepn get better relative to

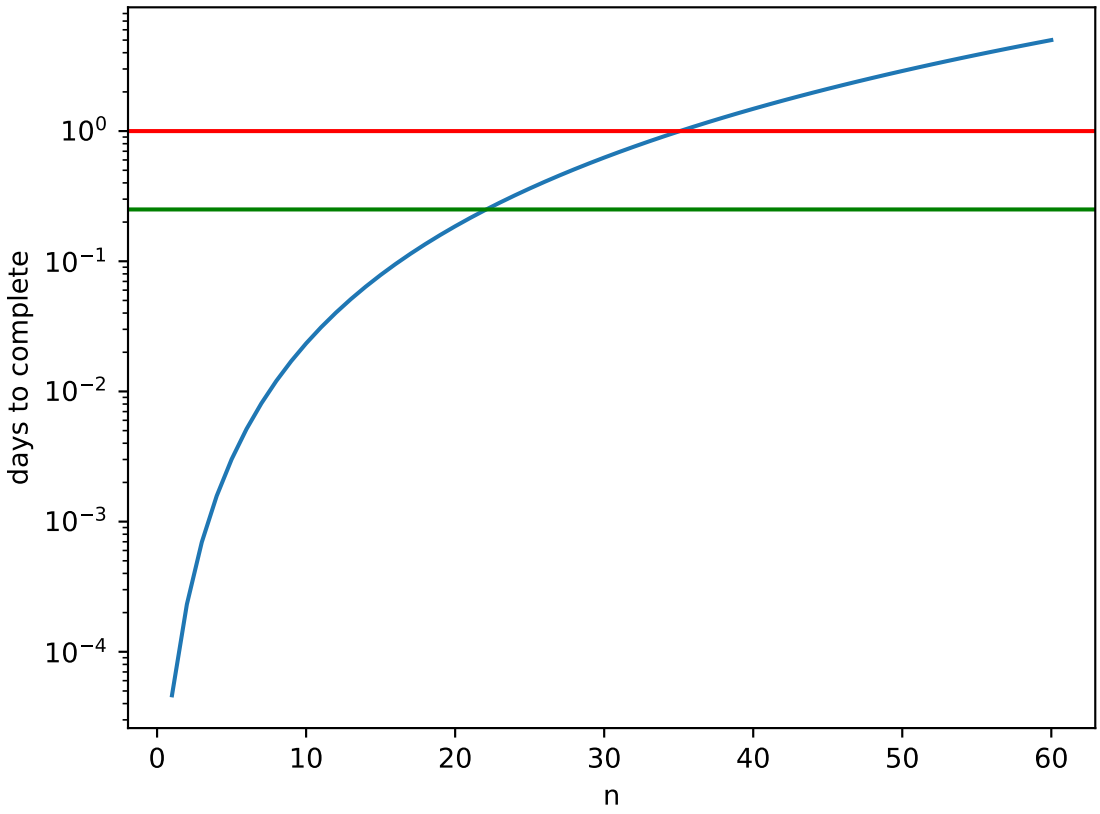

Figure 16: The time required on NARVAL to find the asset importance metric with the N1Greedy pruning method for a storm with n assets damaged.

N1Greedy if more assets are removed. However, here it is N1Greedy that is being used in situations with more assets damaged. Nonetheless, it ensures that simulation times stay reasonable for as long as possible when using pruning methods.

In this paper, due to the short timelines of the EngSci thesis project, N1Greedy was used as the pruning method for all storms. To run the simulation, storms were generated in batches of 200, and 200 NARVAL jobs were submitted to run power system simulations and calculate the bus importance metric for each batch. Each batch was given a week (the NARVAL job time limit) to complete as many storms as possible. Data was saved after every 1000 storms within a batch to prevent data loss. It should be noted that this implicitly does affect the importances of assets in high density areas, as those batches which may end up with a lot of storms in those areas will likely take longer to run, and thus less importance metrics for storms in those areas will be calculated.

