Computational Models for Line Failure Risk and Cascading Power Failures on Bulk Power Systems

By

Eric Anderson

A dissertation submitted in partial fulfillment of

the requirements for the degree of

Doctor of Philosophy

(Industrial and Systems Engineering)

at the

UNIVERSITY OF WISCONSIN-MADISON

2015

Date of final oral examination: August 25th, 2015

The dissertation is approved by the following members of the Final Oral Committee:

Jeff Linderoth, Professor, Industrial and Systems Engineering
James Luedtke, Professor, Industrial and Systems Engineering
Bernard Lesieutre, Professor, Electrical and Computer Engineering
Thomas Rutherford, Professor, Agriculture and Applied Economics
Stephen Wright, Professor, Computer Sciences

Acknowledgements

I would like to thank everyone that makes UW-Madison an excellent place to learn and grow, in particular, to the many Professors I have had the privilege to work with. A major thanks to Jeffrey Linderoth for advising me along the way, slowly converting me to the glory of linux workflows, and teaching me the tricks of the trade. Also, a major thanks to Jim Luedtke for help with major parts of this work. I would also like to thank the rest of the committee Bernard Lesieutre, Thomas Rutherford, and Stephen Wright for their advice and direction from their unique perspectives. I would also like to thank Stephen Robinson for introducing me to Operations Research and showing me the usefulness and power of the field.

Thanks to my Mom, Dad, and Sister for always giving me a supportive environment to develop as a person and the much needed advice on all aspects of life. Thanks to Calan for giving me a bright future to look forward to and the motivation needed to finish this work. Thanks to all my extended family and friends who make life so enjoyable.

Contents

Contents **i**

1 Introduction **1**

1.1 Power Systems Introduction 2

1.2 Cascading Power Failures 10

1.2.1 Northeast 2003 Blackout 11

1.3 Reliability in Power Systems 14

1.4 Thesis Roadmap 16

1.4.1 Overview 21

2 Modeling Cascading Power Failures **27**

2.1 Literature Review 27

2.1.1 Topologic Model 29

2.1.2 OPA Model 33

2.1.3 Power Flow Review 42

2.2 Multi-Stage Stochastic Programming 47

2.2.1 Decision Dependent Uncertainty 48

2.2.2 Cascade Evolution 51

2.2.3 Least Cost Dispatch 52

2.3 Model Flexibility 54
2.3.1 Transmission Expansion 56
2.3.2 Computational Implementation 57
2.4 Conclusion 59

3 Using the OPA Simulation for Transmission Expansion 60

3.1 OPA Cascade Simulation 63
3.1.1 Risk Measures 66
3.1.2 Parameters and Simulation Inputs and Outputs 67
3.1.3 Surrogate Functions 70
3.1.4 Common Random Numbers 71
3.1.5 Capacity Addition 73
3.2 Optimizing Design Problems using Pattern Search 76
3.2.1 Direct Search 79
3.2.2 Line Search Breakpoints 84
3.2.3 Implementing Parallelization in HTCondor 87
3.3 Conclusion 95

4 Line Failure Risk Models for Real-Time Dispatch 97

4.1 Introduction 97
4.2 Economic Dispatch Models 101
4.2.1 Optimal Power Flow using DC approximation 101
4.3 Chance Constraints for Random Branch Flows 103
4.3.1 Random Power Flows 104
4.4 Joint Chance Constraint for System Risk Measure 112

4.4.1 System Risk for Multivariate Gaussian Branch Flows . . . 116
4.4.2 Joint Chance Constraint Model . . . 119
4.4.3 Solution Methodology . . . 121
4.5 Computational Experiments . . . 125
4.5.1 Implementation of the JCC Model . . . 125
4.5.2 Single Instance . . . 125
4.5.3 Sensitivity Analysis . . . 132
4.6 Conclusion . . . 134

5 Reducing Cascading Risk Through Real-Time Dispatch 135

5.1 Cascading Failure Risk Model . . . 136
5.1.1 Overview of Sources of Uncertainty . . . 137
5.1.2 $N-1$ Exogenous Contingencies . . . 138
5.1.3 Random Initial Contingencies for OPA . . . 142
5.1.4 OPA Weighting for JCC . . . 145
5.1.5 Cutting Plane Algorithm for JCC $N-1$ with OPA Weighting . . . 147
5.2 Computational Experiments . . . 149
5.3 Conclusion . . . 151

6 Conclusion 152

6.1 Contributions . . . 153
6.2 Future Work . . . 156

Appendices 164

A Code for Computational Models 165

A.1 Multi Stage Stochastic Program 165

A.2 HTCondor Parallelization Code 165

A.3 Joint Chance Constraint Model 181

A.4 Joint Chance Constraint with OPA Weighting 181

List of Figures

1.1 Location Marginal Prices (LMP) for Midwest ISO's territory 9

2.1 OPA simulation of a cascading power failure 36

2.2 An example of the OPA cascading sequency 38

2.3 Scenario tree for Mixed-Integer Formulation 48

2.4 Example Effective Capacity Distributions 51

2.5 Load Shed Distribution for the OPA simulation and MSIP formulation 53

2.6 Scenario Tree for Two Stage Problem 57

2.7 Load Shed Distribution for the OPA simulation and MSIP formulation 58

3.1 Line failure density function from eq. (3.3) 65

3.2 Risk Measures 68

3.3 Lack of correlation between load shed and first stage line failures 70

3.4 Load Served at Different Stages 72

3.5 Comparing the total load shed for doubling capacity along the coordinate direction. 74

3.6 Plotting expected load shed for capacity additions along coordinate directions. . . . 75

3.7 A cluster of lines correlated with reduced system performance 76

3.8 Expected load shed and conditional value at risk for capacity expansion 77

3.9 DFO methods for continuous variable optimization 79

3.10 Positive spanning sets for $\mathbb{R}^2$ 80

3.11 Generating set for polar cone to conform to local explicit constraints 81

3.12 Polling step with exploratory trial points on a rational lattice 82

3.13 Trial exploration using standard compass search with initial step size 50 83

3.14 Resulting points from compass search with different initial step size values 84

3.15 Line flows for each scenario and their averages 85

3.16 Approximation of function by finding breakpoints 86

3.17 Process flow for parallel OPA evaluations 90

3.18 Folder structure for DAG input in parallelization routine 93

3.19 Folder structure for using DAGs in parallelization routine 94

3.20 Pattern search procedure for transmission expansion design problem 96

4.1 Line failure density function for normalized line flow y' 113

4.2 Line risk substitution for individual branches and possible approximations 116

4.3 Random Power Flows and the Failure Density Function 118

4.4 Reliability frontier for the small test case 128

4.6 A comparison of the three dispatch points for varying risk function parameters . . . 133

5.1 Random variable relationships and sources of uncertainty 137

5.2 $N-1$ Exogenous Contingencies 138

5.3 JCC $N-1$ Risk Model to seed random initial contingencies for OPA 143

5.4 Line failure risk and OPA not correlated 145

5.5 JCC $N-1$ OW correlated with OPA 147

5.6 Load shed distribution 150

List of Tables

1.1 Changing conditions that affect system reliability (from the Northeast outage report [45]). . 16

2.1 Topological measures for the three US power grids 31

2.2 Comparison of Simulation and MSIP outputs 53

3.1 Risk Measures 67

3.2 Data files used in parallelization routine 69

3.3 Scripts and command files used in parallelization routine 88

4.1 Cost and risk results for OPF,CC, and JCC models on the small test case 126

4.2 Generator results using OPF, CC, and JCC models on the small test case. 126

4.3 Time comparison, in seconds, for OPF,CC, and JCC on the large test instances. . . 130

4.4 Risk comparison for OPF,CC, and JCC on the small test instance. 132

4.5 Table of cost for the three dispatch points used in the sensitivity analysis 133

5.1 Rare event risk metrics 150

List of Algorithms

1 OPA cascading algorithm simulating the evolution of the cascading process 66

2 Compass search, a generating set search 81

3 Cutting plane algorithm for joint chance constraint model 124

4 Contingency sampling algorithm for OPA 144

5 Cutting plane algorithm for solving JCC $N-1$ with OPA weighting 148

Listings

3.1 simcplex.in: Parameter definition file 69

3.2 inter-sl5.cmd: HTCondor submit file to run an interactive session on an SL5 machine 87

3.3 CONTROL: commands to run on remote resource 91

3.4 point.cap: Transmission expansion design from pattern search method 95

3.5 point.lsa: Load shed analysis from chosen design 95

A.1 proc.py: HTCondor Queue and Log File Reader 165

A.2 runit: Main Process Flow 166

A.3 power.py: Power Class 168

A.4 tools.py: Common Functions 170

A.5 allocate.py: Direct Search Pattern 175

A.6 consub.py: Build HTCondor Submit Structure 178

A.7 loadshed.py: Take raw load shed and summarize 180

A.8 countLines.py: Take raw line out info and summarize 180

A.9 point.lao: Line outage file from chosen design 181

Chapter 1

Introduction

This thesis focuses on reliability issues for the electricity grid that powers the United States. Electricity is a critical service used by almost every person and company within our country. Reliability issues cost industry billions of dollars every year. Large scale power outages are national disasters due to the loss of services such as cell phones, city wide water pumping, gasoline stations, trains, subways, and cooling which inevitably lead to economic loss and loss of life.

The introduction starts with an overview of the electrical infrastructure of the United States. Following are the basics of how the power system operates on a day-to-day basis and the organizational structure that operates it. Then, the events of August 2003 are explained where part of the Northeast electricity grid collapsed in a cascading power failure leading to millions of people without power, billions in economic loss and the loss of life. This is just one example of a large scale power outage, which is rare, but extremely costly to society. After this, general reliability issues of the power grid is discussed which cost society billions annually.

After defining the problem, this thesis explains the tools and methodology for attempting to make our system more robust against such failures. First, the current literature on cascading

power failures is explored. This is used to model the cascading process as a multi-stage stochastic program with mixed-integer decision variables. It is shown to have a similar distribution to the simulation, however is computationally difficult. We turn to optimizing design problems with the cascading simulation as a subproblem. This allows us to take advantage of accessory information in the cascading simulation as well as parallelize the computational effort to solve large scale problems in a timely manner. Finally, a system risk measure is developed to control endogenous, congestion based line failure risk. This risk measure is used in a real time dispatch model with net injection uncertainty and the cost-risk frontier is explored.

1.1 Power Systems Introduction

The United States electrical infrastructure is a complex physical structure which connects the consumers of electricity with generating assets over a large geographic area. The nature of electricity makes the operation of this infrastructure extremely difficult and is accomplished through numerous organizations and people working together to supply electricity at the least cost while maintaining a given level of reliability.

As of 2004, the electrical infrastructure was comprised of more than $1 trillion in asset value. This includes over 200,000 miles of high voltage transmission lines, 950,000 MW of generation, and 3500 utility organizations serving 283 million people [45]. The introduction briefly looks at generation and the transmission system which efficiently moves the energy over long distances.

Electricity Generation

Electricity is generated using a variety of fuels and procceses. The most common method of electricity generation creates steam by heating water, which spins a turbine producing an alternating

current of electricity. The United States operates its power grid at 60 hz, that is, the direction of electrical flow switches 60 times every second. Since the generators are tied to the grid, when the grid is stable, the generators are rotating synchronously with the power grid. The water can be heated by different fuels such as coal, natural gas, fissioning heavy elements such as uranium, or even using geothermal temperature differentials. The Rankine cycle is a model of converting heat into mechanical energy for steam engines. The efficiency of the cycle is limited by the difference between the turbine entry temperature and the condenser temperature. This means that steam cycle power plants need an external cooling source which removes the waste heat from the working fluid before it begins the cycle again.

The first example of a steam cycle power plant is a coal plant. The chemical energy in coal is converted into thermal energy and byproducts such as carbon dioxide. Power from coal provides around 40% (for the year 2013 [2]) of the electricitty generation in the United States. These plants have more thermal inertia which makes changing the output level a slow process. Modern coal plants achieve efficiencies of 30-40%, that is, the percentage of their input energy which is injected into the grid as electricity is 30-40%.

Nuclear power plants (19% of electricity) also operate on the steam cycle, producing heat by fissioning heavy elements such as Uranium-235. Nuclear plants have relatively slow ramping rates and cheap fuel, which lead them to be dispatched at high output rates continuously. These plants, along with coal plants, provide the majority of baseload power production. Baseload power is the minimum amount of power that needs to be produced continuously throughout the day. Nuclear fuel has a desirable aspect of being extremely energy dense. The energy density of Uranium-235 is roughly 3 million times denser than coal. This means that the waste products of this process are much less than other types of power plants and are also captured completely without being released to the environment.

Natural gas plants (27% of electricity) can capture, in addition to the heat energy, the combustion energy and use it to spin a gas turbine. Gas turbines, while having a lower thermal efficiency, have the desirable trait to quickly throttle production to the desired level in contrast with steam cycles. Additionally, by using a combination of both combustion and heat energy, combined-cycle natural gas plants can reach efficiencies of around 60%. These fast ramping rates and more expensive fuel lead natural gas plants to provide peaking power matching electrical demand over baseload. Recently, with relatively cheaper natural gas and increased efficiencies from combined-cycle plants, natural gas plants are playing more of a role in electrical demand between baseload and peaking levels.

Hydroelectric power (6% of electricity) has many desirable traits and it is the largest source of renewable energy generation installed around the world. By creating a reservoir to hold a large amount of water at a high level, potential energy can be stored. When this water is released, it becomes kinetic energy, which can be captured by a turbine and used to generate electricity. By controlling the flow into the turbine or the amount of turbines spinning, hydro power is capable of not only storing energy, but also quickly adjusting its output rate. However, they have the additional constraint of needing to maintain given reservoir levels throughout the year.

In the past decade, we have introduced a sizeable amount of electricity production from renewable sources such as wind turbines and solar panels (both total 6% of electricity, with wind comprising the majority). These generation sources have a cheap fuel (kinetic energy from the wind and solar radiation), but are unable to control output.

The different characteristics of generators give each a different roles to play in the operation of the power system in order to meet demand. These various generation assets are owned by utilities, independent power producers, large industrial customers themselves, and, more recently, residential

consumers.

Transmission Network

The production from the majority of large generation plants is at lower voltages (10kV - 25kV). Electricity traveling in transmission elements lose energy due to resistive losses, which primarily goes into heating up the power line. The resistive losses are proportional to the current. In order to reduce losses, the voltages are stepped up to between 230 kV to 765kV for long distance transmission elements, which has the effect of reducing the current for a given amount of power. At the demand side, there are radial tree-like distribution networks operating at low voltages (less than 1kV) which connect every demand node to the power grid. The United States power grid is broken into three distinct power grids: Western Interconnection, Eastern Interconnection, and Texas Interconnection. Each has a network of transmission lines connecting all of the generators with all of the loads.

Power flows, according to the laws of physics, along "paths of least resistance", which are modeled with Kirchoff Voltage Laws (KVL) and Kirchoff Current Laws (KCL). This means that electricity flow can't be controlled like many other complex networks such as cell phone and internet traffic, but instead follows laws of physics much like water or gas in pipe networks. In addition, electricity flows at close to the speed of light and currently is hard to store economically, unlike water and gas. There needs to be an instantaneous balance of generation and demand.

Organizational Structure

The primary reliability organization which develops operating and planning standards is North American Reliability Council (NERC) and ten regional councils.

NERC develops standards for reliable operation and planning of the bulk electric system and then monitors and assesses compliance. Also, they provide education and training, while coordinating critical infrastructure protection, such as information exchange between reliability service organizations. Finally, they assess, analyze, and report performance and system adequacy.

The primary focus of the reliability and planning standards is to be able to serve all demand reliability both today and in the future. There are 7 primary tasks:

- Balance power generation and demand continuously;
- Balance reactive power supply and demand to maintain scheduled voltages;
- Monitor flows over transmission lines and other facilities to ensure thermal limits not exceeded;
- Keep system in stable condition;
- Operate so that it remains in reliable condition even if contingency occurs (N-1 critiera) and when a contingency does occur, maneuver to new stable N-1 position. The N-1 criteria states that the system must be robust against any single component of the power grid failing;
- Plan, design, and maintain the system to operate reliably; and
- Prepare for emergencies.

Within the United States, Federal Energy Regulatory Commission (FERC) is the federal agency with control over electricity sales, natural gas and oil pipelines, and hydropower projects and has increased power after the Energy Policy Act of 2005. This differentiates it from NERC in that it can impose mandatory standards.

FERC imposes mandatory reliability standards for the bulk transmission system and imposes penalties on those that manipulate the markets. Its primary tasks are the following:

- Regulate transmission and wholesale electricity sales in interstate commerce;
- License non-federal hydroelectric projects;
- Ensure reliability of high voltage interstate transmission system;
- Monitor and investigate energy markets;
- Penalize violating entities, through civil penalties or other means; and
- Oversee environmental matters relating to major policy initiatives.

The restructuring of the power grid has decoupled utilities from the responsibility of maintaining a control area. The 140 control areas are operated by the 10 regional Independent System Operators (ISOs) or Regional Transmission Organizations (RTOs).

ISO/RTO are tasked with providing least cost energy to everyone within its territory while maintaining a given level of reliability. In order to do this, they create wholesale electricity markets in order to balance generation and load in real time at least cost. Their tasks are the following:

- Manage the wholesale electricity markets while maintaining the reliability of the system;
- Direct the operation of the assets owned by their members; and
- Encompass multiple control areas within the territory that they operate.

Wholesale Electricity Markets

The ISO or RTO use 3 primary markets for determining how much each generator should produce at any given time. The first market is bi-lateral contracts. These are long term contracts between producers and suppliers to exchange a certain quantity of energy at a given price. These contracts help provide reliable, long term forecasting for supply and load. This reduces exposure to the volatility of the day ahead and real time markets.

The day ahead market takes bids from generators which contain an array of costs to produce a certain amount of electricity. The generators also provide additional operational constraints, such as ramping rates and start-up and shut-down times and costs. In addition to generators' physical constraints, forecasts for demand of the following day is used to get an estimate of the amount of generation needed. Using the transmission constraints for the given network, the ISO clears this market during the previous day. This process usually takes around four hours and is done with specialized optimization tools solving the optimal power flow problem with security constraints and unit commitment decision variables. The outcome of this process is the schedule for slow ramping generators and hourly locational marginal prices (LMP) at each location in the network for the following day. These prices are guaranteed and any deviation from the schedule are made up in the real-time market.

The real-time market makes up for errors in forecasting as well as possible failures in generation or transmission. This market operates on five minute intervals and can be extremely volatile. Similar methods as the day-ahead are used to clear this market, although simplified due to the time constraint. Price spikes can occur during times of peak demand and a congested grid in which the price of electricity can often become over 10-20 times as expensive. The reverse is also possible. During times of low demand, it is possible for negative prices on the real time market. This occurs

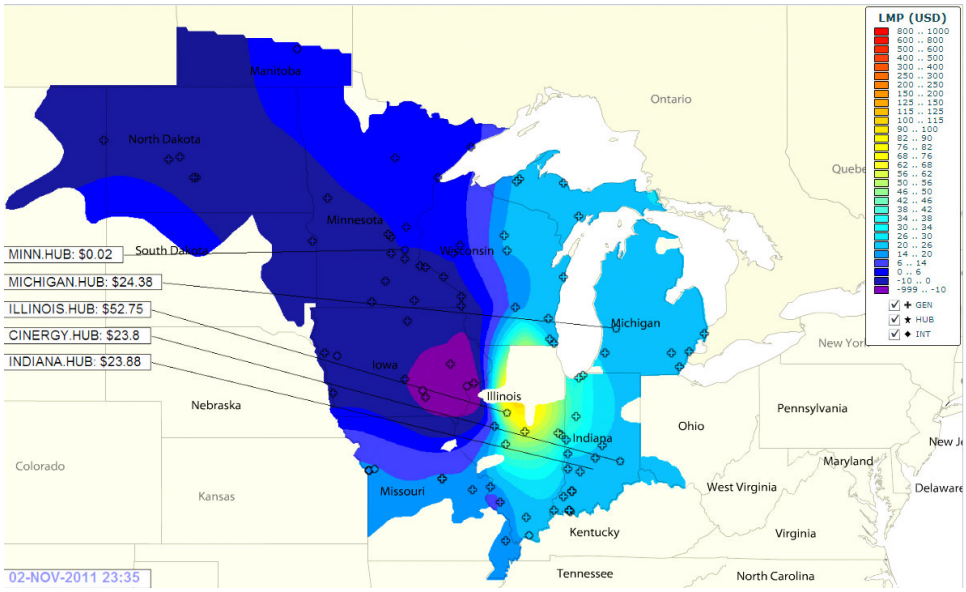


Figure 1.1: Location Marginal Prices (LMP) for Midwest ISO's territory

because the generators are constrained by ramping rates and there is too much production on the grid. In fig. 1.1, there are negative prices across the Midwest and, in particular, Iowa, with much higher prices in Illinois. This LMP snapshot is taken at night when there is low demand for electricity as well as increased production from wind farms located in the Midwest. The differences in prices suggest significant transmission bottleneck constraints between the Upper Midwest and Illinois.

Operating Constraints

In order for the power grid to operate and maintain stability, a number of things must hold. First and foremost, there must be a continuous balance of power generation and demand. The US grids are all operated with a target frequency of 60 hz, but the actual system frequency varies. When there is excess generation on the grid, the frequency will increase and with a lack of generation, frequency will drop. Since the generators are synchronized with the grid, when the frequency deviates from normal, the generators can move out of their operating limits and cause damage.

Generators have internal protections to identify bad operating points and trip offline in order to protect themselves. In order to protect the grid, there is automated tripping of load at certain frequency points to take customers offline and prevent total collapse.

In addition to real power balance, reactive power supply and demand must be used to maintain scheduled voltages. Low voltage can cause system instability and damage to motors and electrical equipment. Also, high voltage can exceed insulation capability and cause dangerous arcs. Reactive power is supplied through capacitor banks and generator output.

In order to protect the transmission elements, the flows over transmission lines and other facilities must be monitored to ensure thermal limits are not exceeded. Lines are heated by electricity flow and equipment can be damaged, such as conductors sagging from stretch and expansion due to high temperatures. These problems are affected by ambient temperature and wind conditions. The flow is limited so the line does not sag into obstructions such as trees and telephones.

1.2 Cascading Power Failures

A cascading power failure is a set of failures of individual components of the transmission system which leads to the redistribution of power over the new topology. When the system is in a stable operating position, individual components often have a negligible effect on system-wide distribution. However, after several failures, the transmission network may not have enough capacity to distribute the power from the generation to the load. This can cause a series of fast acting trips in which a large portion of the grid may fail. While rare, these events are extremely costly.

1.2.1 Northeast 2003 Blackout

Historical Problems

The Northeast blackout of 2003 began in the Cleveland-Akron area of the Eastern Interconnection. This area had a history of problems of low voltages due to the relatively high amount of imported power. In 1998, the system was becoming unstable and everything other than load shed was done to fix the problem. It was shown to not be within the reliability standards, and regulations were loosened instead of addressing the problem. A transformer problem led American Electric Power (AEP), a control area entity, to perform a reliability study on the neighboring control area operated by First Energy (FE). The study again showed voltage instability problems.

The summer of 2003 was fairly typical with less than historical peak load. While the load was less than historal peak, it was consistently greater than the forecasted load. High temperatures, creating a greater air conditioning load, contributed to reactive power shortages. The voltages were low in Cleveland-Akron during the week, but within the new operating limits (92% to 105%). A group of shunt capacitors, out of service for planned maintenance, further reduced reactive power supply. Also, a large nuclear reactor, Davis-Besse, was in a long term forced outage state. This plant is normally able to provide a large amount of real power as well as reactive power support.

Normal Day Turns Bad

A series of initiating events, as well as the failure of FE's control server, led to an unstable system and the inability for operators to do anything about it. In addition, neighboring regional reliability coordinators (Midwest Independent System Operator, MISO, and Pennsylvania-New Jersey-Maryland Interconnect, PJM) had bad information on the state of the system and were unable to provide support. The operators had concern about voltage levels and were trying to get

additional reactive power online. The shunt capacitors were unable to return to operation.

The following initiating events didn't cause the system to collapse, but moved it into an unreliable state, which allowed the collapse to take place. The Stuart-Atlanta 345 kV line tripped due to contact with surrounding vegetation (area voltage at 98.5%) at less than 100% capacity. In addition to the loss of the line, MISO was unaware, which led to unusable output and the inability to provide support. An important generating unit, Eastlake 5, attempts to increase reactive power output, but the internal protection trips the generator offline (area voltage 97.3%). This led to real power imports rising, which increased the need for reactive power as well. Another transmission line loss of Harding-Chamberlin 345 kV continued to depress the voltage (95.9%). Around this time, the underlying 138 kV network began to fail. MISO was unable to perform N-1 contingency analysis and the FE reliability charts flatlined due to the server problems.

Ultimately, it was in reliable state before 15:05, however, after all of these events, the system was no longer N-1 stable. In addition, reliability coordinators were using separate state information due to poor communication.

System Becomes Unreliable

A total of 3 lines tripped at below operating capacity due to tree contact starting with Harding-Chamberlin (44% of capacity) at 15:05. Then the Hanna-Juniper line tripped (88% normal and emergency rating) at 15:32. The Star-South Canton line had multiple tree contacts (55%) starting at 14:27, and finally tripped off for good (93% emergency rating) at 15:41. There were no prior sustained outages from tree contact in the previous years and FE used vegetation management practices consistent with the industry. When designing line limits, the thermal ratings are based on many variables including ambient temperatures and wind speeds. A combination of higher than

nominal ambient temperature and lower wind speeds reduced the cooling capacity of lines.

Perry plant, the largest local generator, was getting voltage spikes at levels close to tripping the generator. They notified FE of the problems, but the operators were unable to recover from this precarious situation. The task force notes that load shed here of 1500MW may have saved the system by increasing voltages and stopping the ultimate trip of Sammis Star line.

High Speed Cascading Failures

At 16:05 the final straw was pulled by the tripping of the Sammis-Star 345kV line separating geographical regions of the grid. Unplanned power shifts across regions caused phase 3 operations in the protective relays of transmission elements, which can trip lines far away from the problem area. The grid continued to stabilize after each one of these individual outages. However, by 16:10, the north and south separated (AEP separates from FE), and the high generation area was no longer directly connected to the high load center. This caused a massive power surge from PJM through New York and Ontario counter clockwise around the great lake to load centers in Michigan. This surge caused the Northeast to separate from the rest of Eastern Interconnect (EI).

There was insufficient generation in the newly formed islands to support the load. The frequency in remaining parts of EI rose to 60.3%, representing 3700 MW of excess generation. The Northeast grid kept breaking apart until islands were formed in which an equilibrium between load and generation could be made. Under-frequency and under-voltage load shedding helped to stabilize the system within the islands. New York dropped 10,648 MW through automatic load shedding and Ontario dropped 7800 MW. Some generators tripped off at unreasonable levels making island stabilization more difficult. Thousands of events occured between 16:10 and 16:13.

The cascade spread, not due to voltage problems, but dyanmic power swings which caused

system instability. The voltage instability was only companion to the angle instability (represents large real power flow swings) which tripped generators and lines. The large power swings come from imbalance between generation and load across regions and the electrical separation of these two areas. The inherent weak points are lines with the highest impedence, which trip off relatively early due to protective relay settings. These are typically long over-head lines with high loadings.

The Blackout Results

In the United States, 45 million people lost power totaling 61,800 MW in Ohio, Michigan, Pennsylvania, New York, Vermont, Massachusetts, Connecticut, and New Jersey. The US loss estimate was between $4 billion and $10 billion. Another 10 million people from Canada lost power leading to an estimated 18.9 million lost man hours and $2.3 billion in lost manufacturing shipments. There were at least 11 fatalities, power took up to 4 days to return, and rolling blackouts continued in Ontario for the following week.[45]

The formation of a large island, based off of hydro plants in western New York and Canada, was the basis for system restoration.

1.3 Reliability in Power Systems

The electricity grid is held to a higher reliability standard than other services since it is critical infrastructure for society. Even though the power system maintains a very high level of reliability, power interruptions still have large costs to consumers. After a discussion of the economic costs, the trends in power outages for the North American grid are looked at.

The economic loss caused by power interruptions to electricity consumers in the United States for 2001 was estimated at $79 billion [30]. This can be compared to the total revenue of

electrical sales in the year 2001 of $247 billion [1]. The single event of the Northeast blackout in 2003 was estimated to have a total impact on US workers, consumers, and taxpayers as a loss of approximately $6.4 billion [4]. This cost is hidden from the system, but may entertain the notion that the true cost of electricity is higher than the current prices.

The current power system has evolved throughout the last century due to economic and reliability issues and the responses of the operating entities to these forces. The power system has self-organized into a dynamic equilibrium where blackouts of all sizes occur [18]. The average frequency of blackouts in the United States is 13 days and has been the same for 30 years [12], which represents this equilibrium.

NERC data shows that distribution of blackouts for the last 15 years (1988-2003) follows a power tail curve with an exponent of around $-1.3 \pm .2$ [12]. This is strengthened further where Hines *et. al.* showed that the frequency of blackouts in the United States is not decreasing, changes seasonally and with the time of day, and follows a power-law distribution [23, 24].

Some emerging conditions on the grid may make protection more important and more difficult, shown in table 1.1. Electric vehicles adopted en masse have a potential to offer useful services to the power grid through the use of a sizable amount of energy storage in aggregate. However, they also represent a considerable stress on the system in different spots and ways than it is used to. The ideal car battery system would have a quick charge, that is, the ability to transfer the energy from the grid to the battery extremely quickly. This would allow consumers to use charging stations similar to the current gasoline stations for combustion engines. A charging station capable of charging many cars would have extremely high and unpredictable volatility that the system needs to protect against.

The increased penetration of renewable energy production also increases stress on the power

Previous Conditions	Emerging Conditions
Fewer, relatively large resources	Smaller, more numerous resources
Long-term, firm contracts	Contracts shorter in duration, more non-firm transactions, fewer long-term firm transactions
Bulk power transactions relatively stable and predictable	Bulk power transactions relatively variable and less predictable
Assessment of system reliability made from stable base (narrower, more predictable range of potential operating states)	Assessment of system reliability made from variable base (wider, less predictable range of potential operating states)
Limited and knowledgeable set of utility players	More players making more transactions, some with less interconnected operation experience; increasing with retail access
Unused transmission capacity and high security margins	High transmission utilization and operation closer to security limits
Limited competition, little incentive for reducing reliability investments	Utilities less willing to make investments in transmission reliability that do not increase revenues
Market rules and reliability rules developed together	Market rules undergoing transition, reliability rules developed separately
Limited wheeling	More system throughput

Table 1.1: Changing conditions that affect system reliability (from the Northeast outage report [45]).

system. Wind and solar are both variable generation devices which do not actively control the amount of electricity being produced. The system needs to maintain ample ramping capability in order to maintain stability. These stresses can be reduced by improving the quality of forecasting efforts on various timescales. In addition, these technologies are geographically constrained by their fuel source availability, wind speeds and solar irradition. However, these natural fuel sources do not align with large demand centers, so the transmission system is needed to efficiently distribute the energy produced.

1.4 Thesis Roadmap

In this thesis, we work to evaluate and reduce the likelihood and effects of rare event failures on the bulk power system. We start by reviewing the current literature on cascading power failures and we use a cascading simulation common in literature to build upon. We first formulate this as a large, multi-stage stochastic program using binary variables to model line failures and an effective capacity distribution to model decision dependent uncertainty. Then, we decompose this

model by scenario in order to get an efficient and parallelizable cascade simulation. We look at transmission expansion and optimize over this long term design problem using derivative free optimization techniques. Then, we look at line failures in the five minute time horizon and develop a dispatch model to constrain the likelihood of line failures over this time frame. We extend this model to the N-1 exogenous contingencies and combine it with the OPA model to evaluate and constrain rare event risk.

The cascading power failure simulation we use, and is used in other literature, can be seen as an empirical model in which the distribution of load shed from these events follow the same power-law distribution that is seen in the real world. It uses topology information as well as linearized power flow constraints that are necessary to capture some of the important effects of power system operation. This model does not attempt to capture short term transients involved in the physical evolution of these cascading processes. Instead, it attempts to capture the distribution of possible outcomes and evaluate the system risk associated with rare events.

We have three primary themes and areas of contribution. First, a common theme seen in this thesis is that instead of trying to deal with these events as they are occurring, we want to reduce the likelihood of initiating events and the expected effects of them once they are started. Since it is often hard to tell at the beginning if a cascade will occur and once it is obvious it is likely too late, we want to reduce the chance to be put in that situation and minimize the expected effects of that situation. In chapters 2 and 3, we look at transmission expansion where the objective function is to minimize the expected value of load shed over a subset of contingencies. In chapters 4 and 5, a system risk constraint is developed that constrains the likelihood of line failure due to endogenous, line loading risk.

The second theme is the multiple sources of uncertainty. The first primary source of uncer-

tainty, that has been a focus of a lot of recent literature, is demand and generation uncertainty. With the increased penetration of solar and wind generation, the power system experiences more deviation from forecast than before. We focus on this demand and generation uncertainty in chapters 4 and 5. Another source of uncertainty is what we term as the effective capacity of a transmission element. It is hard to know exactly how much current can flow on a line before it fails. This failure point is the result of many factors: environmental conditions such as ambient temperature and wind speed, clearance between the transmission line and vegetation or infrastructure, material characteristics of the conductor, and others. Since this level depends on environmental conditions that are continually changing and other unknowns such as vegetation growth, instead of trying to calculate this exactly, we assume that this is uncertain. As we saw in the Northeast 2003 blackout, multiple, critical transmission elements failed at below nominal capacity due to environmental conditions and vegetation growth. Our effect capacity distribution builds this uncertainty directly into the model. We focus on this effective capacity uncertainty in chapters 2 to 5. The final source of uncertainty is the evolution of the cascading process once it has been initiated. We focus on this uncertainty in chapters 2 and 3. We use a cascading model seen in literature that captures some of these effects, however as more detailed and accurate evolution models are developed, they can be incorporated into this framework as well.

The final theme is the multiple time frames at which we attempt to tackle this problem. Power systems have a lot of infrastructural inertia in that there are high capital costs and long lead times for power system assets. Additionally, there has not been much investment in the transmission grid over the last 30 years, however there is increased interest in this topic as of late. In chapters 2 and 3, we are looking at this problem from a long term design perspective. If we have a budget to increase the resilience of the power system to these rare event failures, how should we allocate it among the potential infrastructure options. In chapters 4 and 5 we look at the real time dispatch problem

and incorporate the endogenous system risk of line loading levels into the dispatch decision. These models do not require infrastructure investments, instead they change how the generation units are dispatched. For each dispatch, there is a resulting power flow through the transmission elements. This flow adds to line failure risk, and it is this endogenous risk associated with power flow that we evaluate and constrain in our new dispatch models.

Multi-Stage Integer Program

In chapter 2, we give an overview of current literature on cascading power failures. We follow work on a cascading simulation used in literature that is a short term force in an equilibrium model. This cascading simulation is modeled as a multi-stage stochastic program with mixed integer decision variables (formulation donated as MSIP). The primary contributions of this chapter are the following:

- Describe sequential cascading process and its greedy characteristics mathematically;
- Model decision-dependent uncertainty using binary variables and *a priori* sampling, introducing the concept of *effective capacity*; and
- Formulate the cascading simulation as a MSIP and use this as a subproblem in long-term design optimization aimed at mitigating cascade-induced load shed.

Derivative Free Optimization

In chapter 3, we use a computationally cheaper simulation, common random numbers for variance reduction, and a parallelization routine to optimize capacity expansion problems using the OPA cascading simulation as an evaluation of rare event risk. To optimize over this simulation with high

frequency noise and discontinuities, derivative free optimization (DFO) techniques were employed. The primary contributions of this chapter are the following:

- Explore the effects of capacity expansion on the OPA cascading model;
- An efficient parallel implementation to optimize the simulation on large-scale instances; and
- A Modification of DFO techniques that take advantage of problem-specific characteristics.

Joint Chance Constraint

In chapter 4, we develop a dispatch model to incorporate the endogenous risk from line loading. A system risk measure is controlled via a joint chance constraint (JCC) on the probability of any line failing and constrained under net injection uncertainty. The primary contributions of this chapter are the following:

- Calculation of branch flow covariance matrix under the assumption of a known injection covariance matrix for generation and demand;
- System risk defined by probability that one or more lines fail formulated analytically as a JCC;
- Solve the constrained system risk dispatch model exactly when generation and demand are fixed (not random, known with certainty);
- Solve the constrained system risk dispatch model approximately using the assumption of multivariate Gaussian uncertainty; and
- The cost-risk frontier is explored under this new risk measure and the resulting economic trade-offs.

Joint Chance Constraint with OPA Weighting

In chapter 5, we combine the rare event risk of the OPA cascade simulation with the line failure model developed in chapter 4. Since all lines are not created equally, this model accounts for differences in line importance by weighting the line risk constraint. The primary contributions of this chapter are the following:

- System risk measure is extended to exogenous contingencies ;
- Evaluate rare event reliability with the OPA model; and
- Use a linear weighting system to account for OPA risk in a real time dispatch model.

1.4.1 Overview

Cascading Power Failures

One type of cascading model is a strictly topological model with no power flow elements. While these types of models are simple, they are incapable of capturing effects which are loading dependent. The OPA type model (detailed in section 2.1.2) is a sequential process in which the power flows for the entire network are calculated in order to determine the loading on the various transmission elements. This, in addition to a deterministic or probabilistic failure model, is used to determine whether individual components fail at each stage in the cascade. The cascade concludes when no transmission elements fail.

The OPA type of model has been shown to capture the criticality effects of blackouts (higher than expected large blackouts) we see in real power systems in aggregate. As more complexity is added into this model, this type of model can produce reasonable sequences of line failures for the system, while remaining accurate in aggregate. In this thesis, we use the simple form of the OPA

model in order to capture loading effects, however we are ignoring all effects related to voltages and high speed transients. Our high level models can incorporate some of these more complicated features, however we use the OPA model as a good starting point so that we are not bogged down in details, although they are certainly important details that need to be worked on.

Building this model requires calculating the power flows. While the system is not necessarily stable or balanced throughout the cascade process, the balanced, steady state 3 phase power flow equations are used as a basis for the power flow analysis. The power flows is approximated by using the DC (Decoupled) power flow model, which is a linear approximation to the AC power flow equations. We do not use information about voltages, however there are OPA models that use the full AC power flow equations that could be incorporated into our models with some effort.

Design problems are formed that change aspects of the topology, component parameters, or operation of the system. Using the OPA model to gauge the response of the system to these changes, we have formulated a mathematical optimization problem. These design problems can range from transmission expansions to setting operational reserve levels or even producing protective relay settings for individual components. A primary contribution of chapter 2 is the demonstration of the significant flexibility offered by the mathematical optimization models.

Optimization Difficulties

This problem offers several difficulties which make optimization challenging. A primary difficulty is the sequential process of the OPA model in which operators make decisions under uncertainty and the decisions effect the future stages and progression of the process. In addition, the decision-dependent outcomes only have a probabilistic effect on the progression of the cascade. That is, an overloaded line may not fail in all scenarios, so that at each stage a range of outcomes is needed to

capture the probabilistic effects. In order to properly describe the outcomes of the process under a range of scenarios, a multi-stage tree with even a modest number of stages and outcomes explodes quickly.

Another main challenge is that the change of topology of the power system creates nonlinear and nonconvex effects. It is possible to reduce the cost of electricity by removing transmission elements. On the other hand, by adding transmission lines, a system capable of meeting demand can become infeasible. Since the system is constrained by KVL and KCL, each transmission line, while providing a path for electricity flow, further constrains the system. It is well known that nonconvex behavior is not desirable for optimization problems, and in this case we can have nonconvex behavior between each stage of our process, since by definition, each stage before the final has topology changes.

Modeling Cascading Power Failures

In order to better understand the structure of the problem, it is formulated as a multi-stage stochastic program with decision-dependent uncertainty (MSIP) in chapter 2. In order to do this, logical connections between the stages are made using binary variables, that is variables that can only take the value 0 or 1. In this way, the decision dependent uncertainty can be modeled explicitly and the probabilistic effects can be modeled by sampling the random variables *a priori*. As the number of outcomes per stage increases, probabilistic effects can be modeled more accurately. Using this type of model, the number of stages has to be decided *a priori* as well. This is a major shortcoming, as we are concerned with the effects of the worst cascades, which can take many stages to complete. As the number of nodes, N, is related to the number of outcomes, a, and number of stages, b, by $N \propto a^b$, the model quickly becomes intractable.

However, if a crude approximation to the cascade process can get similar aggregate effects from the outcome, we are able to use this formulation as a subproblem. This can then be embedded in design master problem, an example being transmission expansion, and then solved with an out of the box mixed-integer solver.

Exploring the OPA Simulutation Through Transmission Expansion

The computational complexity of the first approach grows too fast with respect to increases in model accuracy. In chapter 3, the OPA model is simulated instead of more rigorously mathematically defined. In this, we lose some structure of the problem, but by decoupling the stages of the OPA model from the master problem, we have a simple Monte Carlo simulation. Using variance reduction techniques of common random numbers we can resolve the outcome of any configuration of the design problems extremely quickly by running a large number of simulations. In addition, we can parallelize the process in order to evaluate multiple configurations simultaneously. The outcome of these simulations are statistics about the magnitude of the blackout for the various contingencies as well as possible sequence of line failures.

The optimization field has many algorithms that can use a zeroth order oracle, that is for each configuration, the oracle returns a real-valued number (the function value) and may include stochastic variation or numerical error. In our case, the simulation is the oracle and the real-valued number could be the expected load shed or the value-at-risk (which attempts to capture large tail effects of distributions). The main types of derivative free optimization are pattern search, model based, and stochastic approximations.

A pattern search method doesn't attempt to understand anything about the underlying structure but instead tries to search in a specific way such that the function value improves. Certain

patterns can offer local convergence guarantees, that is the solution is at least better than all of those in the neighborhood. This, in combination with a grid search in which the whole space is partitioned and sampled, can get you good global solutions, but global convergence is not provable. These search methods can be improved by using well-positioned exploratory steps and search directions that conform to local topology.

Line Failure Risk Model for Real-Time Dispatch

In chapter 4, we consider a new dispatch model. The model incorporates endogenous line loading risk and constrains it according to some risk preference. This creates an efficient operating frontier along the cost-risk metrics for different operating points. In addition, this model incorporates net injection uncertainty into the evaluation of the system risk measure. Included in the system risk measure is the covariance of the net injection uncertainty. This uncertainty is responsible for some of the system stress due to volatile injection and the system responds to these volatile injections with a subset of generators following an area control error (ACE) signal. This subset of generators is denoted the slack distribution in this thesis.

We begin by modeling the net injections as a Gaussian distribution with a given mean and covariance matrix. Due to our linear assumptions for DC power flow, the linear power transfer distribution matrix can be used to calculate the branch flows (also Gaussian) mean and the branch covariance matrix. A line-failure function is used to model the risk on a single line for a given flow. This can be integrated over the uncertainty in line flow to find the individual contribution of risk for any line. By summing over the lines, we find the expected risk to the system depending on the specific line loadings. This system risk is constrained in a real time dispatch model.

The JCC model is explored for its effects on the generator dispatch points in a small 30 bus

example. The model is also used on a 2300 bus example to show that it is quick to solve (around 2-4x solve times compared to standard DC, although roughly similar as standard chance-constrained model). The cost risk frontier is explored and the chance constraint frontier is plotted, which resides on the interior of the new frontier. The chance constraint model comes from several papers in this research area and puts a chance constraint on the probability of a line to exceed its threshold. With proper parameters, the JCC model can closely represent the same effects as the chance constraint model while having the additional benefit of a system risk measure, instead of a risk measure for every single branch element.

Reducing Cascading Risk Through Real-Time Dispatch

Finally, we conclude by incorporating the rare event risk measure of the OPA cascading process with the real-time dispatch model of endogenous line failure risk. We extend the joint chance constraint model to the N-1 exogenous contingencies. We also use these N-1 exogenous contingencies and the line failure risk model to seed the OPA model with initial contingencies. Using these two sources of information, we develop a linear weighting system to capture the first order effects of the initial contingencies in the OPA process. We use this linear weighting and the cutting plane algorithm from chapter 4 to solve this large, convex problem.

Chapter 2

Modeling Cascading Power Failures

An ideal model is complex enough to capture the important features and effects of the process, while remaining simple enough to work with and remain believable. Arbitrarily complex models can be made to produce the output matching nearly any process, but often there is no reason to believe that the individual mechanisms being modeled are important or represent a real part of the process. In this chapter, we review a simple simulation model for estimating the load shed from a cascading power failure. We analytically capture the relevant features of the simulation model into a mixed-integer stochastic program (MISP) with decision-dependent uncertainty. Our MSIP can be embedded in a variety of optimization problems designed to mitigate the effects of cascading power failures.

2.1 Literature Review

Historically, cascading power failures have been a hard problem to understand, and since they are rare, the process was not studied much. However, after the Northeast Blackout of 2003, more focus has been devoted to this problem.

The simplest models of the cascading process are strictly topological models with some mechanism to fail components such as a failed node leads to its neighbors to fail with a given probability. While these models can capture interesting effects due to degree distribution and clustering, they lack important information about the electrical parameters of the topology as well as loading patterns.

After discussing historical topological models, Hines, Cotilla, *et. al.* [25], [15] provide a rebuttal to strictly topological models. Here, topological measures are looked at in addition to the electrical information behind the topology of the grid.

Finally, the Oak Ridge National Laboratory, Power System Engineering Research Center of Wisconsin University, and Alaska University (OPA) type models are developed. These models use the electrical parameters as well as demand and generation information to solve the power flow problem. The results of this are loading patterns on the topology of the power grid. Using these loading patterns as well as a deterministic or probabilitic failure mechanism, another level of complexity can be added to the cascading failure model. The following quote from Hines et. al. [26] supports the use of this type of model.

> While a perfect model of cascading failure would accurately represent the continuous dynamics of rotating machines, the discrete dynamics associated with relays that disconnect stressed components from the network, the non-linear algebraic equations that govern flows in the network, and the social dynamics of operators working to mitigate the effects of system stress, all power system models simplify these dynamics to some extent. Unlike simple topological metrics, our model does capture the effects of Ohm's and Kirchhoff's laws, by using linear approximations of the non-linear power flow equations [5]. Similar models have been used to study cascading failure in a number of recent

papers [12], [19], [35].

In addition to the fast time scale cascading model, the OPA work included a slow time scale model, which responded to these cascading failures by engineering improvements. It is the dynamic interactions of these two forces that leads to an equilibrium in which blackouts of all sizes occur and the size follows a power law distribution.

In order to use OPA models, power flows need to be calculated and to reduce the computational complexity of the model, a Decoupled (DC) power flow model is used by making a few simplifying assumptions. Using the power flow model, an economic dispatch model can be made to dispatch the generators at least cost, while remaining within its operating constraints. This dispatch model is used to connect the reliability issues of the power grid with economic ones.

2.1.1 Topologic Model

This section first discusses historical topological models and then describe common topological measures. These measures are used to compare to other network structures. Using these measures, it is shown that power grids differ from many other common network structures and thus need to be analyzed on their own.

Historical Topological Models

In 2004, Albert et. al. [3] worked on large blackouts in response to August 2003 and developed a deterministic model of failures based on topological measures. They used 4 methods of removing nodes from the grid one at a time, randomly, highest degree, highest load, and cascading. The main simplifying assumption is that if a generator is connected to a load, its power is available. In addition, power is routed along the shortest path from generation to load. Then, in order to monitor

the effects of the failure patterns, connectivity loss is recorded, which represent the average decrease in number of generators connected to a substation. They conclude by noting possible solutions of increasing redundancy and capacity of the system or decreasing reliance on transmission by using more generation at the distribution substation level.

Kinney and others developed another method for estimating power flows on a given topology [28]. They introduce the concept of efficiency for power lines and use the harmonic composition of the efficiency of lines to calculate an efficiency measure for any given path. Now, the electricity is distributed to a load from a generator along the most efficient path. Then, they modeled an efficiency degradation based on loading through time as well as tolerance measures to fail lines probabilistically.

A handful of these topological models with flow estimates were done in between 2003 and 2010. Some were predicated on behaving like other networks such as scale-free networks [55], [40] or small-world networks [17]. Others used matching models with a profit function to protect against cascading failures [43] or novel recourse strategies such as deliberate weak lines for network islanding [20].

Topological Measures

The topology of a power grid can be described as an unweighted, undirected graph $\mathcal{G}$ with vertices $\mathcal{V}$ and edges $\mathcal{E} \subset (\mathcal{V} \times \mathcal{V})$ that connect the vertices. A particular grid is denoted by a subscript, such as $\mathcal{G}_{EI} = \{\mathcal{V}_{EI}, \mathcal{E}_{EI}\}$, which would be the graph that represents the Eastern Interconnect. The vertices $\mathcal{V}$ on the graph can represent demand nodes, generator nodes, and buses in the transmission network. The edges $\mathcal{E}$ represent elements such as transmission lines and transformers. For convenience, we define $n_v = |\mathcal{V}|$ and $n_e = |\mathcal{E}|$.