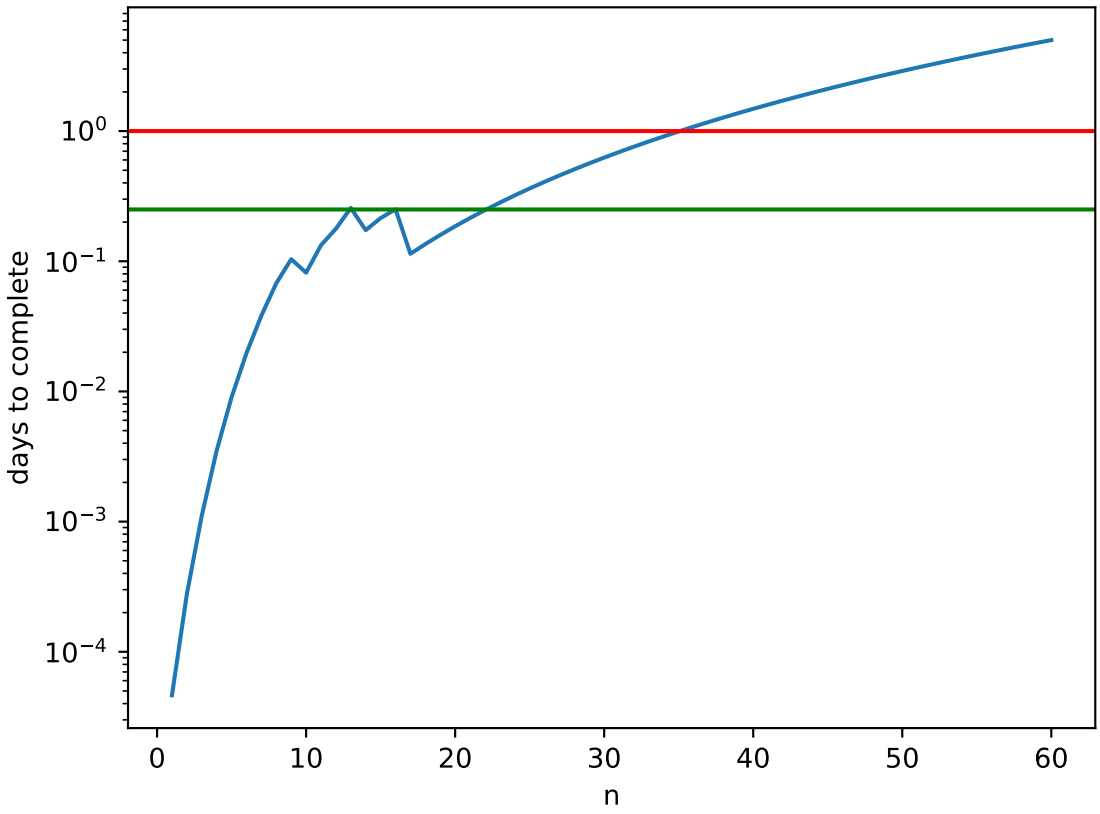

Figure 17: The time required on NARVAL to find the asset importance metric for a storm with a combination of pruning methods based on the number of assets damaged.

4.2 Simulation Results

Bibliography

[1] Chad Shouquan Cheng, Heather Auld, Qian Li, and Guilong Li. Possible impacts of climate change on extreme weather events at local scale in south–central canada. *Climatic change*, 112:963–979, 2012.

[2] Mathaios Panteli and Pierluigi Mancarella. Influence of extreme weather and climate change on the resilience of power systems: Impacts and possible mitigation strategies. *Electric Power Systems Research*, 127:259–270, 2015.

[3] Fauzan Hanif Jufri, Victor Widiputra, and Jaesung Jung. State-of-the-art review on power grid resilience to extreme weather events: Definitions, frameworks, quantitative assessment methodologies, and enhancement strategies. *Applied energy*, 239:1049–1065, 2019.

[4] Jerry J Ancona. A framework for power system restoration following a major power failure. *IEEE Transactions on power systems*, 10(3):1480–1485, 1995.

[5] Jiahao Yan, Bo Hu, Kaigui Xie, Heng-Ming Tai, and Wenyuan Li. Post-disaster power system restoration planning considering sequence dependent repairing period. *International Journal of Electrical Power & Energy Systems*, 117:105612, 2020.

[6] Bo Chen, Chen Chen, Jianhui Wang, and Karen L Butler-Purry. Multi-time step service restoration for advanced distribution systems and microgrids. *IEEE Transactions on Smart Grid*, 9(6):6793–6805, 2017.