Grid	Vertices	Edges	$\hat{k}$	k_{max}	C	L	d_{max}
$\mathcal{G}_{EI}$	41,228	52,075	2.53	29	0.068	31.9	94
$\mathcal{G}_{WI}$	11,432	13,734	2.4	22	0.073	26.1	61
$\mathcal{G}_{TI}$	4,513	5,532	2.45	18	0.031	14.9	37

Table 2.1: Topological measures for the three US power grids

A useful tool for describing topological measures on graphs is the adjacency matrix, A. The elements a_{ij} of A represent whether nodes i and j are connected, such that if $(i, j) \in \mathcal{E}$ then $a_{ij} = 1$, else $a_{ij} = 0$. The degree k_i of a node measures how connected it is to the rest of the network, with

$$k_i = \sum_{j=1}^{n_v} a_{ij} \tag{2.1}$$

A common measure to compare different graphs is the average degree $\hat{k} = 2n_e/n_v$. Cotilla et. al. [15], using data for EI from NERC in 2012 and data for WI and TI from FERC in 2005, found the average degree of the networks, which is given in Table 2.1. This tells us that our power grids are sparsely connected, with around 2.5 transmission elements connecting each vertex, noting that parallel lines are counted as one. The following statistical analysis of topological measures was done by Cotilla et. al. [15] in order to show that power grids are neither small-world networks nor scale free networks.

There are many statistical measures used to compare our power grid graphs to other common graph structures. The first measure is the distribution of the degree, k_i, of all the nodes. One type of network to compare to is a scale-free network which have a power-law degree distribution. These networks have highly connected central hubs, which are inherent weak points to the network. However, high-degree nodes are far less common in power grids than would be expected with a scale-free network.

Two additional measures, which are distance metrics, are diameter and characteristic path

length. The distance d_{ij} between i and j is the minimum number of links needed to traverse from vertex i to vertex j. The diameter is then

$$d_{max} = \max_{ij} d_{ij} \tag{2.2}$$

and the characteristic path length is

$$L = \frac{1}{n_v(n_v - 1)} \sum_{\forall i,j | i \neq j} d_{ij}. \tag{2.3}$$

In addition, the average nodal distance $\hat{d} = \sum_{j=1}^{n_v} d_{ij}$ can be used. As the size of small-world networks increase, the characteristic path length increases roughly with $\ln n_v$, which means the distances between vertices grows slowly. However, the power grid's path length always grows faster than $\ln n_v$ and falls between small-world networks and regular grids, which scale linearly with n_v.

Another useful measure is the cluster coefficient which gives insight into neighborhoods of nodes. Let e_i be the number of edges connected to vertex i and its immediate neighbors N_i by the following $e_i = \sum_{\forall j,k \in \{N_i \cup i\}} a_{jk}/2$. Then the clustering of node i is

$$c_i = \frac{e_i}{(k_i(k_i - 1))/2} \tag{2.4}$$

and the cluster coefficient of the graph is $C = \frac{1}{n} \sum_{i=1}^{n_v} c_i$. Power networks were found to have less clustering than small world networks but much larger than random grids, which may be due to relatively few long distance lines.

The final measure used was degree assortativity, which is the correlation of the degree of two connected nodes. Power networks were found to have small, negative degree assortativity. This was due to distribution feeders, which have a large number of radial lines connecting single loads

to the substation. This behavior was not found in small-world networks.

Hines et. al. [25] conclude that while these topological measures are useful for understanding the structure and perhaps indicating general vulnerabilities, they can lead to erroneous conclusions. For example, Kinney et. al. [28] and Albert et. al. [3] draw different conclusions about such things as the effects of single failures using similar data. Power flow based models are more realistic and thus more useful for vulnerability analysis. However, analogous measures with electrical topological information can be very useful.

2.1.2 OPA Model

The next level of complexity to add when understanding power grids is to use electrical information about the grid as well as loading patterns to determine the power flows. The loadings on particular elements have a large effect on the failure probability of the given element. This type of model was done by three groups, Oak Ridge National Laboratory, Power System Engineering Research Center of Wisconsin University, and Alaska University. This class of models are called OPA models. They look at the power transmission system and consider engineering and physical aspects, as well as economic, regulatory, and political responses to blackouts and increases in load.

In 2001, Dobson et. al [18] found that the opposition of the slow time scale force of growth in load and system capacity and fast time scale of cascading power failures produced a dynamic equilibrium that can be seen in real world data. Many real world complex systems can be seen to have this self-organized criticality property. This criticality means that the blackout size distribution follows a power-law distribution, $f(x) = ax^k$ with an exponent of -1.3 ± 0.2, making large blackouts more likely. In addition to criticality, it also represents an equilibrium. The distribution of blackout size has not changed in the past 30 years. They argue that you can't study large black-

outs by looking at initial triggering events only, but you must look at the root cause, and deeper, long-term forces that drive the evolution of power system.

On a slow time scale, there are several things that happen to the electricity grid. The first main force is the slow growth in load (around 0.7% growth per year for first decade of 21st century [2]). This has the effect of reducing the available capacity margins on power lines and increasing the likelihood of failures as well as possibly further constraining economic dispatch. While the slow time scale is progressing, random exogenous events, acts of nature, happen to fail individual components. These possibly initiate large cascading failures and blackout portions of the system. The engineering response to blackouts in operating policies, maintenance, equipment and controls have the effect of increasing margins on the slow time scale. These forces push against each other and settle in an equilibrium.

The following parts go through the details of these forces mathematically. These are drawn from several OPA papers [18, 12, 19].

Slow Time Scale

The slow time scale is simplified by using days as the time step, represented by index t. There are three main components to the slow dynamics.

1. The demand grows at the beginning of each day. We have $d_{it} = \lambda d_{i\rho(t)}$ where $\rho(t)$ represents the preceding time period and i is a vertex with a load demanding d_{it} on day t at peak load. They used $\lambda = 1.00005$, which corresponds to a yearly growth rate of 1.8% (the yearly average for 1980-2000). To represent daily load fluctuations, all loads are multiplied by a random number r, such that $2 - \gamma \leq r \leq \gamma$ with $1 \leq \gamma \leq 2$.

2. The response to blackouts is to upgrade the transmission system by increasing the maximum

capacity of transmission lines. If a line has overloaded in a blackout, the response is to increase its capacity so that $U_{et} = \mu U_{e\rho(t)}$ with e being an edge with capacity U_{et} on day t. They varied μ between 1.01 and 1.1. This parameter simplifies all of the efforts that go into these responses including increasing the frequency of maintenance, changing operating policy, installing new equipment, and adjusting or adding alarms and controls. These responses are modeled as happening before the next day, but in reality can take place over many different time scales.

3. The response to increased demand is to increase generator power so that all demand can be met. First, they assume that increases in power is quantized and not continuous. The quantity $\Delta G_t = \kappa(D_t/n_g)$ represents the amount of power increase for a generator with κ being a few percent, n_g being the number of generators, and $D_t = \sum_{i\in\mathcal{V}} d_{it}$ is the total demand for time period t. In order to increase generation at a node i, we need $g_{it}^+ + \Delta G_t \leq \sum_{e=(i,j)|e\in\mathcal{E}} U_{et}$, that is, the increased power needs to be able to flow out of its neighborhood. g_{it}^+ represents the maximum power generation at node i on day t. Power is continued to be added to eligible nodes, $g_{it}^+ \leftarrow g_{it}^+ + \Delta G_t$, until the generator capacity margin has risen above a prescribed level. The generator capacity margin is defined by

$$\left(\frac{G-D}{D}\right)_t = \frac{\sum_i g_{it}^+ - D_0 e^{(\lambda-1)t}}{D_0 e^{(\lambda-1)t}} \tag{2.5}$$

with $D_0 e^{(\lambda-1)t}$ being the average power demanded, not including daily fluctuations. The generator capacity margin is used to deal with daily fluctuations in demand. The generator capacity margin of the U.S. has an estimated mean value between 15% and 30% [12].

These forces balance against system failures throughout time. The line and generator failures were modeled as being possible to take place every day and begin with random events with a given

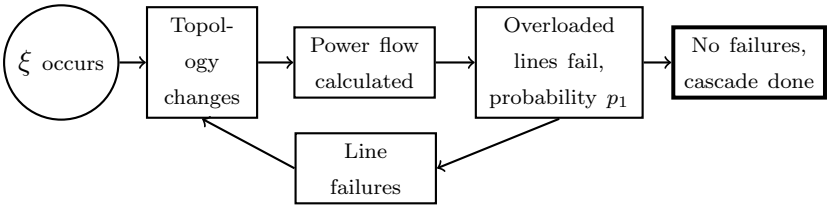


Figure 2.1: OPA simulation of a cascading power failure

probability. The next section goes into detail about the cascading process that is possible after the random events take place.

Fast Time Scale

Individual blackouts triggered by random events (equipment failure, weather, vandalism, attack) can become widespread through a series of cascades. The initial goal of building this cascading failure model was to produce a list of lines that could plausibly be involved in cascading event. It simplifies the process of cascading failures considerable, but is still able to capture important effects of topology changes throughout the process. Figure 2.1 gives a quick overview of the fast time scale simulation used to model cascading power failures.

The OPA model of the cascade process begins with an exogenous event, ξ, that effects the topology of the grid. In their initial version, ξ were the branches that randomly failed in an independent and identically distributed (I.I.D) manner, a Bernoulli trial with probability p_0. Moments after the failure, the power flows are rerouted through the new topology based on the laws of physics. In a longer time frame, it is possible for operators to take actions such as load shed and generator redispatch. The resulting loading on the transmission lines is evaluated. Their model then failed overloaded transmission lines with a Bernoulli trial with probability p_1 between 0.1 and 1. After all overloaded lines are evaluated, a transition is made to the next stage. Either there are no more failures, in which case the cascade is over, or more branch elements have failed, the

topology changes, and the operator is allowed to take recourse. This process repeats until the system is stable and no failures occur. Figure 2.2 shows a visual example of the cascading process.

For a given grid $\mathcal{G}$ and initial demand d_0

Initial ξ For $e = 1, ..., n_e$, draw ω from $U[0, 1]$ and if $\omega \leq p_0$, line e fails.

$\vdots$

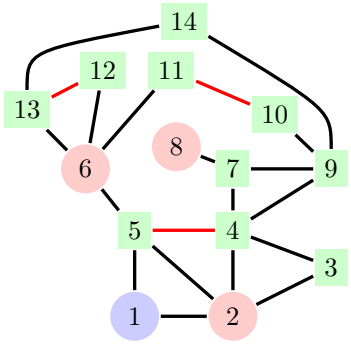
Stage m Calculate the power flows f_m, a column containing the branch flows for all edges in $\mathcal{E}$, by using DC power flow equations with demand vector d_m.

For $e = 1, ..., n_e$, if branch flow $f_{em} \geq U_{em}$, draw ω from $U[0, 1]]$ and if $\omega \leq p_1$, line e fails.

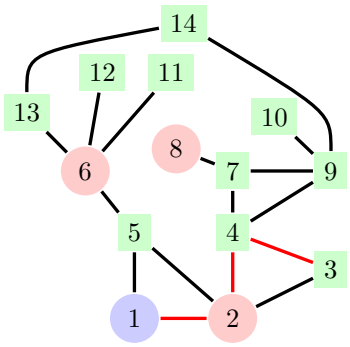
Dynamic Equilibrium

These opposing forces eventually find an equilibrium. The equilibrium tends to be near critical points, which are points that have maximum power flow through the network for the nominal system capacity. The system self-organizes toward these points, which maximize efficiency of its assets. When the system approaches these critical points, the power flow becomes limited due to two possible causes:

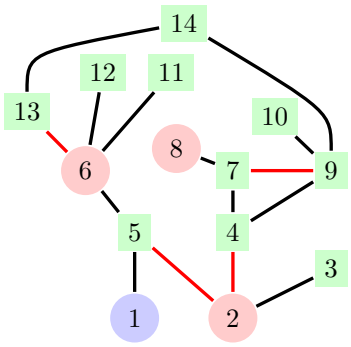
- The power flows are limited due to transmission line constraints. This type of critical point has larger blackouts, but happens less frequently. In addition, the blackouts typically have multiple lines tripping.
- The power flows are limited due to generation capacity. This results in frequent blackouts, but of smaller size.



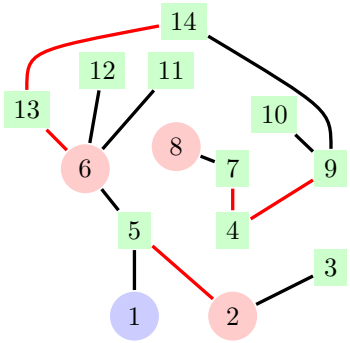
(a) Initial Event: The three red lines are outaged and the power flow redistributes.



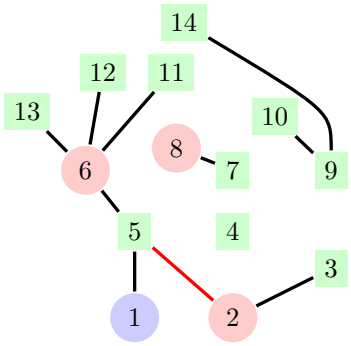
(b) Stage 1: Lines 1-2, 2-4, and 3-4 are overloaded. Lines 1-2 and 3-4 fail, but line 2-4 remains in operation at an overloaded state.



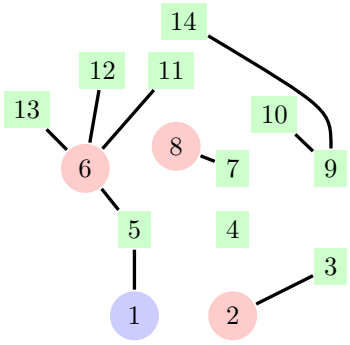
(c) Stage 2: On the new topology, lines 2-5, 6-13, and 7-9 become overloaded. The cascade progresses by outaging lines 2-4 and 7-9.



(d) Stage 3: This has the effect of routing all power destined for load 8 through the north passage. Lines 13-14, 4-7, and 4-9 are outaged along the path.



(e) Stage 4: Finally, line 2-5 that is still overloaded is outaged.



(f) Stage 5: The system stabilizes into islands with generator 1 serving load 6. However, loads 2 and 8 are out of service.

Figure 2.2: An example of a cascading power failure. Node 1 is a generator and nodes 2, 6, and 8 are loads.

However, it is also possible to be in an operating regime which is close to both types of critical points simultaneously. When this is the case, blackouts of all sizes occur. While this operating regime may not be good from a reliability standpoint, it has the desirable characteristic of being able to deliver the maximum power for the system. That is, when these two points are balanced, the system is capable of maximum throughput from its generators to its loads with minimal excess capacity. This is important from an economic perspective and a critical reason the system self-organizes to these types of points. This type of point tends to be the cheapest way to supply all the loads with power, while statisfying the minimum system reliability standards.

Additional Complexities

The OPA model can be extended to include many additional complexities. It is always a balance of how much resources you have to solve the problem and the amount of resolution you need in the solution. The base OPA model is easy and fast to replicate, however by trying to gain increased resolution in the output, the model becomes increasingly complex and difficult to solve in a timely manner.

In related work, Chen et. al. [13] find many similar conclusions to the OPA model by using an extension which included a hidden failure model. A hidden failure is undectable in normal operations, but as the system becomes disturbed, a relay may incorrectly trip. These protective relays are in operation to protect the line from overloads and disconnect it from the system. This work introduces a loading dependent failure model that trips neighboring lines and nearby components fail. This hidden failure is exposed the first time a disturbance nearby occurs and if it doesn't fail then, it is assumed to be properly operating and future disturbances will not undergo this hidden failure mechanism. The probability of these happening in the real world are non-negligble according to NERC data [16].

This work displays many similar results to the OPA papers. First, the power law behavior near critical loading can be seen by varying the system loading levels. They find that by increasing spinning reserves the risk of big blackouts is greatly reduced. The blackout size distribution tends toward an exponential distribution as reserve levels are increased. By lowering the hidden failure rate, the system becomes more robust and larger blackouts become less likely. Finally, they note that prompt control actions can reduce the risk of big disturbances. While all of these results seem fairly straightforward, it points to the fact that the important aspects of the cascading process are being modeled and the effects are similar to what would happen in the real world.

Bienstock made several modifications [9] to the original OPA model in order to remove some undesirable effects of the simulation, notably the erratic behavior of its output under small changes in the input. To do this, he introduced the concept of memory to the system. In order to see if a line is in or near an overloaded state, it uses a running time average of the current state and previous states.

$$\tilde{f}_{et} = \alpha f_{et} + (1 - \alpha)\tilde{f}_{e\rho(t)} \tag{2.6}$$

Here, f_{et} represents the power flow on edge e at time t. Then, $\tilde{f}_{et}$ is used in the overload and line failure calculations.

In addition to including a memory, he also smoothed out the definition of an overloaded line by creating a step in between normal and overloaded states in which the failure probability was more than nominal but less than in the overloaded state. Using $0 \leq \epsilon \leq 1$, for edge e, the following

failure model smooths the effects of overloaded lines failing.

$$\tilde{f}_{et} \geq (1+\epsilon)U_{et} \quad \text{The line fails with certainty} \tag{2.7}$$

$$(1-\epsilon)U_{et} < \tilde{f}_{et} < (1+\epsilon)U_{et} \quad \text{The line fails with probability } \frac{1}{2} \tag{2.8}$$

$$\tilde{f}_{et} \leq (1-\epsilon)U_{et} \quad \text{The line remains in operation} \tag{2.9}$$

An AC power blackout model was developed at the University of Manchester (Manchester Model) by Nedic [36] and the original OPA authors. This model is able to represent real world disturbances more accurately by using the full non-linear model that describes power flow. This gives resolution into areas for generator instability, under-frequency load shedding, redispatch of active and reactive sources, and emergency load shedding. The model has both automated control systems and operator recourse. This model was used in OPA criticality works by Mei et. al. [34], including one with mechanisms using voltage stability margins [33]. However, the additional complexity comes at the cost of having to solve a nonlinear program and the loss of convergence gaurantees. This is all done in order to represent something that is a companion to, but not the main driver of the cascading process (according to the Northeast outage working group, the main driver was angle instability not voltage instability).

Mei *et. al.* have worked to improve the accuracy of OPA by increasing the level of detail [35]. In the fast dynamics, along with protective relays being modeled as hidden failures, they included a failure mechanism for the operational dispatch center that is responsible for generator redispatch. In addition, they used a failure model where an underloaded line is failed with probability $p_1 \left(f_e/U_e\right)^N$, with $N = 10$. For the slow dynamics, they added a step that models a planning department by increasing the capacity of lines which have a loading rate (f_e/U_e) greater than a set point.

In 2013, Qi *et. al.* extended this model to include slow dynamics of vegetation growth

and management as well as differential equations representing line heating [38]. Neglecting spatial variation and heating/cooling effects from the environment, they model line temperature by a differential equation

$$\frac{dT(t)}{dt} = \alpha I^2 - \nu(T(t) - T_0) \tag{2.10}$$

where $T(t)$ is the line temperature at t, I current, T_0 initial temperature, and α and ν are calculated parameters. This can be solved by assuming constant branch flow, f, to calculate the temperature over time

$$T(t) = e^{-\nu t}\left(T_0 - T_e(f)\right) + T_e(f) \tag{2.11}$$

which can be used to find the final equilibrium temperature, $T_e(f)$ (a function of its constant power flow f on the line), and time until a given temperature. Using the calculated temperature, the horizontal span, and an elongation parameter, the line sag distance can be calculated. When the minimum distance between lines and vegetation passes a breakpoint based on transmission line characteristics such as operational voltage, the line will fail. They model the vegetation with a daily growth rate model and include a managment simulation in which they identify future potential hazards and cut down trees over time. The statistical analysis of their model agreed well with historical data in China.

2.1.3 Power Flow Review

In order to run the OPA simulation of cascading failures, the ability to calculate power flows on the system is critical. To model a balanced, three phase power flows, in full resolution we need to model complex power. The alternating current of the power system can be represented by sine or cosine waves. A three phase power system has three lines for each transmission element and each line has a wave that is out of phase with the other two. Using one wave as a reference, the

other phases attempt to be 120 degrees behind reference and 120 degrees ahead of reference. This improves the efficiency and quality of power for loads over a 2 line system as well as not requiring an excessive number of lines for each transmission element.

Complex power has both real and reactive parts. Alternating currents on a circuit affect components of energy storage such as inductors (changing the current as opposed by a voltage) or capacitors (store electrical charge). Over one full cycle of the electricity changing direction, across any individual element there can be real power transferred. However, there is also power which is stored and released within one cycle and the net energy transfer of this power is 0. This is called reactive power and is modelled as the imaginary component of complex power.

In a balanced three phase system, at every vertex i, we have

$$S_i = P_i + jQ_i = V_i I_i^* \tag{2.12}$$

where P is real power, Q is reactive power, V_i is complex voltage, and I_i^* is the complex conjugate of current. In addition, with Kirchoff's Current Law, we have $I_i = \sum_{k=1}^{n} Y_{ik} V_k$, where Y is the admittance bus matrix. The admittance is the inverse of imdedence, that is

$$Y = G + jB = 1/Z = 1/(R + jX) \tag{2.13}$$

where B, the imaginary part of admittance, is susceptance and X, the imaginary part of impedence, is reactance. Now we have that the complex power at every vertex i is

$$S_i^* = P_i - jQ_i = V_i^* \sum_{k=1}^{n} Y_{ik} V_k \tag{2.14}$$

By converting these into rectangular coordinates, we get two equations for each bus

$$P_i = \sum_{k=1}^{n} |V_i||V_k| \left[G_{ik}\cos(\theta_i - \theta_k) + B_{ik}\sin(\theta_i - \theta_k)\right] \tag{2.15}$$

$$Q_i = \sum_{k=1}^{n} |V_i||V_k| \left[G_{ik}\sin(\theta_i - \theta_k) - B_{ik}\cos(\theta_i - \theta_k)\right] \tag{2.16}$$

For each bus, in addition to P_i and Q_i, we have its voltage $|V_i|$ and its phase angle θ_i. So, we have $4n_v$ variables and $2n_v$ equations. By supplying the value to $2n_v$ variables and defining a slack bus, we can find unique values for the remaining $2n_v - 1$ variables. Depending on the type of bus, different variables are supplied. If it is a generator, both P and V are supplied. A load is defined by a P and Q. The slack bus is a generator in which the phase angle is set to 0. The phase angle θ and reactive power production Q are found for each generator and the phase angle θ and the voltage $|V|$ are found for the loads.

The non-linear AC power flow equations model the net power and reactive power injects as well as voltage and phase angle at each vertex of the power system. A few simplifying assumptions are made to allow the equations to become linear for the DC (decoupled) power flow equations. Then, using the DC power flow model, basic economic dispatch and unit commitment models are shown.

Decoupled Power Flow

This model makes assumptions such as lossless lines, small voltage angle differences, and a flat voltage profile. This is a common simplifying model which is used routinely in economic and reliability analysis of power systems. A flat voltage profile implies that $\forall i \in \mathcal{V}$, we have $V_i = 1$.

Small voltage angle difference in conjunction with sine and cosine give the following approximations

$$\cos(\theta_i - \theta_j) \approx 1 \tag{2.17}$$

$$\sin(\theta_i - \theta_j) \approx \theta_i - \theta_j \tag{2.18}$$

In addition, since these lines are lossless, R and G are 0. Using this approximations, Equation (2.15) becomes

$$\theta_i - \theta_j = X_e f_e \qquad \forall e = (i,j) \text{ s.t. } e \in \mathcal{E} \tag{2.19}$$

where $X_e = B_{ij}^{-1}$. We also have conservation of energy at each bus in the network, implying that the sum of generation and demand is equal to net flow into the transmission network, given by

$$\sum_{e \in \mathcal{E} | e=(i,j)} f_e = g_i - d_i \qquad \forall i \in \mathcal{V}. \tag{2.20}$$

When a line has failed, the power flow has obviously dropped to 0, that is $f_e = 0$. In addition, since there is no link between the phase angles, Equation (2.19) should no longer be enforced.

Economic Dispatch

To clear the electricity market at a single point in time, a least cost dispatch model is used. This model takes bids from generators, a known demand, as well as transmission and ramping constraints and finds a set of generator outputs that meet the demand at least cost. Using a quadratic cost function for the generators (this cost function can be thought of as a bid from generators which includes the profit the generator would like to make for each marginal unit of production), the least cost dispatch Model (2.21) can be built.

The following model is a quadratic program with linear constraints. The objective function

is to minimize the cost of generation. Typical least cost dispatch and unit commitment models make various assumptions to allow for linear constraints versus the physical nonlinear constraints to which the power system is subject. Model (2.21) is the most basic least cost dispatch model, which could be used for clearing the real-time market. Models in use can have extensions such as "$N-1$ constraints", which are reliability and security requirements.

$$\min \sum_i c_i^2 x_i^2 + c_i^1 x_i + W_i(\tilde{d} - d_i) \tag{2.21a}$$

$$x_i - d_i = \sum_j y_{ij} \qquad \forall i \in \mathcal{I} \tag{2.21b}$$

$$X_e f_e = \theta_i - \theta_j \qquad \forall e = (i,j) \in \mathcal{E} \tag{2.21c}$$

$$x_i \in \left[G_i^-, G_i^+\right] \qquad \forall i \in \mathcal{I} \tag{2.21d}$$

$$y_e \in [-U_e, U_e] \qquad \forall e = (i,j) \in \mathcal{E} \tag{2.21e}$$

Here, $c_i^2 x_i^2 + c_i^1 x_i$ can be seen as the cost function for the generator at node i and $W_i(\tilde{d} - d_i)$ is the cost for shedding load. The generator is bounded between a maximum G_i^+ and minimum G_i^- that represent its ramping rate and other physical limits over a specific time interval. If node i cannot generate power, then $G_i^+ = G_i^- = 0$.