[7] Sylvie Thiébaux, Carleton Coffrin, Hassan Hijazi, and John Slaney. Planning with mip for supply restoration in power distribution systems. In *Twenty-Third International Joint Conference on Artificial Intelligence*, 2013.

[8] Carleton Coffrin, Pascal Van Hentenryck, and Russell Bent. Last-mile restoration for multiple interdependent infrastructures. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 26, pages 455–463, 2012.

[9] Pascal Van Hentenryck, Carleton Coffrin, Russell Bent, et al. Vehicle routing for the last mile of power system restoration. In *Proceedings of the 17th Power Systems Computation Conference (PSCC'11), Stockholm, Sweden*, pages 22–26. Citeseer, 2011.

[10] Yushi Tan, Feng Qiu, Arindam K Das, Daniel S Kirschen, Payman Arabshahi, and Jianhui Wang. Scheduling post-disaster repairs in electricity distribution networks. *IEEE Transactions on Power Systems*, 34(4):2611–2621, 2019.

[11] Sarah G Nurre, Burak Cavdaroglu, John E Mitchell, Thomas C Sharkey, and William A Wallace. Restoring infrastructure systems: An integrated network design and scheduling (inds) problem. *European journal of operational research*, 223(3):794–806, 2012.

[12] Wayne E Smith et al. Various optimizers for single-stage production. *Naval Research Logistics Quarterly*, 3(1-2):59–66, 1956.

[13] Vishvas H Chalishazar, Ted KA Brekken, Darin Johnson, Kent Yu, James Newell, King Chin, Rob Weik, Emilie Dierickx, Matthew Craven, Maty Sauter, et al. Connecting risk and resilience for a power system using the portland hills fault case study. *Processes*, 8(10):1200, 2020.

[14] Yunfei Mu, Lin Li, Kai Hou, Xianjun Meng, Hongjie Jia, Xiaodan Yu, and Wei Lin. A risk management framework for power distribution networks undergoing a typhoon disaster. *IET Generation, Transmission & Distribution*, 16(2):293–304, 2022.

[15] Abdullah M Braik, Abdullahi M Salman, and Yue Li. Reliability-based assessment and cost analysis of power distribution systems at risk of tornado hazard. *ASCE-ASME Journal of Risk and Uncertainty in Engineering Systems, Part A: Civil Engineering*, 6(2):04020014, 2020.

[16] William Hughes, Peter L Watson, Diego Cerrai, Xinxuan Zhang, Amvrossios Bagtzoglou, Wei Zhang, and Emmanouil Anagnostou. Assessing grid hardening strategies to improve power system performance during storms using a hybrid mechanistic-machine learning outage prediction model. *Reliability Engineering & System Safety*, 248:110169, 2024.

[17] Stephen M Strader, Thomas J Pingel, and Walker S Ashley. A monte carlo model for estimating tornado impacts. *Meteorological Applications*, 23(2):269–281, 2016.

[18] National Oceanic and Atmospheric Administration. Noaa tornado data. WebPage, July 2024.

[19] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020.

[20] Alexandre Serrano-Fontova, Haiyu Li, Zhiyu Liao, Magnus Rory Jamieson, Rosa Serrano, Alessandra Parisio, and Mathaios Panteli. A comprehensive review and comparison of the fragility curves used for resilience assessments in power systems. *IEEE Access*, 11:108050–108067, 2023.

[21] Ali Asghar Zekavati, Mohammad Ali Jafari, and Amir Mahmoudi. Regional seismic risk assessment method for electric power substations: A case study. *Life Cycle Reliability and Safety Engineering*, pages 1–11, 2022.

[22] Chao Sheng, Qian Tang, and HP Hong. Estimating and mapping extreme ice accretion hazard and load due to freezing rain at canadian sites. *International Journal of Disaster Risk Science*, 14(1):127–142, 2023.

[23] Peter L Watson, William Hughes, Diego Cerrai, Wei Zhang, Amvrossios Bagtzoglou, and Emmanouil Anagnostou. Integrating structural vulnerability analysis and data-driven machine learning to evaluate storm impacts on the power grid. *IEEE Access*, 2024.

[24] National Weather Service. The enhanced fujita scale (ef scale). WebPage, n.d.