The day-ahead market uses unit commitment models. This model has power flow equations for each time t in the day $t \in \mathcal{T}$. These are integer programs due to the introduction of binary variables w_{it}, which take on the value 1 if the generator is switched on and 0 if the generator is off.

The following logical constraint enforces this by

$$w_{it} g_i^- \leq g_{it} \leq w_{it} g_i^+. \tag{2.22}$$

The cost function can then include a fixed cost for operating a generator, such as increased staff during operation. This cost is not dependent on the level of production but rather if the plant is in an on or off state. The cost function for each node i and time period t is

$$c^2 x_{it}^2 + c_i^1 x_{it} + c_i^0 w_{it} \tag{2.23}$$

This subproblem is used in a full-day model in which the power flow problem is solved for each time period, while remaining feasible with respect to ramping rates and on and off times for generators.

2.2 Multi-Stage Stochastic Programming

We use the framework of stochastic programming to model the dynamics of cascading power failure. Each stage represents an epoch at which one or more lines fail based upon their loading. The multi-stage stochastic program as a large mixed-integer linear program can be approximated.

Mixed-Integer Program

The mixed-integer formulation of cascading power failures uses binary variables to model line availability for stages in the cascade. In addition to these extra variables, some input parameters are needed to formulate the problem. First, the number of stages, n_T, for the cascade needs to be decided. If large blackouts are of interest, this number should be large enough to not exclude the worst-case scenarios. Also, the number of outcomes n_0 at each node of the scenario tree needs

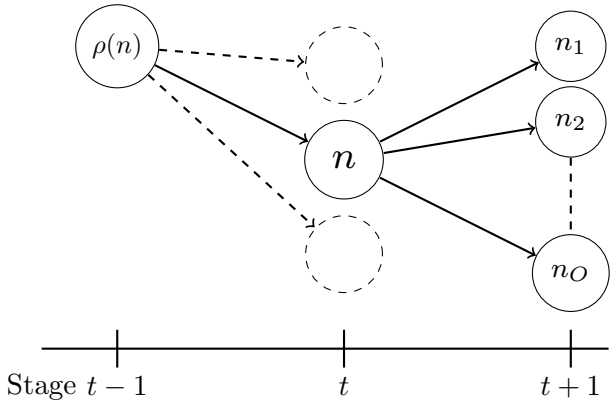


Figure 2.3: Scenario tree for Mixed-Integer Formulation

to be decided. These outcomes represent the decision-dependent uncertainty. The size of the problem is related to the size of the stochastic tree, which is $n_o^{n_T}$. As the subproblem is difficult, the computational complexity of this problem increases rapidly with the number of outcomes and stages.

2.2.1 Decision Dependent Uncertainty

In order to model the decision-dependent uncertainty, a cumulative distribution function (cdf) for line failures based on loaded is needed. This model uses a parameter, R, to represent the effective capacity of a line. Scenarios are created by sampling from the cdf before formulating the MIP. This *effective capacity* represents the loading of the transmission line that causes its failure, and we capture the failure of a line with a binary variable, z.

In Mixed-Integer Programming, there have been standard equations developed to model logical conditions. We have two logical conditions that need to be modeled in order to represent this decision dependent uncertainty. First, the condition that the line will fail if it has more power

flow than its effective capacity.

$$\text{If} \qquad \left|y_{e\rho(n)}\right| > R_{en}$$

$$\text{Then} \qquad z_{en} = 0$$

To model this logic, a Big-M constraint can be used, where M represents a large number defined below.

$$y_{e\rho(n)} - R_{en} \leq M_e^R(1 - z_{en}) \tag{2.24}$$

$$y_{e\rho(n)} + R_{en} \geq -M_e^R(1 - z_{en}) \tag{2.25}$$

with $M_{en}^R = U_e - R_{en}$.

Now, when the line is available in stage n, that is $z_{en} = 1$, then the line flow in the predecessor node is within the effective capacity, -$R_{en} \leq f_{e\rho(n)} \leq R_{en}$.

The second logical condition is that when the line is unavailable, the power flow on that branch is zero and the phase angles between the two nodes are not constrained.

$$\text{If} \qquad z_{en} = 0$$

$$\text{Then} \qquad y_{en} = 0 \quad \text{and}$$

$$\theta_{in} - \theta_{jn} - X_e y_{en} \text{ is arbitrary}$$

This can be achieved through the following equations.

$$-U_e z_{en} \leq y_{en} \leq U_e z_{en} \tag{2.26}$$

$$\theta_{in} - \theta_{jn} + X_e y_{en} \geq -M_e^{\theta}(1 - z_{en}) \tag{2.27}$$

$$\theta_{in} - \theta_{jn} + X_e y_{en} \leq M_e^{\theta}(1 - z_{en}) \tag{2.28}$$

with $M_e^{\theta} = 2\theta_{max} + X_e U_e$.

Failure Density Function

The OPA simulation uses a step function for the failure probability of a line shown in **??**. We modify the effective capacity distribution to give ramp up line failure risk according to a piecewise linear function also shown in **??** To approximately model this analytically, a sample from the distribution can be used to form the effective capacity of the lines. Let $\omega_n \equiv [0, 0, 1, 0, \cdots, 1]$ be a vector sampled from a Bernoulli distribution and let $\alpha_n \in \mathbb{R}^{|\mathcal{E}|}$ be a vector sampled from a uniform distribution between L_e and U_e for each line. Now, a line fails if $y_{ne} \geq \alpha_{ne} U_e$ and $\omega_{ne} = 1$. With this sampling, the effective capacity can be designed to incorporate this information.

$$R_{ne} = \begin{cases} \alpha_{ne} U_e & \text{if } \omega_{ne} = 1 \\ U_e + \epsilon & \text{if } \omega_{ne} = 0 \end{cases} \tag{2.29}$$

The resulting set of effective capacities R can be represented by a cumulative distribution function. The distribution in Figure 3.1 is one example of a viable line failure distribution input. It is the result of a uniform distribution for α between L and U and Bernoulli trial with probability p for ω.

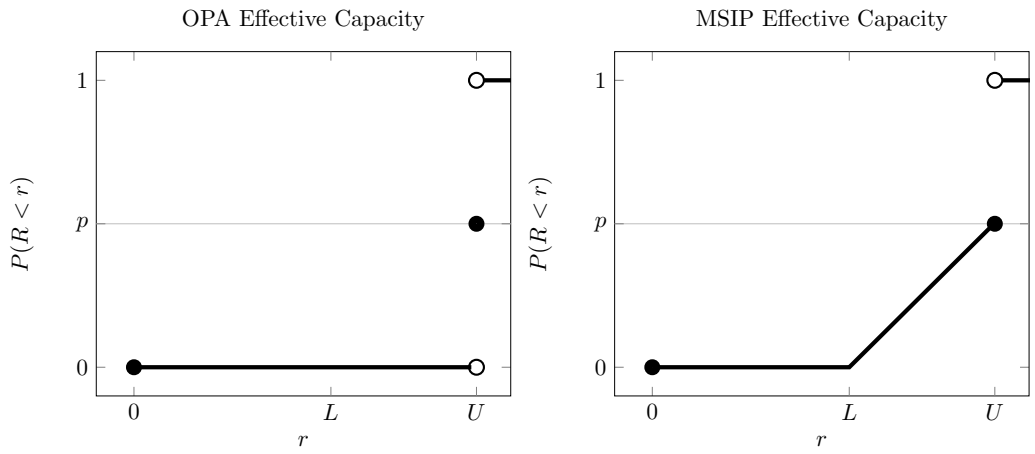


Figure 2.4: Example Effective Capacity Distributions

2.2.2 Cascade Evolution

The cascading process begins with an intial exogenous event, $\xi \in \{0,1\}^{|\mathcal{E}|}$. This may be treated as a random vector on a probability space with sample space Ξ. The following equation enforces a line failure throughout the whole cascade for all $e \in \mathcal{E}$ such that $\xi = 1$.

$$z_{en} \leq 1 - \xi_e. \tag{2.30}$$

Now, the cascade will evolve according to the dispatch decisions chosen and the pre-sampled effective capacities of the particular lines during the various stages of the cascade. The operators will be able to optimize the cost, which includes the cost of load shed, and are constrained to be within the following set,

$$\Omega(\xi) \equiv \{(d, x, y, z) \mid \text{Equations (2.19), (2.20) and (2.24) to (2.30) hold } \} \tag{2.31}$$

where $d = [d_0, d_1, \cdots, d_N]$ is the set of vectors of demand served for each node in the scenario tree. Now, this representation of the cascading process can be used in many ways. One example

is a chance constrained model, in which the operator requires a line to be available in a certain percentage of scenarios. Another example is to use this a sa subproblem in a larger planning model.

2.2.3 Least Cost Dispatch

The following model is a least cost dispatch model that includes the effects from cascading failures due to a set of contingencies $\xi \in \Xi$.

$$\min \quad \mathbb{E}_{\Xi} \sum_{in} \left[c^2_{in} x^2_{in,m} + c^1_{in} x_{in,m} + W_{in}(\hat{d}_i - d_{in,m}) \right] \tag{2.32a}$$

$$(d, x, y, z)_m \in \Omega(\xi_m) \qquad \forall \xi_m \in \Xi \tag{2.32b}$$

This can be adapted to several different problems in power systems. The main difficulty with this multi stage stochastic program is computational complexity. The program can be calibrated in order to produce similiar load shed distributions to the OPA model, however the number of outcomes and stages in the model needs to be small or the computational hurdle becomes too large.

In order to get reliable output from the optimization routine, this model needs to be calibrated against the OPA simulation. The primary calibration parameters are L and p in the line failure distribution shown in Figure 3.1 as well as the costs for load shed, W, which depend upon the stage in the cascade. The OPA simulation is a greedy algorithm, attempting to maximize demand served in the current stage without regard to the line failure consequences. In order to capture this effect, more weight was placed on demand served in earlier stages of the cascade.

The root problem used is comprised of 4 intitial outages and each outage has a scenario tree

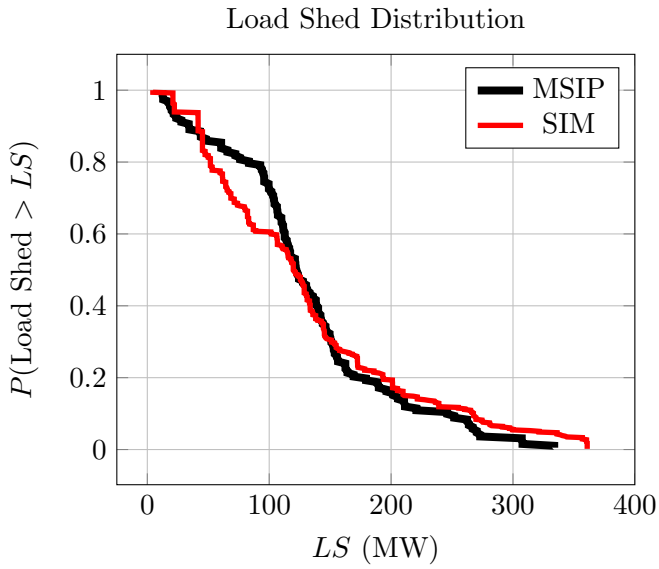


Figure 2.5: Load Shed Distribution for the OPA simulation and MSIP formulation

Initial Contingency	SIM $\mathbb{E}[LS]$	SIM St.Dv.$[LS]$	MSIP $\mathbb{E}[LS]$	MSIP St.Dv.$[LS]$
2,12,21	196	128.1	184	112.7
5,25,34, 82	254	187.1	160	69.3
12,14,34,111	131	85.12	151	59.0
13,24	145	123.4	123	84.4

Table 2.2: Comparison of Simulation and MSIP outputs

that is 4 stages long with 2 outcomes at each node. The power system modeled is the IEEE 118 bus grid with a nominal demand of 3668 MW and around 29,600 MW in branch capacities. The parameters chosen for the MSIP model were $p = .5$, $L = .575$. The weights W_s for the stages of the cascade were $[500, 10, .05, .0001]$, which seemed to capture the greedy behavior of the OPA simulation. The OPA simulation used the step function failure model with $L = .99$ and $p = .5$. The output of the simulation and MSIP formulation are shown in Table (2.2) along with the load shed distribution for initial line failures on lines $[12, 14, 34, 111]$ in Figure (2.7).

2.3 Model Flexibility

The primary strength of this formulation is the flexibility it has in using additional constraints and objectives to inform decision making about power systems. All of these models can then be solved with commercial solver software without the need to develop additional specialized routines. The first example is a chance constrained model, which enforces a probabilistic constraint on the number of line failures in any scenario. In this model, we constrain the probability that fewer than a given number of lines failures to be close to 1. A typical case in power systems would be that the system should not go beyond 1 contingency in a large percentage of scenarios.

$$P\{\#\text{ Line Failure} \leq k\} \geq 1 - \epsilon \tag{2.33}$$

This can be done by adding another binary variable for each scenario that represents whether or not that scenario has fewer than k failure. These new binary variables are summed and constrained by the given probability.

$$\sum_i z_{is} \leq M_s \hat{z}_s + |\xi_s| + k \tag{2.34a}$$

$$\sum_s \hat{z}_s \leq \epsilon |\mathcal{S}| \tag{2.34b}$$

where $M_s = |\mathcal{E}| - |\xi_s| - k$. The objective of this program would be to minimize load shed as in Model (2.32).

The second example is a redispatch model that moves to a generator configuration that is close to the original while minimizing the expected size of the cascade. This model tries to find an operating configuration that is within a given distance from the current operating regime and minimizes the worst-case scenario load shed. The input for this model is the current operating

configuration (p_0, d_0) as well as a distance vector δ that represents how far each generator can move from its current output levels. In this case, the root node of the stochastic tree has variables that represent initial generator output levels and the child trees are constrained from how far they can move from the initial configuration. Also, a continuous variable are added that represents the amount of load shed in the worst-case scenario. The objective is to minimize this worst-case scenario.

$$x - x_0 \leq \delta \tag{2.35}$$

$$l \geq \sum_i \hat{d}_i - d_{is}. \tag{2.36}$$

This program could be used when the system is becoming close to unstable and cost concerns become less of a priority than system stability.

Another exampe is a reserve-planning model that allocates reserves among the generators in such a way as to minimize the worst-case failure. In this case, an operating configuration need not be given. The program can either search for an operating configuration as well as reserves or given an operating configuration, allocate the reserves. These operating reserves, r, determine where the system is able to relieve congestion in a contingency.

$$x_{in} + r_{in} \leq \overline{P}_i \tag{2.37}$$

$$x_{in} \leq x_{i\rho(n)} + r_{i\rho(n)} \tag{2.38}$$

$$\sum_i r_{in} \leq \beta \sum_i \hat{d}_i \tag{2.39}$$

where β is the level of operating reserves alotted for the system.

Finally, the example we use is a transmission expansion model that allocates a budget for capacity expansion on the grid in such a way as to minimize the expected size of the cascade.

We then look at the computation results and compare them with the OPA simulation to a simple heuristic which allocates the expansion budget.

2.3.1 Transmission Expansion

Using the MIP formulation, a stochastic program can be developed to model planning decisions with respect to cascading power failures. Consider the problem of transmission expansion. A set of contingencies, $\xi \in \Xi$, has been identified as the primary risk for initiating a cascading event. There is a budget to use for expansion and the objective is to allocate the budget in such a way as to minimze some risk measure of load shed for the cascading power failures.

Let u be the design variable, which is decided at the root node and represents additional capacity on power lines. This affects the constraint set such that equations (2.26) and (2.29) become

$$-(U_e + u_e)z_{en} \leq y_{en} \leq (U_e + u_e)z_{en} \tag{2.40}$$

$$R_{ne} = \begin{cases} \alpha(U_e + u_e) & \text{if } \omega_{ne} = 1 \\ (U_e + u_e) + \epsilon & \text{if } \omega_{ne} = 0 \end{cases} \tag{2.41}$$

The objective is to allocate a budget for capacity additions in order to reduce the expected blackout size. W_{in} is the cost of load shed at bus i in node n of the scenario tree. Let u be the vector of capacity additon and w be a vector of binary variables representing whether that transmission element gets new capacity. B_u and B_w is the budget for capacity expansion.

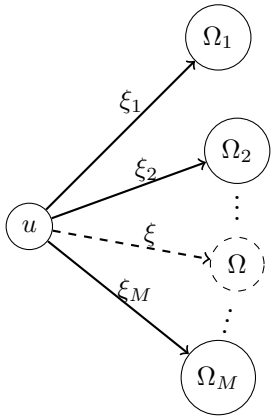


Figure 2.6: Scenario Tree for Two Stage Problem

$$\min \quad \mathbb{E}_{\Xi} \sum_{in} \left[c^2_{in} x^2_{in,m} + c^1_{in} x_{in,m} + W_{in}(\hat{d}_i - d_{in,m}) \right] \tag{2.42a}$$

$$(d, x, y, z)_m \in \Omega(\xi_m) \qquad \forall \xi_m \in \Xi \tag{2.42b}$$

$$\underline{X}_e w_e \leq u_e \leq \overline{X}_e w_e \qquad \forall e \in \mathcal{E} \tag{2.42c}$$

$$\sum_e u_e \leq B_u \tag{2.42d}$$

$$\sum_e w_e \leq B_w \tag{2.42e}$$

2.3.2 Computational Implementation

The model outlined in (2.42) is solved using Gurobi 4.5. The stopping criteria was either a 40% optimality gap or 10,000s. The program was run for several different expansion budgets, with both the total budget and the maximum number of lines being changed. The output of this model is a vector u which represents the amount of capacity to add to each power line on the grid. This is used to modify the initial grid and run the OPA simulation to find the effects it has on the system.

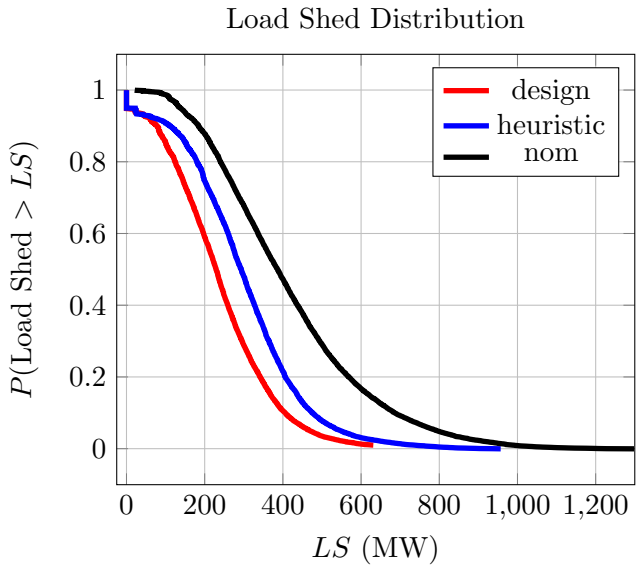


Figure 2.7: Load Shed Distribution for the OPA simulation and MSIP formulation

In order to find out whether these solutions are reasonable or not, a heuristic was developed to compare against.

The heuristic used is based on a large number of OPA simulation runs in which the power lines were ranked in descending order based on the percentage of runs in which that given power line has failed. Then, for a given total budget B_u and maximum number of changed lines B_w, the heuristic picks the top B_w lines in the list and then distributes the budget B_u evenly over the lines. The OPA simulation is used to compare the results of the two models. Since the MSIP was calibrated based on the 4 contingencies, a second set of 4 random initial contingencies to start the OPA process to evaluate the effects.

In Figure 2.7, we see that we were able to shift the load shed distribution using the design u found from the MSIP solution of Equation (2.42). In order to evaluate the solution, we ran the OPA simulation and compared against a simple heuristic. The heuristic used the results of a number of trials of the OPA simulation and distributed capacity evenly among the 10 lines which had failed the most. This strategy did reasonably well. The MSIP was able to beat this strategy on some

occasions, but not always. The MSIP has a coarse representation of the underlying uncertainty due to the computational difficulty in solving a multi-stage stochastic program. The size of the scenario tree was 4 stages with 3 outcomes per node, and even when it was this small, there was often at least a 40% optimality gap.

2.4 Conclusion

Using the OPA simulation as a reference for how the grid may respond to cascading power failures, a mixed integer model was developed which represents the cascading effects over a fixed number of outcomes and stages. While this model can be difficult computationally due to the decision dependent uncertainty, it is extremely flexible. The model can be used to include cascading effects in a wide range of power system problems with optimality criteria ranging from least cost to minimize worst-case problems. A computation example was done on transmission expansion, which showed the model was able to better than a reasonable heuristic, even though it was only solved to a 40% optimality gap. Future work can be done in order to improve the solve time for these types of models based on techniques developed in stochastic and mixed integer optimization.

Chapter 3

Using the OPA Simulation for Transmission Expansion

The mixed integer stochastic program built in Chapter 2 is too difficult to optimize at the scale we need to model the uncertainty in the cascading process. In order to work around this difficulty, we do not approximate the cascade analytically, but rather we simulate the cascading process. The primary benefit is removing the temporal linked binary variables for line outages from the master problem allowing sequential evaluation of the decision dependent uncertainty. This allows us to parallelize the OPA evaluations to increase the computational effort.

We explore the long term design problem of transmission capacity expansion using the OPA simulation to evaluate rare event stress on the transmission infrastructure. We begin by laying out the OPA simulation which involves sequential evaluations of a linear program modeling economic dispatch with a weighted load shed term in the objective. The lines fail in the same manner described in Chapter 2, and the failure density function can be seen as the line having a random effective capacity that when exceeded the line fails with certainty. As the simulation proceeds, the

topology changes and a new linear program is solved. The first primary benefit of this sequenctial procedure is the ability to hot start the LP solver. The constraint for a line that has failed which links the connected nodes phase angle is relaxed. The LP solver can use the previous basis and perform a small number of pivots to find an optimal solution to the new problem representing the changed topology.

Following the discussion on the LP subproblem and the failure density function, we show the algorithm used to evaluate an OPA simulation trial for a fixed demand and contingency. This single trial is sampled from the cascade evolution distribution Ω. We explore the uncertainty in load shed distribution due to the uncertainty caused by the cascade evolution and a small subset of initial contingencies in Ξ. We describe the statistical measures of load shed distribution we are interested in Section 3.1.1 and focus primarily on the function describing expected value of load shed for the simulation exploration and design problem optimization.

We then look at whether we can create a surrogate function for the OPA cascade simulation in order to develop a computationally cheap evaluation that is correlated with the function representing expected load shed. We plot the values of load shed by stage in the cascade and see that the cascades with large load shed do not necessarily have large load sheds in the first stage. Additionally, the number of lines that fail in the first stage of the cascade is not correlated with the load shed at the final stage. While we certainly have not explored all possible surrogate functions, we proceed by using the full cascade simulation instead of a surrogate function.

Since we need to use the full cascade simulation and there is a high standard deviation in the load shed distribution, it is important to use variance reduction techniques to reduce the computational effort needed to achieve a small confidence interval on the expected value of load shed. Common random numbers significantly improves the ability to compare different systems

and reduce the variance of the difference of the performance of two systems. For design problems, this is exactly what we need. We would like to evaluate how systems perform relative to each other.

After explaining the common random number scheme used, we proceed to explore the function describing the expected value of load shed. This OPA simulation and the expected value of load shed over many trials provides fundamental difficulties to the optimization process. The function is neither convex nor continuously differentiable. The load shed is inherently noisy and its distribution is wide, often characterized by a power law distribution. Some of these effects can be smoothed away by employing a wide range of potential initiating events, however it may be important to optimize against a small subset of events that have a higher likelihood of occurrence and are known to be risky. In this case, the non-convexity and discontinuity are most apparent.

After learning about the function characteristics, we begin to explore how we can optimize the design problem of transmission expansion. We look to the class of derivative free optimization techniques of direct search methods which are simple, flexible, and powerful. Altough we do not have the nice function properties that guarantee global or local convergence, they have been found to work well for these types of black box optimization routines. By using a grid based pattern search method that describes a spanning set, we can be assured of local convergence in regions of the functions with nice properties. Additionally, we can use exploratory steps that can be based off of accessory information that can speed up the search routine and not ruin the local convergence properties.

In order to tackle this problem, we use massive computational effort through parallelization of the OPA simulations. We describe this parallelization process using HTCondor on the UW-Madison Center for High Throughput Computing. We employed HTCondor DAGMan to manage the job submission procedure for each iteration in order to smooth job submissions and ensure a small

number of idle jobs at any given time to reduce the stress on the computational infrastructure. The file and submission structure is described as well as the scripts needed to manage this process. Years of computational time can be done within a short time horizon. This enables a fine mesh search pattern on the function in order to improve on a completely brute force search procedure. Finally, we show the results from implementation of this simple DFO method and the improvements in expected load shed from the OPA simulation.

3.1 OPA Cascade Simulation

The load shed distribution of the fast time scale OPA model (given in Section 2.1.2) has been shown to have the same power-law distribution seen in real-world blackout data. This simulation can be seen as a surrogate model for the response of the power system to rare-event stress. As such, it is useful to explore the effects different parameters can have on the distribution and even optimize over them to find any characteristics or trends there may be. To begin, we need to develop an efficient evaluation of the cascading power failure simulation for eventual use in an optimization procedure.

A brief description of the parameters, variables, indicies and sets are given and the LP subproblem for the cascade simulation follow. Variable x_j represents the generation from generator j in the set of all generators $\mathcal{J}$. The LP has a quadratic cost function with c_j^2, c_j^1, c_j^0 representing the cost parameters for generator j. Branch flow is represented by y_e for branch e in the set of all branches $\mathcal{E}$. The parameter b_e represents the susceptance of branch e and is used to describe the branch flow based on the difference of phase angles between the two connected nodes. The variable l_i represents load shed at node i in the set of all nodes $\mathcal{I}$, the parameter W representing the cost of load shed, and the parameter d_i is the nominal demand. Finally, a_{ij}^x is an incidence matrix which

takes a 1 when generator j is located at node i and a^y_{ie} is a -1 when branch e is from nodes i and a 1 when branch e is to node i. The following linear program (3.1) is a load shedding version of the standard DC OPF economic dispatch model.

$$\min_{(x,l;\theta,y)} \quad \sum_{j\in\mathcal{J}} \left[c^2_j x^2_j + c^1_j x_j + c^0_j\right] + Wz \tag{3.1a}$$

$$\sum_{j\in\mathcal{J}} a^x_{ij} x_j - \sum_{j\in\mathcal{J}} a^b_{ie} y_e + l_i = d_i \qquad \forall i \in \mathcal{I} \tag{3.1b}$$

$$y_e - b_e \sum_{i\in\mathcal{I}} c^b_{ie}\theta_i = 0 \qquad \forall e \in \mathcal{E} \tag{3.1c}$$

$$z - \sum_{i\in\mathcal{I}} l_e = 0 \tag{3.1d}$$

$$x_j \in \left[G^{min}_j, G^{max}_j\right] \forall j \in \mathcal{J} \tag{3.1e}$$

$$y_e \in [-U_e, U_e] \qquad \forall e \in \mathcal{E} \tag{3.1f}$$

$$l_i \in [0, d_i] \qquad \forall e \in \mathcal{E} \tag{3.1g}$$

where z is the total load shed for a particular dispatch point with vectors (x, y, l) representing generation, branch flows, and nodal load shed.

We look to understand the effect of adding capacity u to the system prior to an initiating event for the OPA cascading procedure. The subproblem (eqs. 3.1) is modified to allow the branch flows to attain their new capacity level, that is

$$y_e \in [-U_e - u_e, U_e + u_e] \qquad \forall e \in \mathcal{E} \tag{3.2}$$

In addition, the failure density function from the previous section shown in Figure 3.1 also change in order to account for the additional capacity. This means that not only does the capacity level change, the point L, in which the line becomes risky also moves, so that our line risk function is

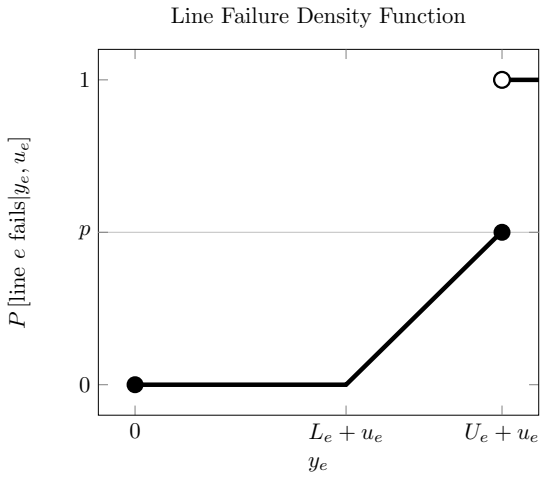


Figure 3.1: Line failure density function from eq. (3.3)

now

$$g_e(y_e, u_e) = \begin{cases} 1 & y_e > U_e + u_e \\ (y_e - u_e)U_e^{-1}p - L_e p & U_e + u_e \geq y_e > L_e U_e + u_e \\ 0 & \text{o/w} \end{cases} \tag{3.3}$$

We use the subproblem from Equation (3.1) and the failure density function to simulation the OPA cascading process. The algorithm requires sequential solves of the DC OPF with changes to the topology. The solution is still dual feasible, we take advantage of this using LP hot starts and the sequential solves typically require only a small number of pivots to find the new optimal solution. The cascading algorithm is given in Algorithm 1. The algorithm is run for a fixed design vector u, demand vector d, and initial contingency ξ representing a set of lines that has failed to initiate the cascading sequence. The topology is changed according to the contingency ξ and the new DC OPF problem is solved. The algorithm proceeds by using the failure density function to determine the probability that a line fails and then samples a Bernoulli variable with this probability to determine the outcome. If no lines fail, the cascade is considered over. If lines fail, the topology is changed, the new LP is solved, and the process repeats until no lines fail.

Algorithm 1 OPA cascading algorithm simulating the evolution of the cascading process governed by uncertainty space Ω

procedure OPA(u, d, ξ)
  Solve (3.1) to find base case load shed z_0
  ξ occurs and corresponding changes to the grid are made
  Stage $s \leftarrow 1$
  while Not DONE **do**
    Solve (3.1) to find power injects and branch flows for adjusted grid
    $\mathbb{O}_s \leftarrow \emptyset$
    for $\forall e \in \mathcal{E}$ **do**

$$\mathbb{O}_s = \begin{cases} \mathbb{O}_s + \{e\} & \text{w/ prob. } g_e(y_e, u_e), \text{ draw } \omega_{es} \\ \mathbb{O}_s & \text{o/w} \end{cases}$$

    if $\mathbb{O}_s \neq \emptyset$ **then**
      Modify Grid with $\mathbb{O}_s$
      $s \leftarrow s + 1$
    else
      $s^* \leftarrow s$, calculate z_{s^*}, DONE
  Load Shed $\lambda(u, d, \xi, \omega) = z_{s^*} - z_0$

3.1.1 Risk Measures

The OPA algorithm simulates one outcome for a fixed u, d, and ξ. Let ω represent this sampled uncertainty and $\lambda(u, d, \xi, \omega)$ be the load shed from this single OPA trial. Repeating this process for N^t trials give a distribution of load shed values. We are primarily interested in the expected value of load shed over the initial contingencies Ξ and the cascade evolution Ω. Let function $f(u)$ for design u, fixed demand d and contingency ξ be calculated as

$$f_d(u) = \mathbb{E}_{\Xi,\Omega}\left[\lambda(u, d, \boldsymbol{\xi}, \boldsymbol{\omega})\right] \tag{3.4}$$

For N^t trials with load shed $Z_n = \lambda(u, d, \xi^n, \omega^n)$ and $n \in 1, ..., N^t$, we can describe the distribution of load shed by a set of risk measures. We just looked at the expected load shed and the sample mean for N^t trials can be given in Table 3.1. In addition to the mean, we have the sample variance, which is a measure of the width of the distribution and the standard error, which measures how accurately we have calculated the mean based on how many samples we have taken.

Sample Mean	f	$\hat{Z}(N^t) = \sum_n \frac{Z_n}{N^t}$
Sample Variance	s_d^2	$S^2(N^t) = \sum_n \frac{(Z_j - f)^2}{N^t - 1}$
Standard Error	s_e^2	$Var[f] = \frac{s_d^2}{N^t}$
Confidence Interval	$CI(1-\eta)$	$f \pm z_{1-\eta/2}\sqrt{s_e^2}$
Value at Risk	$VaR(\eta)$	$\{Z_a \vert a = \lfloor \eta N^t \rfloor, Z_i \leq Z_{i+1}\}$
Conditional Value at Risk	$CVaR(\eta)$	$\mathbb{E}[Z_n \vert Z_n \geq VaR(\eta)]$

Table 3.1: Risk Measures

With the mean and standard error we can describe $1-\eta$ confidence intervals, which says that the mean is within a given range at least $1-\eta$ percent of the time. This confidence interval is calculated using the inverse of the standard normal distribution. The value at risk is a measure of the percentile of the distribution, so that the η value at risk is the value of the load shed at percentile η. The conditional value at risk is the expectation of the load shed conditioned on it being above a percentile η. This finds the area underneath the tail of a distribution. All of the risk measures are shown along with the formulas to calculate them for a set of N^t samples in Table 3.1. Additionally, these statistical measures are plotted for the load shed distribution for a sequence of design points u in which capacity is being added to one line in Figure 3.2

3.1.2 Parameters and Simulation Inputs and Outputs

We now go into the details involving the OPA simulation, in particular, the parameters that are used in the algorithms and the input and output files used in the computational implementation. As with all of the models in this thesis, the dispatch model is done on a network topology with parameters defining susceptance for branch flow, generators and their cost coefficients, and the nominal demand of the system. These paramters are stored in a <.gr> file and is one of the inputs

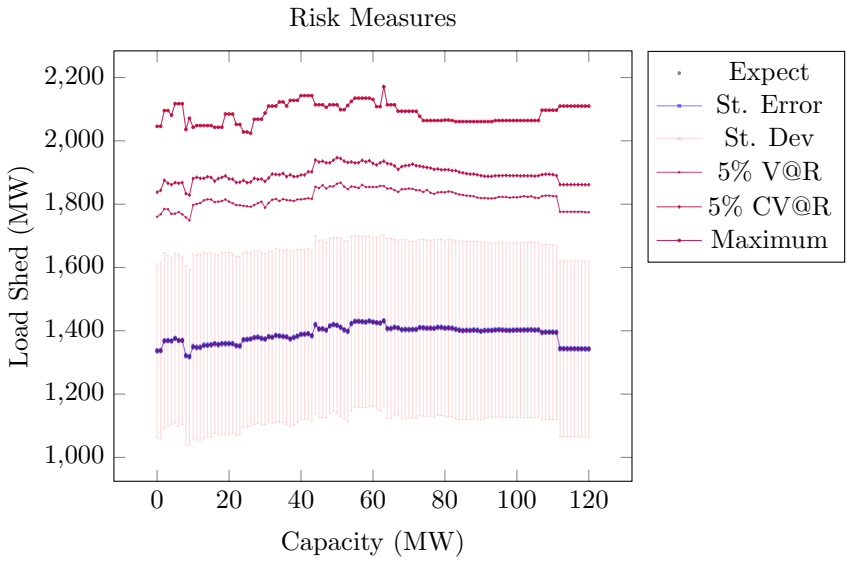


Figure 3.2: Value at Risk, Conditional Value at Risk, and Maximum risk measures for load shed distribution

to the simulation routine. All of the other parameters, which are used in the cascade Algorithm 1 are defined in a <.in> file and an example is showed in Listing 3.1. This includes the line failure distribution parameters L and p as well as a scenario tree definition, which connects this simulation to the MSIP model in the previous Chapter. The scenario tree is built by defining the number of stages and how many outcomes per node. The number of outcomes per node is allowed to vary depending on the stage of the simulation. Since this is a simulation and we are decomposing the scenario tree structure, we typically use a large number of outcomes at the root node and then resolve these completely. The number of initial contingencies is the parameter called scenarios and for each scenario the initial outages are defined in the Outage section. Finally, trials is how many times the scenario tree is solved, so that a 100 child node tree with 500 trials results in 50,000 samples of a full cascade simulation. Finally, a seed is given for the random number generators so that a common random number scheme can be used if desired.

The solve methodology takes information from the parent node in the scenario tree about the new topology required, relaxes the required branch flow constraints, solves the problem, and

File		Size	Description
<.gr>		8K	Defines bus, branch, and generator parameters (grid2.gr ≡ IEEE 118 bus,8K)
<.in>	Listing 3.1	4K	Simulation parameter definitions
<.cap>	Listing 3.4	4K	Capacity addition file
<.dem>		80K	The load shed from every trial cascade simulation
<.lin>		2.1M	The line number for every time a line has failed in a cascade during all the trials
<.lsa>	Listing 3.5	4K	Statistical analysis of load shed from cascade trials in <.dem>
<.lao>	Listing A.9	4K	Count of line outages from cascade simulations output in <.lin>

Table 3.2: Data files used in parallelization routine

stores the results in a tree data structure. The process repeats until all nodes in the scenario tree are resolved. If no lines fail in a node of the scenario tree, the child nodes are truncated since the OPA cascade algorithm has ended. After a scenario tree is solved, the final nodes of the tree are looped over and the load shed of that sample is output into a <.dem> file and each line number that has filed is output into a <.lin> file. These raw data files can be large and there are python scripts to do a load shed analysis on the <.dem> file to give the risk measures associated with the load shed distribution in a <.lsa> file. The <.lin> file is counted with a python script and output in a <.lao> file in order to find out how many times a given line has failed over all the trials. These input and output files are tabulated in Table 3.2.

```
#Problem and Solve Method
solvemethod=cplex
problemtype=sim
lineoutage=yes
#Power Grid
gridfile=grid2.gr
#Line Failure Distribution
p=.5
L=.95
#Scenario Tree
stages=18
outcomesV=100 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
scenarios=4
trials=500
#Outages
a=12 14 34 11
b=12 2 21
c=82 5 25 34
d=13 24
#Random Numbers
```

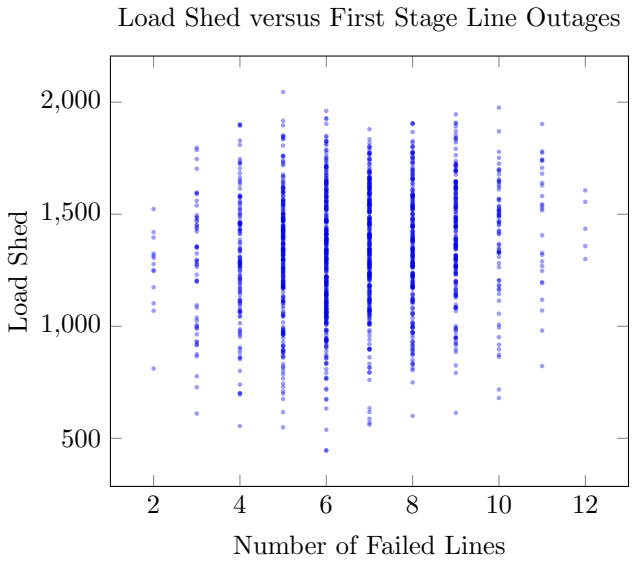


Figure 3.3: The poor relationship between load shed and the number of lines that failed in the first stage.

```
21 seed=689294
```

Listing 3.1: simcplex.in: Parameter definition file

3.1.3 Surrogate Functions

The OPA cascade simulation requires many LP solves in order to simulate the cascading process for many trials. One potential way to reduce the computational effort needed would be to develop a surrogate function which is correlated with the expected load shed function we are trying to minimizing. If we could find a surrogate that was cheap to compute and by minimizing the surrogate function, the expected load shed function would also decrease, this function could be used in the optimization procedure in many ways. For our surrogate function, we consider using information resolved in the first stage of the cascade algorithm.

We used the OPA simulation to look at potential first stage approximators of the OPA cascading process. The two potential approximators we look at are the load shed after the first stage in the cascade and the number of lines that have failed after the first stage of the cascade.

We begin by looking at the number of lines that have failed after the first stage of the cascade. Figure 3.3 shows a scatter plot of load shed in the 1st stage with load shed in the final stage. Through visual inspection, we see that there is little to no correlation between the number of failed lines in the first stage of the OPA simulation and the final load served. This may be due to the difference of importance of transmission lines in the network; we explore this idea in the final Chapter 5.

Next, we explore whether or not the load served at intermediate stages of the cascade has any correlation with the final load served. In Figure 3.4 we see the progression of the load shed over the course of the cascade. These scatter plots chart the load served, out of a potential 3,668 MW. At stage zero the load served is 3,668 for all scenarios and we see the distribution of the final load served between about 1,500 and 2,500 MW. After the first stage, the load served for some scenarios drops below the nominal load. However, the amount of load served after the first stage does not appear to be well correlated with the final load served. As the stages progress up to 20, we see that the load served in intermediate stages of the cascade give little insight into whether it is a bad cascade or not.

3.1.4 Common Random Numbers

The stochastic uncertainty of the OPA process leads to large standard errors for the risk measures. Variance reduction techniques are important to reduce the computational burden and a common random number scheme was employed. Common random numbers essentially give each test system the same set of experimental conditions. By doing so, the variance in the difference between two systems is reduced and less computational resources are needed [31].

The OPA simulation uses effective capacities to determine whether a line fails for a particular

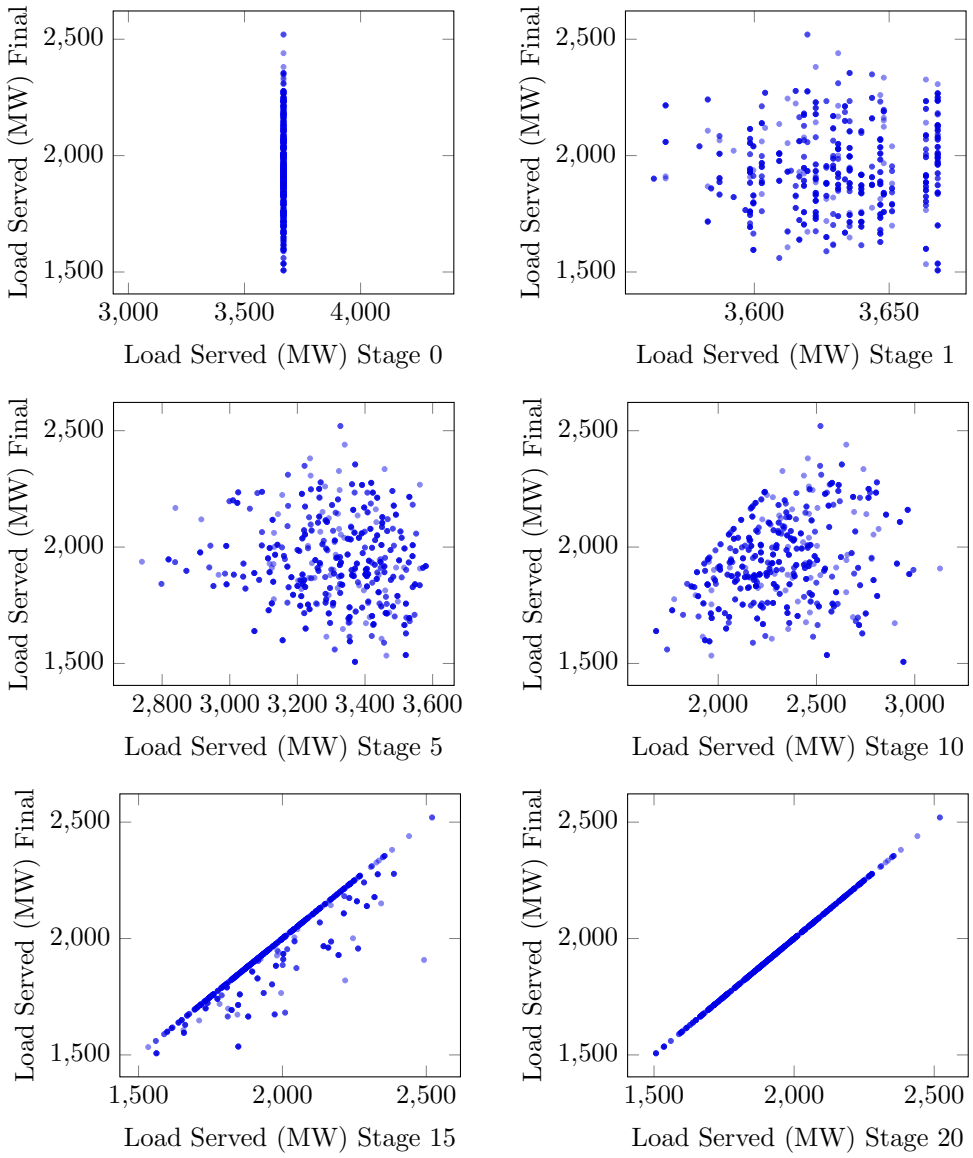


Figure 3.4: Load Served at Different Stages

stage and branch flow loading. In order for the alternative configurations to be under similar experimental conditions, a random number seeding strategy was used to ensure alternative configurations would recieve the exact same effective capacity for that particular stage in that particular node of the scenario tree. Suppose we have two systems with expected load shed L_{ij} for systems $i = 1, 2$ and trials $j = 1, 2, ...N$. Now lets study the metric $Z_j = L_{1j} - L_{2j}$ and let the true comparison be $\mathbb{E}[Z] = \mu_Z$. In order to make decisions about this, we need to be fairly confident in our estimation $\hat{Z}(n) = \sum_j \frac{Z_j}{n}$ of μ_z. The standard error of our sample mean $\hat{Z}(n)$ is

$$Var\left[\hat{Z}(n)\right] = \frac{Var[X_1] + Var[X_2] - CoVar[X_1, X_2]}{n} \tag{3.5}$$

In order to reduce the variance, we need $CoVar[X_1, X_2] > 0$. Be using a common random number scheme giving the alternative system configuration the same experimental conditions, we end up with a postive covariance. This makes sense, as a sample path with consistently low effective capacities should do worse in all systems. In practice, this common random number scheme is very successful for our problem and makes comparing systems less costly.

3.1.5 Capacity Addition

We now explore the landscape of the function with respect to capacity additions by looking at the results from simple changes to the design variables. We start by looking at what happens when you add capacity along the coordinate directions. We restrict ourselves to looking at lines that already exist and adding capacity to these lines. We do not impose any restrictions on the amount of capacity to add to lines. Thermal line limits are often constrained in order to maintain a minimum clearance between the lines and vegetation or other infrastructure. So, in addition to being able to build a parallel line, it is also feasible to increase this clearance by installing taller

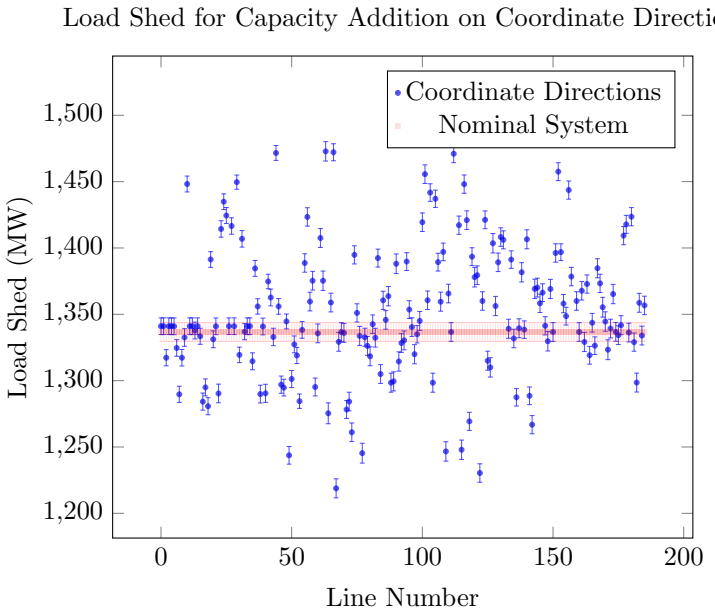


Figure 3.5: Comparing the total load shed for doubling capacity along the coordinate direction.

poles or increasing vigilence in vegetation management.

Our first Figure 3.5 shows what happens if you simply double the capacity on each line individually and do nothing to the rest of the system. The square points are the nominal system and its standard error plotted along each direction and the circle points correspond to the line number in which the capacity was doubled. Perhaps contrary to what would be expected or hoped for, the OPA simulation does not respond favorably in the majority of the coordinate directions and less than 40% of lines lead to a reduction. There may be a few reasons for this. First, by increasing the limit of one line, it may now be possible to overload neighboring lines and increase their likelihood of failure.

Additionally, by looking along just one direction and slowly increasing capacity, we see that the function can change dramatically. Even when the number of trials are increased so that the standard error is low, the function evaluations corresponding to adding capacity on one transmission element are variable and can have large discontinuities. We have plotted three transmission elements

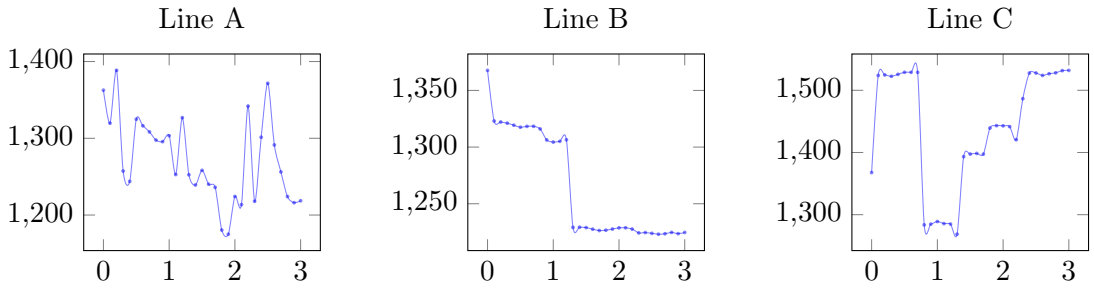


Figure 3.6: Plotting expected load shed for capacity additions along coordinate directions.

in which the capacity is varied from zero to three times its nominal capacity in Figure 3.6. The point in which the capacity is zero is still enforcing the phase angle linkage so that when the branch flow is zero, the phase angles of the two connecting nodes must be equivalent.