[25] Fernando Bereta dos Reis, Patrick Royer, Vishvas Hiren Chalishazar, Sarah Davis, Marcelo Elizondo, Jeffery Dagle, Alexandre Nassif, Andrija Sadikovic, Elli Ntakou, Olga Soto, et al. Methodology to calibrate fragility curves using limited real-world data. In *2022 IEEE Power & Energy Society General Meeting (PESGM)*, pages 01–05. IEEE, 2022.

[26] Sarah J. Barnes and Frank W. Agnew. *April's Fury: Alabama Power's Transmission Organization Battles Historic Losses after April 27th Storms*, pages 14–25. ASCE, 2012.

[27] MY Xiao, WX Zhou, G Bitsuamlak, and HP Hong. Fragility of transmission tower-line system subjected to concurrent wind and ice accretion. *Journal of Constructional Steel Research*, 222:108925, 2024.

[28] Seyedeh Nasim Rezaei, Luc Chouinard, Sébastien Langlois, and Frédéric Légeron. A probabilistic framework based on statistical learning theory for structural reliability analysis of transmission line systems. *Structure and Infrastructure Engineering*, 13(12):1538–1552, 2017.

[29] Hanyue Li, Ashly L Bornsheuer, Ti Xu, Adam B Birchfield, and Thomas J Overbye. Load modeling in synthetic electric grids. In *2018 IEEE Texas Power and Energy Conference (TPEC)*, pages 1–6. IEEE, 2018.

[30] Texas A&M University College of Engineering. Electric grid test cases. WebPage, June 2024.

[31] Konrad Purchala, Leonardo Meeus, Daniel Van Dommelen, and Ronnie Belmans. Usefulness of dc power flow for active power flow analysis. In *IEEE power engineering society general meeting, 2005*, pages 454–459. IEEE, 2005.

[32] Ray Daniel Zimmerman, Carlos Edmundo Murillo-Sánchez, and Robert John Thomas. Matpower: Steady-state operations, planning, and analysis tools for power systems research and education. *IEEE Transactions on power systems*, 26(1):12–19, 2010.

[33] AL Morelato and AJ Monticelli. Heuristic search approach to distribution system restoration. *IEEE transactions on power delivery*, 4(4):2235–2241, 1989.

[34] Digital Research Alliance of Canada. Narval. WebPage, November 2024.

A Other Algorithms

Algorithm 2 Island Detection

```
1: function TestIslanding(branches, buslist)
2:     foundBuses ← empty list
3:     islands ← empty list
                                                          ▷ Check for empty branches
4:     if branches is empty then
5:         for each bus in buslist do
6:             Append bus to islands
7:         end for
8:         return
9:     end if
                                                          ▷ Check for single branch case
10:     if size(branches, 1) = 1 then
11:         Append {branches[1,1], branches[1,2]} to islands
12:         for each bus in buslist do
13:             if bus is not in islands[1] then
14:                 Append bus to islands
15:             end if
16:         end for
17:         return
18:     end if
                                                          ▷ General case
19:     for each bus in buslist do
20:         if bus is in foundBuses then
21:             Continue
22:         else
23:             curIter ← {bus}
24:             island ← {bus}
25:             Append bus to foundBuses
26:             while true do
27:                 nextIter ← empty list
28:                 for each bus in curIter do
                                                          ▷ Find branches where bus is on the "from" end
29:                     adjBranches1 ← branches where bus on the from end of it
30:                     for each branch in adjBranches1 do
31:                         if bus on "to" end of branch not in nextIter and not in foundBuses then
32:                             Append bus on "to" end of branch to nextIter
33:                         end if
34:                     end for
                                                          ▷ Find branches where bus is on the "to" end
35:                     adjBranches2 ← branches where bus is at the "to" end
36:                     for each branch in adjBranches2 do
37:                         if bus on "from" end of branch not in nextIter and not in foundBuses then
38:                             Append bus on "to" end of branch to nextIter
39:                         end if
40:                     end for
41:                 end for
42:                 if nextIter is empty then
43:                     Append island to islands
44:                     break
45:                 end if
46:                 curIter ← nextIter
47:                 Append nextIter to island
48:                 Append nextIter to foundBuses
49:             end while
50:         end if
51:     end for
52: end function
```