Looking further at the ability of increased capacity to influence neighboring lines, we focus on line C from Figure 3.6. Line C highlights the difficulty of optimizing over this function by showing the discontinuities of the OPA simulation with respect to capacity additions. In Figure 3.7, we visually show that these discontinuities can be associated with changes in the frequency of line failures for lines in the given cascade simulations and additionally the frequency of line failures may be correlated with nearby lines. Here we see multiple jumps in the expected value of load shed, and during the last jump, we also see a spike in frequency of line failures for a subset of the transmission lines that are in the same region of the transmission network. The frequency of line failures can often be lined up with corresponding changes in expected load shed.

Finally, we look at a surface plot of adding capacity on separate branches in Figure 3.8. Exploring the effects of adding capacity to lines 67 and 79, we vary the amount of capacity from zero to a very large number. We then truncate the pictures at the point where the additional capacity has no more effect on the OPA simulation and show only the part where changes occur. We see that the surface is bumpy, that is it has some higher frequency effects with low amplitude. Additionally, we note there are larger trends that have much higher amplitude effects. It is important for any

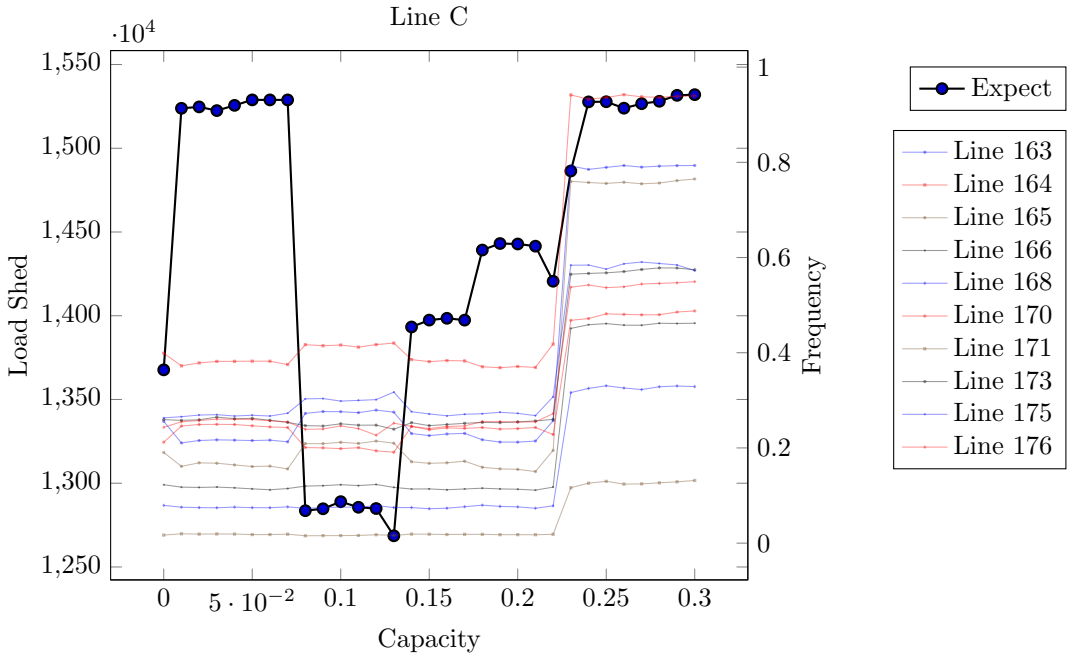
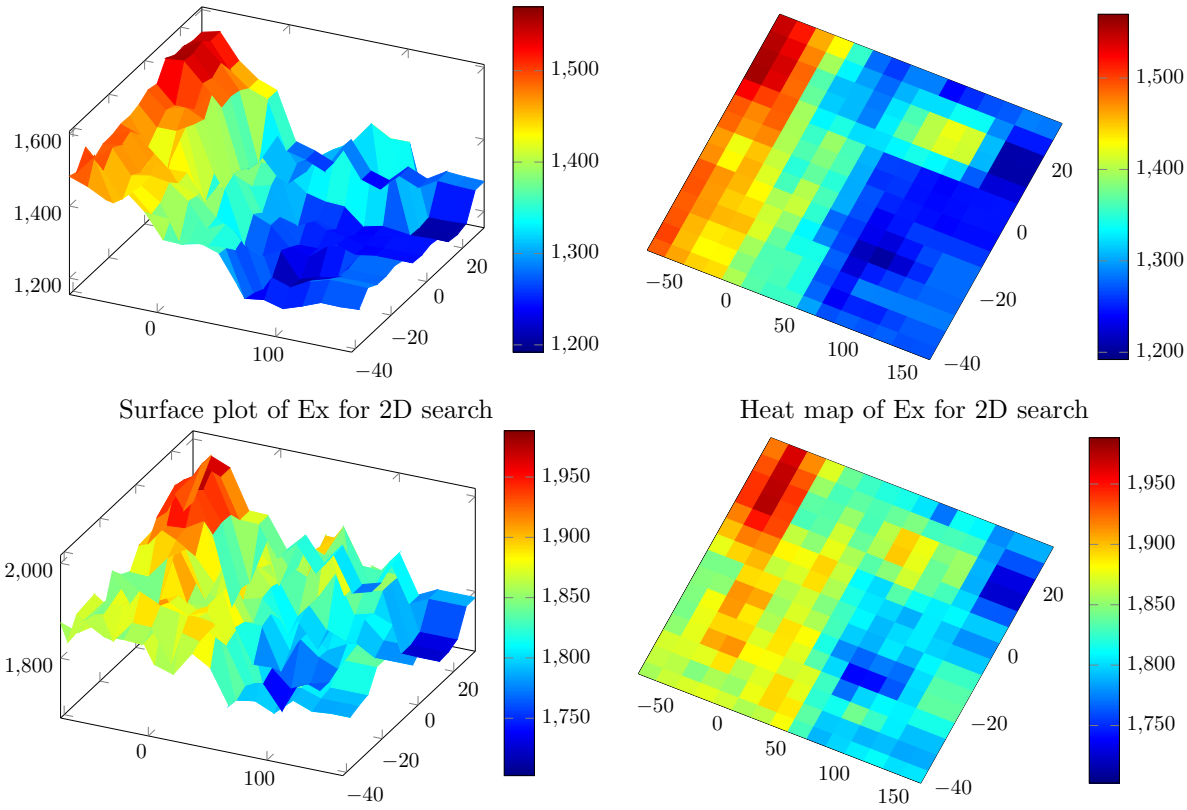


Figure 3.7: A cluster of lines correlated with reduced system performance

optimization method developed to be robust to these high frequency bumps so that they do not get trapped in local minima. Additionally, the conditional value at risk is plotted as well for this same set of trial points in Figure 3.8. The CVaR plot for the expected value of the 5% tail has some similar trends and minima as the expected value, however it differs in some notable ways as well. The expected value has a large region in which the value is near minima. In the CVaR plot, the minima is a smaller region. Additionally, the CVaR also has more high frequency bumps creating local minima and maxima. This may be true of the actual function, however, since this is measuring the tail of the distribution, it may be related to sample size as well.

3.2 Optimizing Design Problems using Pattern Search

This section gives an overview of derivative free optimization. The foundation for direct search and model based search of DFO techniques is explored to be used in solution methods for the OPA optimization procedure. The DFO field has been around for some time now and has seen a



Surface plot of Ex for 2D search

Heat map of Ex for 2D search

Surface plot of CVAR for 2D search

Heat map of CVAR for 2D search

Figure 3.8: 2d heat map of expected load shed and conditional value at risk for adding capacity to two different lines, 67 and 79

resurgence over the last decade and a half with the only textbook at less than 5 years old (Conn, Scheinberg, Vicente) [14]. Kolda et. al. provide an extensive work on direct search methods and its extensions in [29].

This class of optimization strategies covers a wide-range of problems and techniques. In general, the problem has a function $f : \mathbb{R}^n \to \mathbb{R}$ that takes a decision variable $x \in \mathbb{R}^n$ and returns a scalar.

$$\min_{x \in \mathbb{R}^n} f(x) \tag{3.6}$$

The primary problem attribute that makes DFO a good choice for solution method is that standard gradient or Newton based method do not work. This can be due to a variety of reasons, a common one being simulation-based optimization. Here, the derivative is unavailable symbolically (through hard work or automatic differentiation schemes) and, perhaps due to stochastic or numerical noise, is unable to be calculated with finite difference methods. Even if the underlying function is smooth, a costly function evaluation may make the finite difference approach undesirable due to the considerable time to calculate full gradients.

While still useful for smooth functions with a Lipschitz continuous derivatives, these methods can shine in nonsmooth and even non-convex application. Direct search methods work by searching in many directions in order to guarantee a descent direction is chosen. This has the ability to provide robustness against noise that may mislead gradient based methods using a single search direction. In addition, by using relatively large step size the trial points provide a smoothing affect to the function which allow it to ignore high-frequency noise until it is close to a lower-frequency, higher amplitude minimum. Contrary to some problem applications, most convergence results depend on assuming a smooth function with a Lipschitz continuous derivative. No guarantees can be made for the nonsmooth problems, however in practice these techniques are relatively successful for this

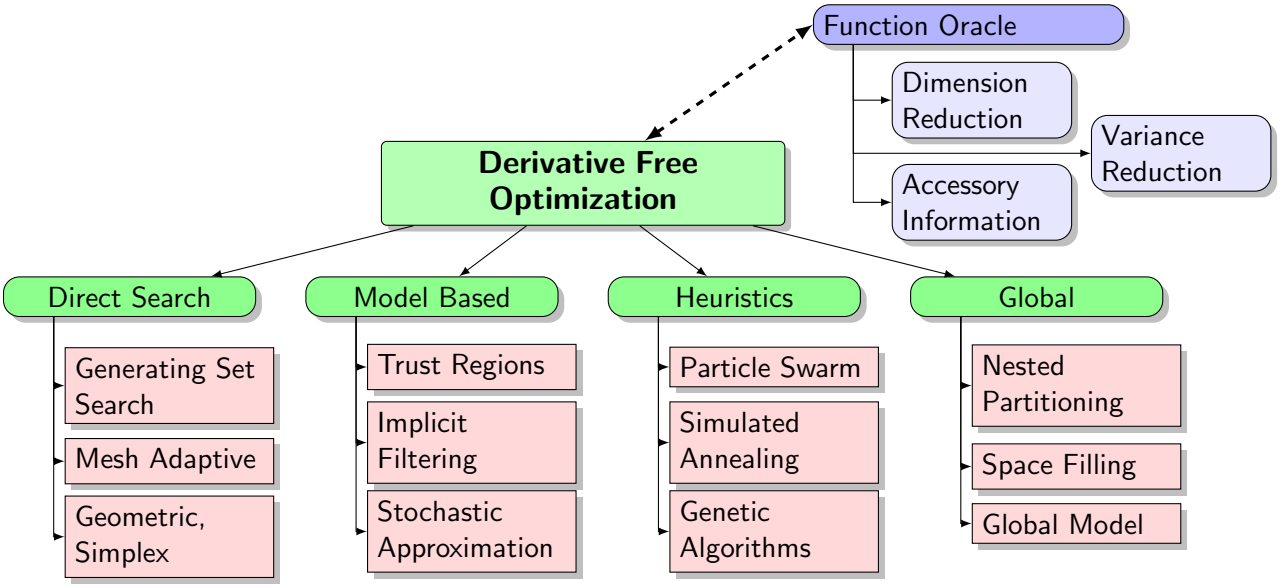


Figure 3.9: DFO methods for continuous variable optimization

class of problems.

3.2.1 Direct Search

The convergence of direct search methods for well behaved functions depend on positive spanning sets as well as the cosine measure are used to bound the angle between the polling directions and the negative gradient. Using the subsequence of unsuccessful iterates, the trial steps become arbitrarily small and x approaches a limit point.

A positively spanning set $\mathcal{G}$ of $\mathbb{R}^n$ can write any vector $v \in \mathbb{R}^n$ as a positive combination of points $d_i \in \mathcal{G}$, $\beta_i \geq 0 \forall i$

$$v = \sum_i \beta_i d_i \tag{3.7}$$

Kolda, Lewis, and Torczon [29] call this a generating set of $\mathbb{R}^n$ which makes the foundation for their class of generating set search methods. This is a large class of problems which generalize lattice methods of Berman [6] [7] as well as their own older methods [46] [32] and include the original

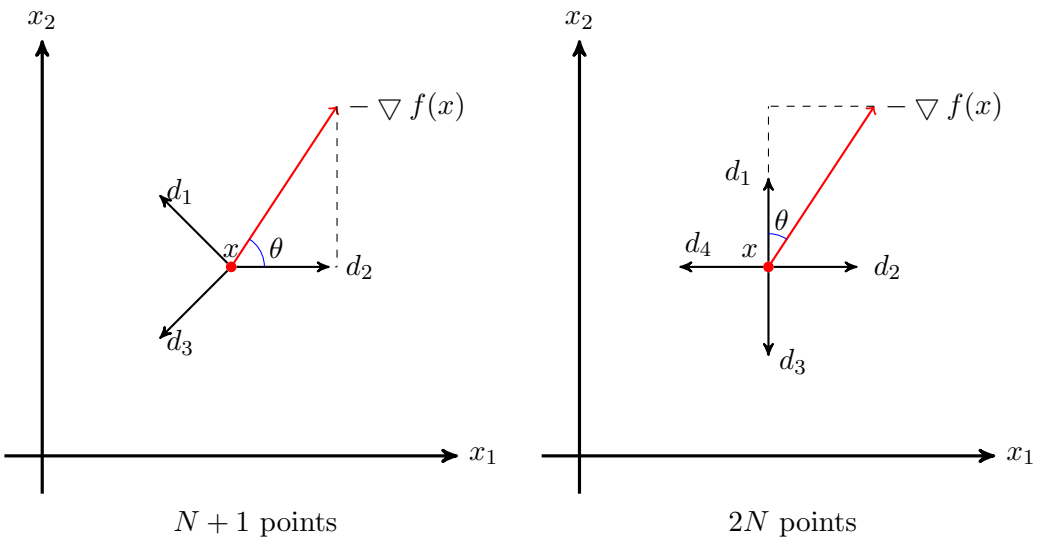


Figure 3.10: Positive spanning sets for $\mathbb{R}^2$

Hookes and Jeeves method[27]. Generating sets can be adapted for explicit linear constraints to conform to local topology.

We only concern ourselves with the nearby active constraints and a satisfactory set of polling directions would be a positive spanning set of the tangent cone of the nearby active constraints. The compass search for standard bound constraints conforms to the constraints ideally and no modifications need to be made other than letting $f = \inf$ for x out of bounds and not waste the time on the function evaluation.

By searching in all directions of a generating set of $\mathbb{R}^n$, we are guaranteed to have a direction which is somewhat aligned with the descent direction f, if it is smooth and has a Lipschitz continous derivative. To make this matter concrete, the cosine measure of a set is the worst case scenario for the descent direction aligning with any direction of the generating set $\mathcal{G}$.

$$\kappa(\mathcal{G}) \equiv \min_{v \in \mathbb{R}^n} \max_{d \in \mathcal{G}} \frac{v^t d}{||v||\,||d||} \tag{3.8}$$

This can be calculated for various generating sets, for example the compass search generating set

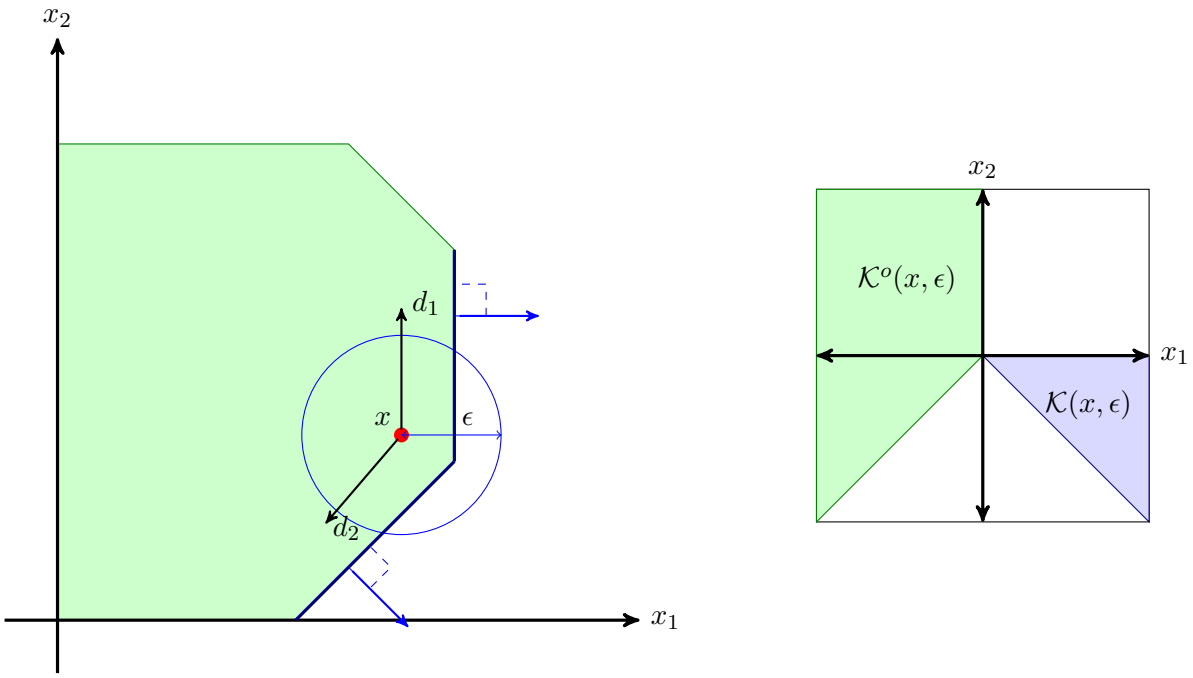


Figure 3.11: Generating set for polar cone to conform to local explicit constraints

gives $\kappa(\mathcal{D}_\oplus) = \frac{1}{\sqrt{n}}$ where n is the dimension of the problem. This begins to show why this method struggles as the dimension of the problem increases. The cosine measure of the search directions must be bounded below to ensure the search directions do not deteriorate.

Algorithm 2 Compass search, a generating set search

procedure CS($f : \mathbb{R}^n \to \mathbb{R}$)
  $x_0 \in \mathbb{R}^n$ Initial guess
  $\triangle_{tol} > 0$ Termination criteria
  $\triangle_0 > \triangle_{tol} > 0$ Initial neighborhood
For each $k = 1, 2, ...$
  Let $\mathcal{D}_\oplus = \{\pm e_i | i = 1, ..., n\}$ be the set of coordinate directions
  if $\exists d_k \in \mathcal{D}_\oplus$ such that $f(x_k + \triangle_k d_k) < f(x_k)$ **then**
    $x_{k+1} \leftarrow x_k + \triangle_k d_x$
    $\triangle_{k+1} = \triangle_k$
  else
    $x_{k+1} \leftarrow x_k$
    $\triangle_{k+1} = \frac{1}{2}\triangle_k$
    if $\triangle_{k+1} < \triangle_{tol}$ **then** terminate

As long as our step size goes to zero $\Delta_k \to 0$ as our work effort increases $k \to \infty$, we converge to a stationary point. There are two primary ways to ensure this happens. The first is to use a

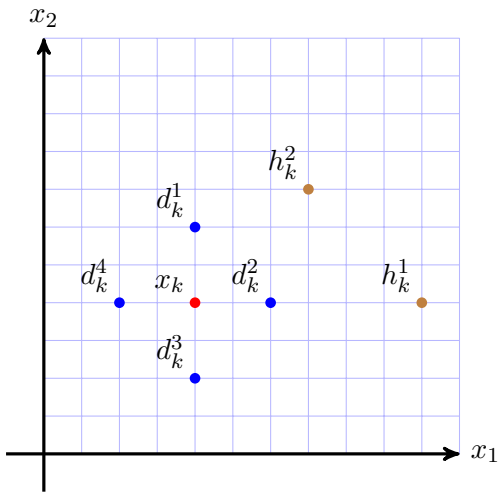


Figure 3.12: Polling step with exploratory trial points on a rational lattice

forcing function that constrains the trial step to have more than simple decrease. Another way is to ensure all the trial points lie on a rational lattice. If all the trial points are then integer combinations, all future trials lie on this lattice. As it is rational, only a finite number of points are evaluated before there is an unsuccessful iteration. As there are only a finite number of evaluations in between unsuccessful steps and at each unsuccessful step the step size is reduced, that step size converges to 0. In Figure 3.12 we see an example of a polling step with exploratory trial points on a rational lattice. For a more detailed convergence proof for standard compass search and the more general generating set search, Kolda et. al. [29].

Flexibility

This class of direct search methods was chosen because of the flexibility in its framework. As long as the trial points lie on a rational lattice, the method converges to a local stationary point. This means that the lattice can be rotated to conform to local topology, it admits exploratory points, and can even use model based methods to improve its search direction and exploratory steps. This includes aligning search directions with approximate gradients that can be calculated using simplex

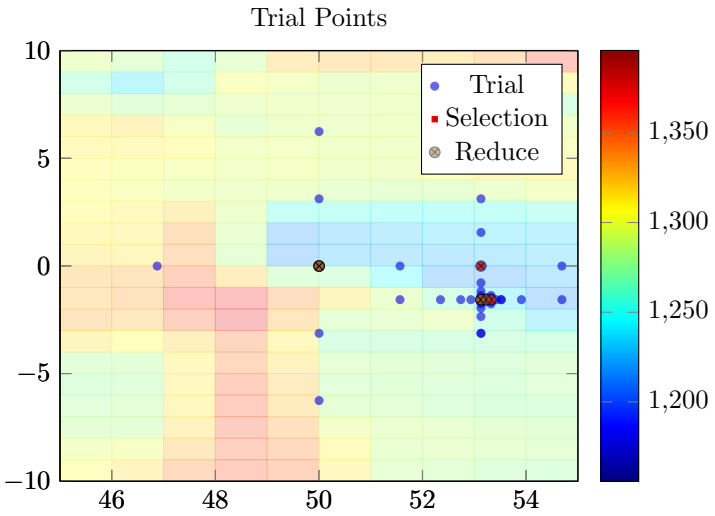


Figure 3.13: Trial exploration using standard compass search with initial step size 50

directions or previous trial points. This framework allows us to test using accessory information to speed up the optimization procedure and compare its effects to model based methods.

To get an idea of the strengths and weaknesses of direct search on the OPA simulation, the standard compass search algorithm was implemented on a reduced 2 dimensional subspace. In Figure 3.8, we saw the rough search space and the need for filtering higher frequency effects and similar properties for conditional value at risk. In fig. 3.13, we se the compass search gets trapped in a local minima. If a small step size is used, the search gets trapped in nearby minima when more progress can be made elsewhere. We ran 4 different compass searches with initial trial steps of 1, 25, 50, and 75 and the final solution points are plotted in Figure 3.14. Large step sizes tend to improve final solution results and it is important to note that they all found very different solutions. This highlights the need for a strategy to find local minima that are nearer to the global minimum.

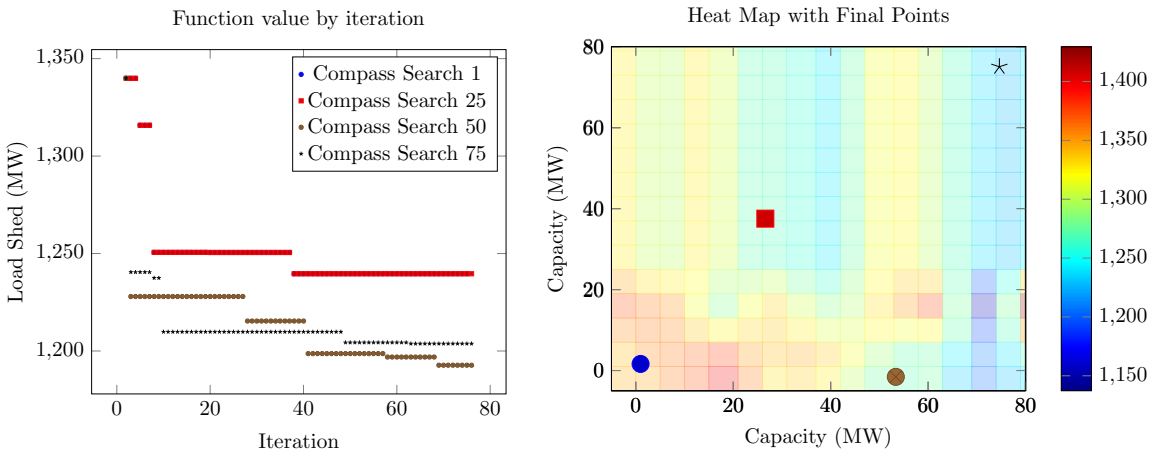


Figure 3.14: Resulting points from compass search with different initial step size values

3.2.2 Line Search Breakpoints

To improve the efficiency of the search, we can use information from the actual line flow statistics for different stages in the cascade. Changes with respect to 1st stage line flows can identify changes in final stage load shed, although not whether it improves or degrades the performance.

We need an algorithm that can handle the noisy output of the OPA model and uses the accessory information in the simulation. This information includes the line with the most failures, clustering, topology information, electrical properties, and correlations between lines and load shed. Using direct search as the foundation, we modify the exploratory steps to take advantage of this information. This provides local convergence guarantees while giving more robustness to the solution methodology.

Additionally, we can find the point in which adding additional capacity won't have any effect. The following equation

$$\nu_e(u) = \max_{\Xi, \Omega} \{\boldsymbol{y}_{es}\} \tag{3.9}$$

$$s = 0, 1, ..., s^*$$

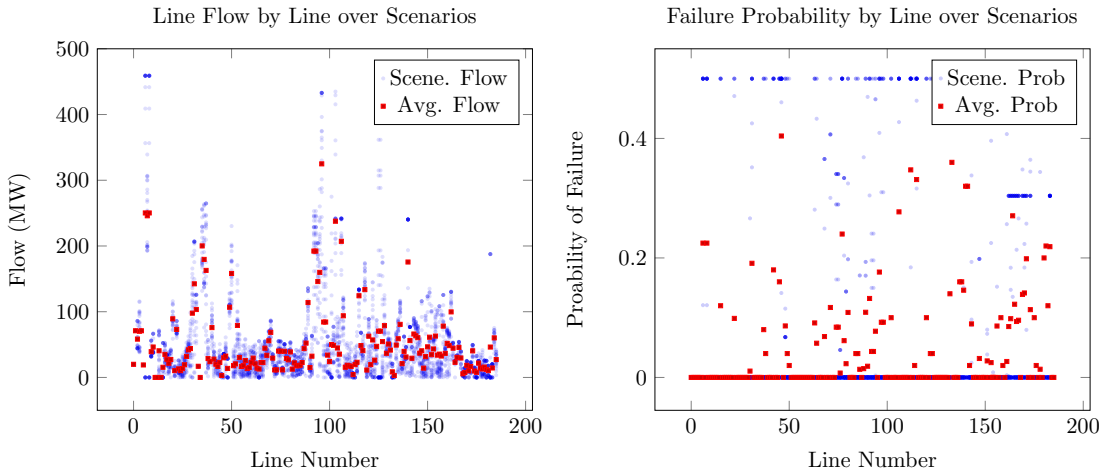


Figure 3.15: Line flows for each scenario and their averages

represents the maximum flow over all stages and trials of the cascade. By choosing u such that the line has no chance to fail, any additional capacity does not change the outcome.

In order to globalize the serach process, we take exploratory trial points along specific directions. We would like to maximize how much we learn with each trial point. In order to do that, we look at how different two systems are. In order to compare the systems we use first stage line failure probabilities.

$$\tau(y^1, u^1, y^2, u^2) = \left[\sum_{e \in \mathcal{E}} \left(g_e(y_e^1, u_e^1) - g_e(y_e^1, u_e^2) \right)^2 \right]^{\frac{1}{2}} \tag{3.10}$$

This information can be used to tell if two operating points have different cascading properties. By looking at the distance between two operating points, breakpoints can be found where the distance between operating points risk characteristics are extremely different from another, close, operating point.

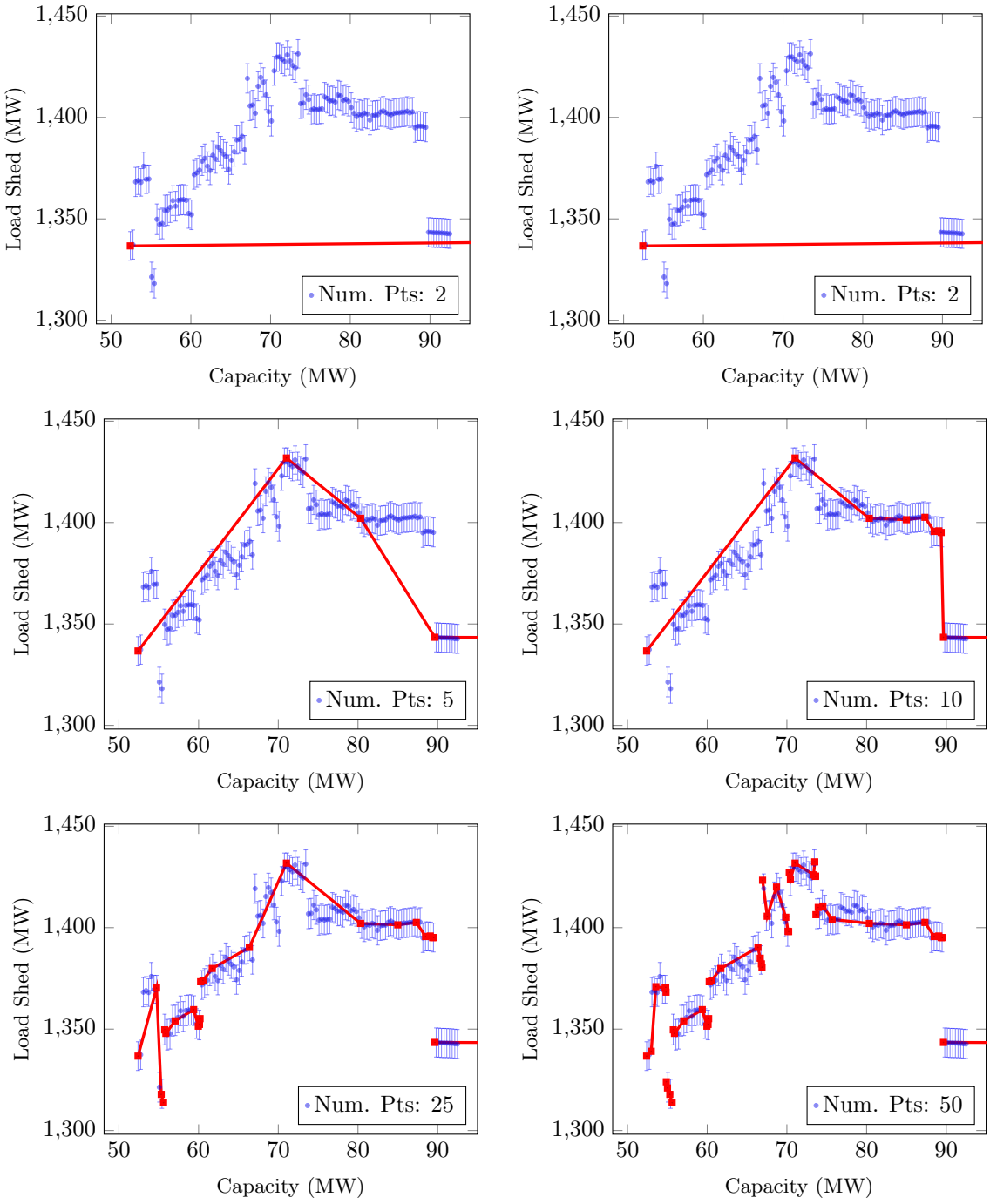


Figure 3.16: Approximation of function by finding breakpoints

3.2.3 Implementing Parallelization in HTCondor

The Center for High Throughput Computing provides computation resources for UW and affiliated researchers. Jobs can be submitted through HTCondor [44] which manages the collective pool of around 1 million CPU hours per day. Users submit jobs to the cluster, which assigns resources that process the job. Requirements can be given to ensure that the resource is capable of performing the job. In order to get access to a larger portion of the cluster, low memory and disk requirements help. Overhead associated with the process, such as being assigned resources and data transfers can be minimized but not removed. As such, the workflow was designed for job times of around 5-30 minutes and perform all analysis locally with the raw data to reduce network data exchanges.

The majority of the HTCondor cluster and UW-Madison uses the Scientific Linux distrubution with version 5 and 6. The submit node used had SL6 installed, and I compiled the C++ binary directly on that machine. In order to access the portion of the cluster which uses SL5, I had to find a remote resource with SL5 installed and build the binary remotely on that machine. HTCondor gives a way to have an interactive session the remote resource. In the HTCondor submit file (Listing 3.2) I request an interactive session and restrict the remote resource to be a MatlabBuildJob, which ensures an SL5 remote resource. I transfer the source code and associated libraries as a tar file, unpack and build on the remote resource and close the session, which initiates a transfer of all newly created files, including the SL5 binary for the C++ code.

```
1  universe = vanilla
2
3  output = process.out
4  error = process.err
5
6  log = process.log
7
8  +IsMatlabBuildJob = true
9  requirements = IsMatlabBuildSlot
10
11 +WantFlocking = true
12
13 should_transfer_files = YES
14 when_to_transfer_output = ON_EXIT
15
16 transfer_input_files = source_code.tar.gz
```

File		Diagram	Description
proc.py	Listing A.1	HTCondor Queue Reader	HTCondor queue reader and command instructor
runit	Listing A.2	Runit Daemon	Principal process flow manager
allocate.py	Listing A.5	Pattern Search Logic	Pattern search logic
consub.py	Listing A.6	Dag File Structure	Create condor submit structure based on given search points from pattern search logic
CONTROL	Listing 3.3	CONTROL: job t	A condor file of bash commands to run on the remote resource
countLines.py	Listing A.8	Data Analysis	Take <.lin> file and count number of line outages
loadShed.py	Listing A.7	Data Analysis	Take <.dem> file and do statistical analysis of load shed

Table 3.3: Scripts and command files used in parallelization routine

```
17
18 notification = never
19
20 queue
```

Listing 3.2: inter-sl5.cmd: HTCondor submit file to run an interactive session on an SL5 machine

After having the main C++ program compiled for two primary linux kernals, we can also access other HTCondor networks through the Open Science Grid and a simple flag WantFlocking ([21]). This allows us to have a large base of computational resources to request to do our job. However, many of these resources are memory and storage constrained. In order to tap into these resources, we restrict the memory requirements of our computational process to 500MBs and storage at 3GBs, which gave plenty of buffer room for program operation and still allowed the capture of the majority of the resources. The large data output from the OPA simulation includes stage by stage details of net power injects, branch flows, and load shed. This data is then analyzed on the remote resource using python script files to find the risk metrics of the load shed as well as any accessory information needed in the optimization algorithm. The output of the analysis is less than 8K for load shed and outage data that is needed for most of the optimization routines.

Now, we begin to outline the parallelization routine used with condor to optimize the capacity

expansion problem using the OPA simulation to evaluate rare event stress. In Figure 3.17, the main process flows are represented. The rectangles represent processes or scripts that create or modify data. The clouds represent data used in these processes, however some files are omitted from this diagram in exchange for overall process clarity. Dashed lines are initiating events or creation of files from the processes and dashed lines are data transfers, inputs, or structural relationships between data. The diamond decision box represents the daemon that keeps the overall parallelization algorithm running. The principal scripts used in these processes are tabulated in Table 3.3.

The runit daemon begins by determining whether the optimization routine is currently running. This allows the process to be started and stopped at will and ensures robustness to loss of network connection. The daemon queries the proc.py script to check the condor queue and the current folder structure to see if any jobs are currently running or there has been steps taken in the optimization routine. If there is, it continues where it left off, otherwise it initiates the first step in the optimization algorithm. It has the option to give a set of initial lines to search from function improvement, otherwise it evaluates the nominal system. The condor file structure is made, the job is submitted, and the daemon waits for the results.

After the process set up, we begin a standard iteration. The daemon waits for the HTCondor queue to become empty and when it does it begins to initiate processes. It starts by summarizing the output to be used in the pattern search logic. After this is done, the pattern search logic determines which lines should be searched for improvement and the condor file submit structure is created based on these search directions. The daemon submits the condor jobs and then begins waiting again for their completion.

HTCondor DAGman was used for job submission in the intra-iteration process flow. DAGman allows jobs to be described in a directed acyclic graph which gives control over the order in which

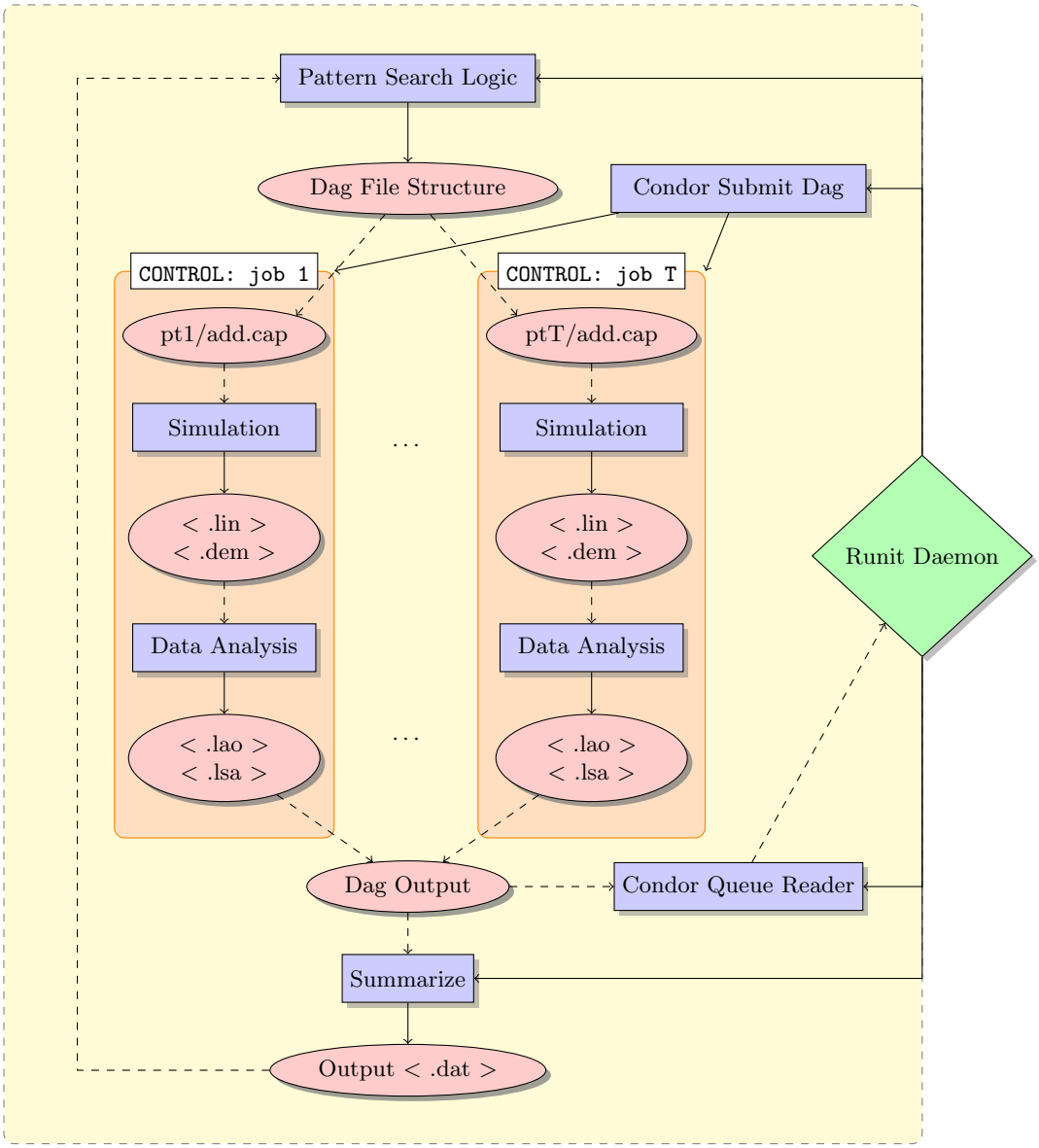


Figure 3.17: Process flow for parallel OPA evaluations and a simple pattern search DFO method. Red clouds represent data and blue boxes represent processes. Dashed lines are for data input or transfers and solid lines are processes or data creation.

jobs are submitted and their dependencies. It also has additional features to be a friendly load on the HTCondor system. In various iterations, there may be over 1000-2000 potential trial points that need to be evaluated. Instead of submitting these all instaneously, these jobs are processed sequentially and have a small delay in between job submission as to not overload the system. In addition, it limits the number of jobs that are sitting idle in the queue, which does not slow down the optimization routine but reduces the demand on the condor job management system. DAGman does add some overhead to the process, however, due to the immense process capabilities of the HTCondor system, these are dwarfed by the sheer amount of computational power you gain. Being a friendly load on the system is a small price to pay.

DAGman uses a very specific file structure in order to submit jobs that have shared resources. The file structure for the input to DAGman is given in Figure 3.18. To make the DAG submission job, Powerin is given as the input directory. Powerin contains a folder called shared that holds the resources needed for every job. DAGman creates a job for each folder inside Powerin and that folder holds the files necessary for that particular job. In our case, we include a <.cap> file which defines the design decision u for that particular trial point. In the shared folder, there is a file called CONTROL (Listing 3.3) that DAGman uses to process each job. CONTROL is a list of shell command to initiate on the host resource.

```
#Run simulation for trial point defined by additions and output file names as DONE
./msip big_sim_cplex.in additions.cap DONE
#Count lines in line outage file and output to want filename
./countLines.py DONE.lin want.lao
#Do statistical load shed analysis on raw load shed distribution and output to want filename
./loadShed.py DONE.dem want.lsa
```

Listing 3.3: CONTROL: commands to run on remote resource

Additionally, the grid definition file, as well as the simulation parameters, are stored there as they do not change from job to job. Packed in sl5.tar.gz and sl6bin.tar.gz are the binaries for those particular instances, and DAGman only brings the correct binary depending on the host resource being used. SLIBS.tar.gz contains the common libraries that are used by the programs and python

scripts. The python data analysis scripts are also used at the remote resource to process the raw data files in order to reduce the network transfers. The REMOVE file cleans up the host resource before the job ends and every newly created file that is not removed is transferred back to the submit node. The other files contained in Powerin are scripts used to create the job folders and run the pattern search logic. After the folder structure has been created, a condor dag job can be created by running the following command.

```
mkdag --data=Powerin --outputdir=step$iteration --cmdtorun=arrive.py --pattern=want.lao --
    pattern=want.lsa --type=Other --maxidle=500
```

DAGman uses the Powerin directory to create the DAG and create an output directory called stepN depending on which iteration the optimization algorithm is on. It creates a job for every folder inside Powerin, except shared. It begins with the script called arrive.py and upon completion, it checks to see if want.lao and want.lsa have been created. If they have not, it is assumed the job has failed and DAGman resubmits the job. Additionally, it limits the maximum number of idle jobs in the queue to 500.

The output folder for DAGman contains subfolders for each job that contain log files as well as the output files that were transferred back to the submit node. After the daemon sees that all the condor jobs are done, it initiates summarization and then proceed to the next step in the algorithm. The overall folder structure for this optimization procedure is given in Figure 3.19. The Powerin folder is cleaned of all job subfolders in between each iteration and new job subfolders are created depending on the new trial points created from the pattern search logic. Important files about the search points used are copied over to the output folders before they are cleaned from the input folder. The output folders can be used to trace what has happened up to the current point in the algorithm in order to allow the daemon to continue off where it last was if it was restarted, either intentionally or unintentionally.

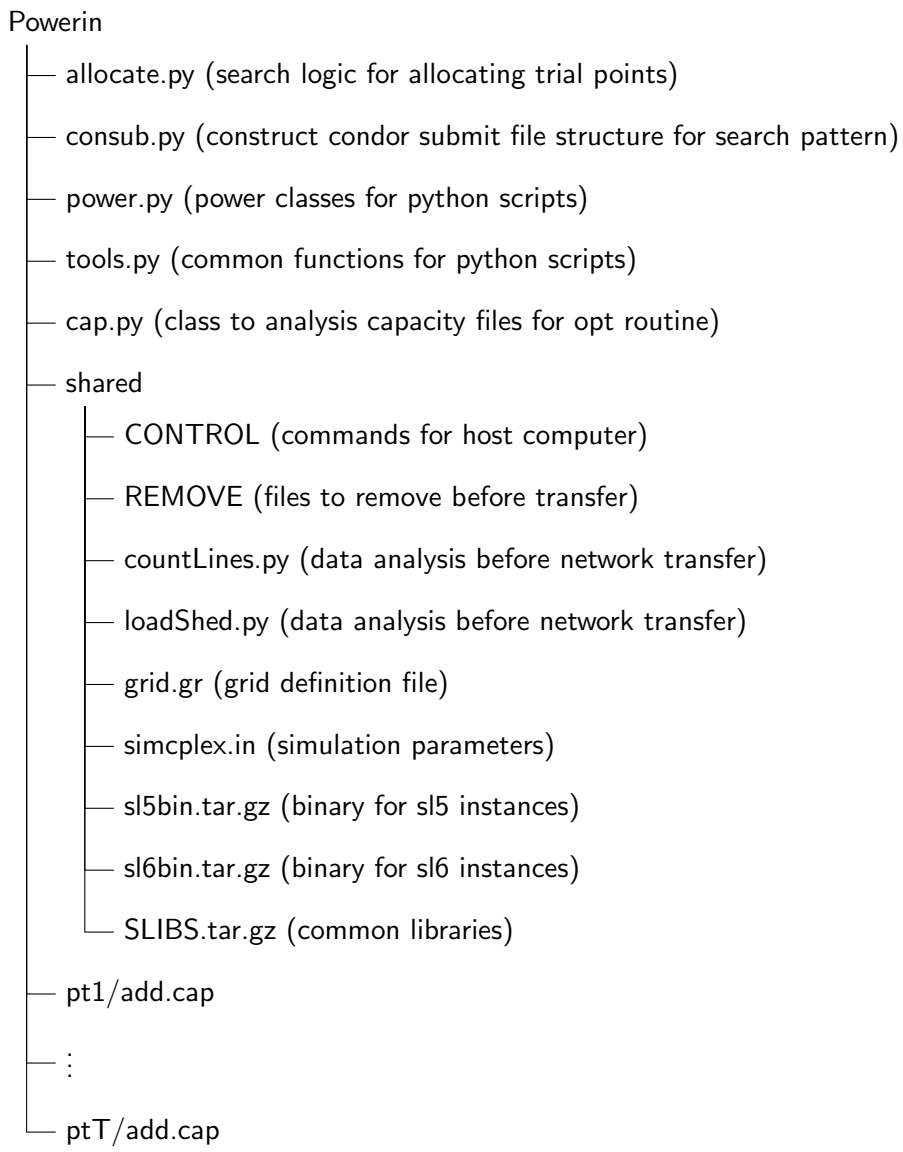


Figure 3.18: Folder structure for DAG input in parallelization routine

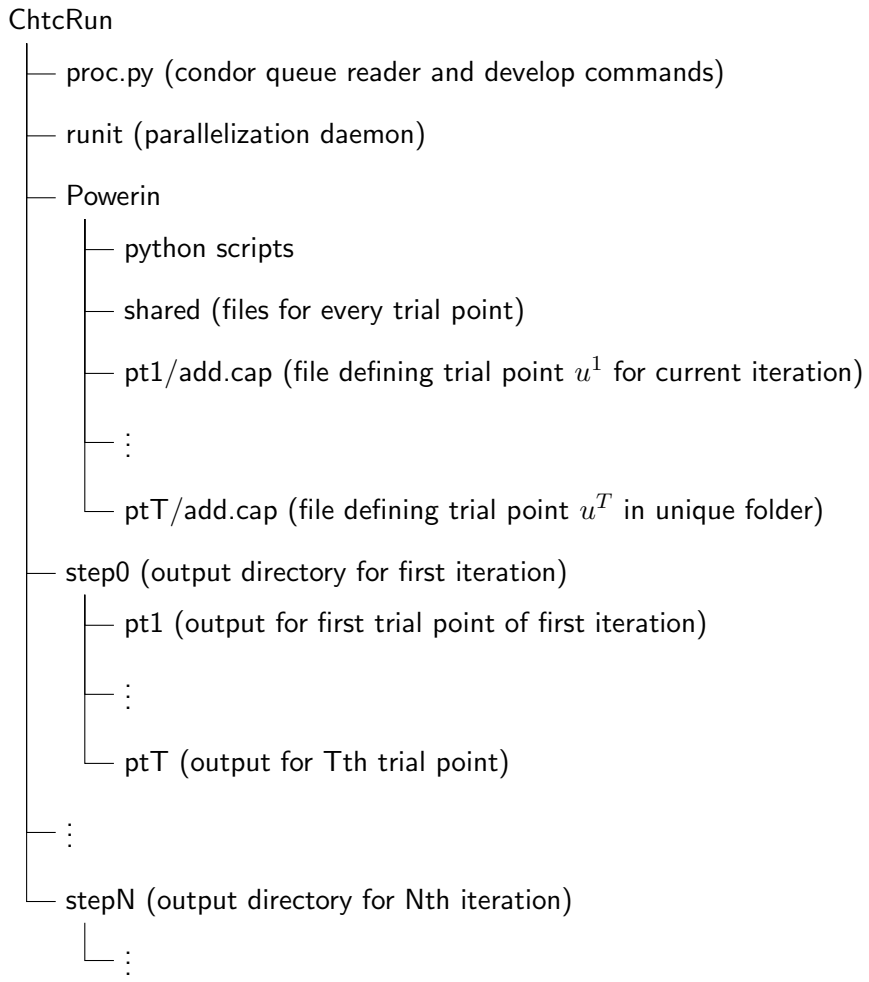


Figure 3.19: Folder structure for using DAGs in parallelization routine

Using a relatively naive pattern search with a fine mesh grid aligned on the coordinate directions, the brute force power of HTCondor was able to achieve significant improvement in the function value. The function value and associated risk measure are plotted in Figure 3.20. The majority of the improvement was made over the initial iterations of the algorithm. The algorithm stopped when the function value made no improvement over the last iteration. The final design point is shown in Listing 3.4 and the risk measures tabulated in Listing 3.5.

```
Design Capacity -individual
22 10.84   40 7.96    46 4.02
48 3.46    49 14.84   67 37.70
68 2.77    71 2.24    76 24.25
79 16.26   80 6.74    84 12.35
85 7.78    102 9.72   112 1.93
113 9.88   115 89.08    123 29.54
124 11.60    132 2.33   133 2.58
138 19.60    151 8.52   164 2.97
165 6.43   169 14.82    170 4.78
171 6.14   180 15.97    181 0.00
```

Listing 3.4: point.cap: Transmission expansion design from pattern search method

```
Samples: 15000
Average: 110.925696667  CI 95% [ 109.185588979, 112.665804354 ]
Standard Deviation: 108.734079904
Standard Error: 0.887810044716
Min: 0.0
Max: 1129.83
95% V@R: 318.56
95% CV@R: 438.24252
```

Listing 3.5: point.lsa: Load shed analysis from chosen design

3.3 Conclusion

In this Chapter, we take a simulation optimization approach to transmission expansion for minimizing load shed due to cascades. We parallelize the computational effort needed in order to evaluate the wide load shed distribution. We continued by exploring the function defined by the expected value of load shed over a small subset of initial contingencies as well as the cascade evolution. We saw that doubling the capacity along coordinate directions often does not lead to reduced load shed. We traced this to overloading neighbors when additional capacity was added to one line but not others. We also saw that discontinuities of load shed could be correlated with the frequency of line

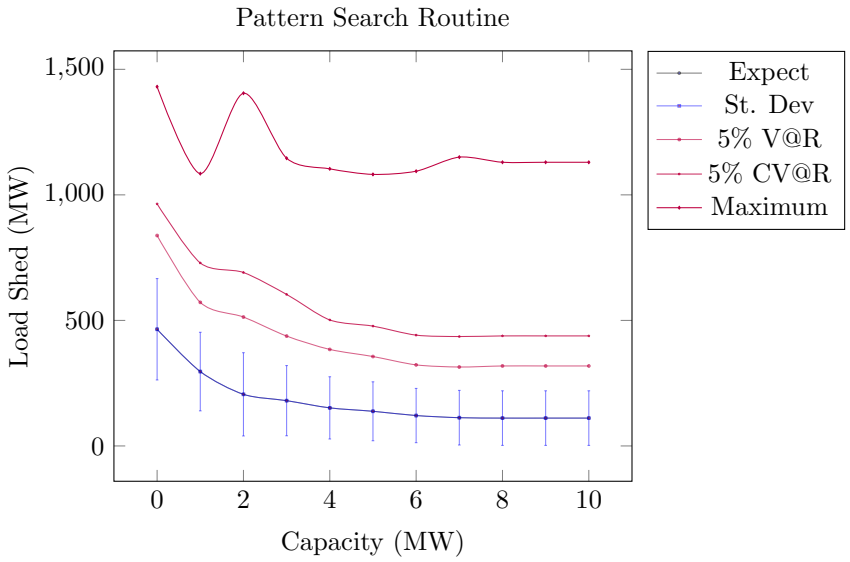


Figure 3.20: Pattern search procedure for transmission expansion design problem

failures in the OPA cascade simulation. We used this to develop a line search procedure that finds these discontinuities. Finally, we implemented the parallelization in HTCondor using the DAGman functionality for job submission. This allowed for job submission smoothing as well as restarting hung jobs or jobs that have failed. We used this parallelization and a naive pattern search with a fine mesh grid to achieve improvement in the function value for the transmission expansion design problem.

Chapter 4

Line Failure Risk Models for Real-Time Dispatch

4.1 Introduction

The use of optimization models for operation of bulk power systems has been critical in creating efficient markets for wholesale power generation over the last few decades. Optimization models are solved to clear multiple markets, including day ahead and real time markets, while maintaining physical constraints related to generator characteristics and power flow constraints on the high voltage transmission system. The nonlinear and nonconvex equations for balanced three phase power flow make these problems particularly difficult to solve and approximations, such as decoupled (DC) power flow, are commonly used in economic models. These optimization models and solution methodologies have become standard tools for independent system operators tasked with operating the bulk power system.

In this chapter, we focus on the operation of the real time dispatch market (five minutes and

less) and the associated reliability issues. Reliability issues have a large impact on the economy, estimated at $79 billion in 2001 [30], relative to the total cost of electricity, $247 billion in 2001 [2]. Additionally, large power outages are disruptive to society, are costly (the 2003 Northeast blackout was an estimated loss of $6.4 billion to the economy), and can lead to the loss of life [45]. Blackout frequency changes seasonally and with the time of day. The load shed from blackout events follows a power-law distribution where large blackout are more likely than expected [24]. This blackout distribution has been stable over the last thirty years and represents a dynamic equilibrium [19, 24]. Small frequent blackouts are associated with small reserve margins in generation and large infrequent blackouts are associated with a highly utilized transmission system [19]. We develop a risk measure that captures the utilization of transmission elements in a systems perspective. This is similar to the use of severity measures to capture line loading risks from a systems perspective [48, 51, 52]. Our model is equivalent to their system risk measure when there is fixed demand and we use a linear approximation of the risk measure. However, we solve our original log-convex risk measure exactly via nonlinear programming. We also extend our model to the case of a multivariate Gaussian distribution for net injection uncertainty where we make a linear approximation to retain a tractability.

There has been increased importance placed on models that account for the growing generation uncertainty due to renewables such as wind and solar. This uncertainty is being handled for the traditional line threshold model of economic dispatch using chance constraints (CC). Several groups have developed chance constrained models to ensure that the power flow on any transmission element is less than its capacity for a large percentage of scenarios. Several extensions are made which involve arbitrary slack distribution [8], application to large penetration of wind farms [49], with HVDC lines [47], and others [39, 48] which is discussed in Section 4.3. Instead of relying on the traditional line threshold, we directly enforce a constraint on the quantity of interest, the

probability that a line will fail. The nominal capacity of transmission elements is based on thermal characteristics of the line as well as a set of environmental characteristics, which represent the worst case scenario for different operating times. Typically, there may be a summer rating and a winter rating for each transmission element, where the winter rating has colder ambient temperature (less heating and less sagging) thus a higher capacity. The limiting constraint determining the line capacities are typically due to the acceptable sagging level derived from a predetermined risk level of fault [42]. Dynamic line limits are being explored to account for real time environmental conditions [10, 50, 53, 54].

In this chapter, we quantify the endogenous risk of line failure due to line loading. Exogenous failure events, primarily caused by weather, such as falling tree limbs taking out circuits, are accounted for through the N-1 security requirements. In order to quantify the endogenous risk, we make several assumptions on the failure density function, which are used elsewhere in literature and have a physical interpretation. Our failure density function assumes that below some loading level there is no endogenous risk associated with the loading and that above a critical level the risk is monotonically increasing. We use a piecewise linear function to model the risk and this model has been used in cascading power failure research in papers [11, 13, 19, 26, 37]. While there is a hard limit on transmission lines that causes them to trip with certainty (due to protective relay elements), the system is always operated far away from these points. If the system was operating close, random fluctuation in power injects would cause the system to trip much more regularly than is seen. Our model is built on the assumption that increasing power flow increases the risk of line failure, and we can only make probabilistic statements about these failure rates.

We combine individual line risks to form a system risk measure that represents the probability that one or more lines fail (due to loading), which we constrain to be small depending on the operators' risk tolerance. The combination of a system risk measure and demand uncertainty

leads to our joint chance constraint (JCC) model. When there is no uncertainty in demand and generation, this system can be solved exactly using nonlinear programming and its linear approximation reduces to other system risk models that have apeared in literature [48, 52]. Under demand uncertainty, the line risk's failure becomes dependent on both the mean and standard deviation of line flow. In order to solve this model, a linearization of the system risk measure is made. Once this approximation is made, the problem becomes solvable for large test instances even when exogenous N-1 contingencies are considered. In section Section 4.2 we look at the standard economic dispatch model, referred throughout as optimal power flow (OPF). Using the DC power flow approximation, we show the linear program used in the real time market with a quadratic cost function. In the following Section 4.3, we begin by exploring the uncertainty introduced into the system by wind generation and variable load. We then show the chance constraints (CC) used in other work to ensure the reliability constraints are met a large percentage of the time. In Section 4.4, we develop a system risk measure defined by the probability that one or more lines fail by assuming the probability a line fails is related to line loading and can be represented by a piecewise linear function. Using this risk measure and the assumption of fixed generation and demand, we can solve this model exactly using nonlinear programming. Finally, we look to combine uncertain wind and load with our system risk measure. Under the assumption of a multivariate Gaussian distribution for wind and load, we find the covariance matrix to enforce the resulting variation in branch flows. The risk constraints are convex with respect to the mean and standard deviation of branch flow. We then describe the solution methodology in Section 4.4.3 which uses an iterative algorithm and cutting planes to describe the convex risk constraints. Finally, we explore the computational results in Section 4.5 to understand the characteristics of the standard economic dispatch model, chance constraint model, and our joint chance constraint model.

In order to solve our problem, a cutting plane algorithm is used to underestimate the line risk

function (convex, not analytic) as well as the branch flow standard deviations (second order cone). This multiobjective problem of cost and risk form a frontier that is shown in the computational section. The OPF and CC models give dispatch points on the interior of this frontier, which are inefficient according to our system risk measure. The CC and JCC models take around the same amount of time to solve and are an order of magnitude slower than OPF. Our JCC model can reduce the expected number of lines over their hard threshold compared to the CC model, which is what the chance constraint model aims to do on an individual line level. Finally, our JCC model is robust to deviations in line risk parameters as well as other failure density function models.

4.2 Economic Dispatch Models

Bulk power systems rely on dispatch models to clear the markets for power in a timely manner. These models are critical to ensure that the market is cleared minimizing cost while maintaining a set of reliability constraints. The DC power flow model is typically used to clear economic markets for both real-time and day-ahead operation, where the day-ahead operation has the additional complexity of committing slow ramping resources and is solved with mixed-integer programming.

4.2.1 Optimal Power Flow using DC approximation

The DC power flow model is a simplification of the AC power flow model, which more accurately represents the true physics of the balanced three phase electrical system. The DC model makes the assumptions that the power lines are lossless, voltages are equal to nominal, and the phase angle differences are small. While losing some accuracy and important information about the voltages, the problem becomes more tractable. Methods can be used to find approximate voltages, which can be important indicators for system stability. Additional complexities are ignored for clarity

such as shunt elements which consume power [56].

The topology of the power grid, with N_l lines and N_b buses, can be represented using an incidence matrix, $C \in \mathbb{R}^{N_l \times N_b}$, where $c_{ei} = 1$ if line e begins at node i and -1 if line e ends at node i. The set of all edges is denoted as $\mathcal{E}$ and the set of all nodes as $\mathcal{I}$. In the DC power flow model, the line susceptance is the constant of proportionaility between bus voltage and liv, so let $D = diag\,(b_1, b_2, ...b_{N_l})$ be the diagonal susceptance matrix. The branch flows $y \in \mathbb{R}^{N_l}$ for given a set of phase angles $\theta \in \mathbb{R}^{N_b}$ at each bus are determined by Kirchoff's Current Law and can be written as

$$y = DC\theta \tag{4.1}$$

for the DC power flow model. This results in the typical DC line constraints $y_e = b_e(\theta_i - \theta_j)$ for each line e connecting from node i to node j. Applying C^T to the branch flows y in Equation (4.1) give the net injects for a given set of branch flows and represents conservation of energy at each node. The system matrix $B = C^T DC$ can be used to write the DC power flow equations for the net injects $x \in \mathbb{R}^{N_b}$

$$x = C^T y = B\theta \tag{4.2}$$

This equation has one degree of freedom and by removing the slack bus inject (row) and phase angle (column), then the following system

$$\tilde{x} = \tilde{B}\tilde{\theta} \tag{4.3}$$

has a unique solution. Here, $\tilde{x} \in \mathbb{R}^{N_b - 1}$ has the slack node removed and $\tilde{B}$ is a $(N_b - 1) \times (N_b - 1)$ matrix with the row for the slack bus inject and the column for the slack bus angle has been removed. The OPF model uses the DC simplifications from the full AC model to ensure the model

can be solved reliably in a timely manner. A standard OPF model is shown here and used in the computational section.

The following program Equation (4.4) is minimizing a quadratic cost function of generation x_j for each generator j in the set of all generators $\mathcal{J}$ with cost coefficients c_j^2, c_j^1 and c_j^0. There are two incidence matricies, c_{ij}^g for generators and c_{ie}^b for branches. c_{ij}^g takes the value 1 when generator j is connected to node i in the set of all nodes $\mathcal{I}$. c_{ie}^b takes the value 1 if the edge e, in the set of all edges $\mathcal{E}$, originates in node i and a -1 if edge e terminates in node i. We also have nominal demand d_i for each node i, branch capacities U_e for each branch e, and generator limits G_j^{min} and G_j^{max}.

$$\textbf{OPF:=} \min_{(x;\theta,y)} \sum_{j\in\mathcal{J}} \left[c_j^2 x_j^2 + c_j^1 x_j + c_j^0\right] \tag{(}$$

$$\sum_{j\in\mathcal{J}} c_{ij}^g x_j - \sum_{j\in\mathcal{J}} c_{ie}^b y_e = d_i \qquad \forall i \in \mathcal{I} \tag{4.4b}$$

$$y_e - b_e \sum_{i\in\mathcal{I}} c_{ie}^b \theta_i = 0 \qquad \forall e \in \mathcal{E} \tag{4.4c}$$

$$y_e \in [-U_e, U_e] \qquad \forall e \in \mathcal{E} \tag{4.4d}$$

$$x_j \in \left[G_j^{min}, G_j^{max}\right] \forall j \in \mathcal{J} \tag{4.4e}$$

4.3 Chance Constraints for Random Branch Flows

As the penetration of renewables increases, there is an increasing interest in the inclusion of uncertainty in dispatch models. The primary approach has been to treat wind generation and load variables as random variables and enforce probabilistic constraints replacing the standard deterministic reliability constraints. Probabilistic constraints, or chance constraints, enforce that a constraint involving a random variable must be satisfied a large percentage of the time. Chance constraints have been used in a variety of power system problems, from a unified model including

large wind penetration and HVDC lines [48], to models using an arbitrary slack distribution [8]. Additionally, their are models that have different assumptions on the probability distributions of the random variables, from the common independent Gaussian assumption [8, 39] to no assumption on the uncertainty except for the ability to sample from it [47, 48, 49].

We look first at chance constraints for branch flows, which begins by looking at the root uncertainty in net injections due to wind and load. The deviation of net injections from forecast leads to a response by automatic generator control to ensure that supply and demand are constantly in balance. Under the DC power flow model, the branch sensitivities to changes in net injections are a linear relationship. This allows us to calculate the uncertainty in branch flows dependent on the uncertainty in wind and load. A common assumption is that the uncertainty in deviation from forecast is a Gaussian distribution, and that its mean and covariance is known. This assumption allows for a tractable formulation of the chance constraint without sampling from the uncertainty distribution.

4.3.1 Random Power Flows

The power injections at each node are subject to random fluctuation over varying time intervals. The variation over the 5 minute time scale are dealt with using ancillary services such as regulation and reserve. The sources of the fluctuations can be due to random demand or generation. These fluctuations affect the power flows on transmission lines and cause a linear shift in the approximated DC power flow model. The risk of the system is dependent on the characteristics of these resulting random power flows. In greater time intervals, the generators can redispatch for economic or reliability reasons.

Over the course of the 5 minute interval dispatch period, demand fluctuates from its expec-

tation. This fluctuation can be due to the behaviors of aggregated residential load, commercial or industrial processes, or power production for renewable sources such as wind and solar. Assuming these random injects are Gaussian, the linear approximations made in the DC power flow leads to the branch flows also being Gaussian random variables. The mean vector and covariance matrix of branch flows can be calculated if the mean and covariance of the random injections are known.

Let $\boldsymbol{\delta^m} \in \mathbb{R}^{N_m}$ be a random vector describing the deviation from forecast on a subset of nodes $k \in \mathcal{M}$ which are random ($|\mathcal{M}| = N_m$). Bold lettering $\boldsymbol{\delta^m}$ represents a random variable with a probability distribution supported on $\mathbb{D}$. Define an incidence matrix $C_M \in \mathbb{R}^{N_b \times N_m}$ where $c_{im} = 1$ if random inject m is connected to node i. Additionally, we define the forecasted demand at each node as $d \in \mathbb{R}^{N_b}$ where N_b is the number of buses. The forecasted demand is supplied by generation $x^g \in \mathbb{R}^{N_g}$ where N_g is the number of generators and the incidence matrix C_g brings them into the bus space. The vector of net injections are defined by

$$\boldsymbol{x} = C_g \left(x^g + \beta \boldsymbol{\Delta}\right) - \left(d + C_M \boldsymbol{\delta^m}\right) \tag{4.5}$$

where the left term is generation/controllable and the right term is load/uncontrolled. Both terms have a fixed component due to the forecasted system and a random component due to the random injections. For the rest of this chapter, we assume the random injects $\boldsymbol{\delta^m}$ are assumed to be a multivariate Gaussian distribution with known mean μ^m and covariance Σ^m.

The random injects cause loading on the transmission lines to fluctuate and the flows are random variables themselves. Taking the derivative of Equation (4.2), we get $d\tilde{x} = \tilde{B} d\tilde{\theta}$. The changes to the phase angles given a vector of changes to net injects is $d\tilde{\theta} = \tilde{B}^{-1} d\tilde{x}$. This can be

used to find the sensitivity of branch flows to net injects

$$dy = Adx \tag{4.6}$$

where A is the injection shift factor matrix $A = DC \begin{bmatrix} 0 & \tilde{B}^{-1} \end{bmatrix}$ and $0 \in \mathbb{R}^{N_b}$ is a column of zeros. For a slack distribution β with $\sum_{j \in \mathcal{J}} \beta_j = 1$ and aggregate demand change $\Delta = \sum_{i \in \mathcal{I}} dx_i$, the net injects also change by $-\beta\Delta$ so that $dy = A\left(dx - C_g \beta \Delta\right)$. The random power flows $\boldsymbol{y}$ can be found by applying Equations (4.1) and (4.2) to the random net injections Equation (4.5)

$$\boldsymbol{y} = y^0 + AC_G\beta\boldsymbol{\Delta} - AC_M\boldsymbol{\delta^m} \tag{4.7}$$

with

$$\boldsymbol{\delta^y} = AC_G\beta\boldsymbol{\Delta} - AC_M\boldsymbol{\delta^m} \tag{4.8}$$

being the random component of branch flows. Without loss of generality, we can assume that the random injects have zero mean. If we know them to have a nonzero mean, we can shift the forecasted system to account for it. The forecasted system $\left(x^g, y^0, \theta^0\right)$ is the expectation of Equations (4.5) and (4.7)

$$C_G x^g - d = C^T y^0$$

$$y^0 = B^T C \theta^0$$

with zero mean $\mathbb{E}_\Omega\left[\boldsymbol{\delta^m}\right] = [0 \cdots 0]^T$ and $\mathbb{E}_\Omega\left[\boldsymbol{\Delta}\right] = 0$. From Equation (4.6), we have that the random component of the flow is made part from the random injects and part the generators response. The random branch flows $\boldsymbol{\delta^y}$ are a linear function of the random injects $\boldsymbol{\delta^m}$, thus $\boldsymbol{y}$ has a multivariate

Gaussian distribution as well. The branch covariance matrix can be calculated

$$\mathbb{E}_{\mathbb{D}}\left[(\boldsymbol{y}-\mu^{y})\,(\boldsymbol{y}-\mu^{y})^{T}\right]=\mathbb{E}_{\mathbb{D}}\left[\left(A(C_{G}\beta\vec{1}^{T}-C_{M})\boldsymbol{\delta^{m}}\right)\left(A(C_{G}\beta\vec{1}^{T}-C_{M})\boldsymbol{\delta^{m}}\right)^{T}\right]$$

where $\vec{1}=\{1\}^{N_m}$ is a vector of ones that is of N_m length. Substituting $G=A(C_G\beta\vec{1}^T-C_M)$, we have

$$\begin{aligned}\mathbb{E}_{\mathbb{D}}\left[(G\boldsymbol{\delta^{m}})\,(G\boldsymbol{\delta^{m}})^{T}\right]&=\mathbb{E}_{\mathbb{D}}\left[G\boldsymbol{\delta^{m}}\boldsymbol{\delta^{m}}^{T}G^{T}\right]\\&=G\mathbb{E}_{\mathbb{D}}\left[\boldsymbol{\delta^{m}}\boldsymbol{\delta^{m}}^{T}\right]G^{T}\end{aligned}$$

Since the expectation is linear and the covariance Σ^m of $\boldsymbol{\delta^m}$ is known ($\Sigma^m=\mathbb{E}_{\mathbb{D}}\left[\boldsymbol{\delta^{m}}\boldsymbol{\delta^{m}}^{T}\right]$), we have power flow covariance matrix, after substitution of G, of

$$\Sigma^{y}=A(C_{G}\beta\vec{1}^{T}-C_{M})\Sigma^{m}(C_{G}\beta\vec{1}^{T}-C_{M})^{T}A^{T}. \tag{4.9}$$

This equation can be found piecewise by taking the variance of the random component of the branch flow for a given branch e and substituting $\pi_e=\sum_{j\in\mathcal{J}}A_{ej}\beta_j$ (the effect on branch e from aggregate deviation from forecast Δ). Reforming this assumption, we have the random component of branch flow $\boldsymbol{\delta_e^y}$ on branch e defined by

$$\begin{aligned}\boldsymbol{\delta_e^y}&=\sum_{j\in\mathcal{J}}A_{ej}\beta_j\boldsymbol{\Delta}-\sum_{k\in\mathcal{M}}A_{ek}\boldsymbol{\delta_k^m}\\&=\pi_e\sum_{k\in\mathcal{M}}\boldsymbol{\delta_k^m}-\sum_{k\in\mathcal{M}}A_{ek}\boldsymbol{\delta_k^m}\\&=\sum_{k\in\mathcal{M}}\left[\pi_e-A_{ek}\right]\boldsymbol{\delta_k^m}\end{aligned}$$

where $\boldsymbol{\delta_k^m}$ is the random inject $k \in \mathcal{M}$. Using the notation that the covariance between random inject k_1 and k_2 is $\Sigma_{k_1 k_2} = \text{CoVar}[\boldsymbol{\delta_{k_1}^m}, \boldsymbol{\delta_{k_2}^m}]$, the covariance between branch e_1 and e_2 is calculated by

$$\text{CoVar}[\boldsymbol{\delta_{e_1}^y}, \boldsymbol{\delta_{e_2}^y}] = \sum_{k_1 \in \mathcal{M}} \sum_{k_2 \in \mathcal{M}} (\pi_{e_1} - A_{e_1 k_1})(\pi_{e_2} - A_{e_2 k_2}) \Sigma_{k_1 k_2} \tag{4.10a}$$

$$\text{CoVar}[\boldsymbol{\delta_{e_1}^y}, \boldsymbol{\delta_{e_2}^y}] = \pi_{e_1} \pi_{e_2} \sum_{k_1 \in \mathcal{M}} \sum_{k_2 \in \mathcal{M}} \Sigma_{k_1 k_2} \tag{4.10b}$$

$$- \pi_{e_1} \sum_{k_1 \in \mathcal{M}} \sum_{k_2 \in \mathcal{M}} A_{e_2 k_1} \Sigma_{k_1 k_2} - \pi_{e_2} \sum_{k_1 \in \mathcal{M}} \sum_{k_2 \in \mathcal{M}} A_{e_1 k_1} \Sigma_{k_1 k_2} \tag{4.10c}$$

$$+ \sum_{k_1 \in \mathcal{M}} \sum_{k_2 \in \mathcal{M}} A_{e_1 k_1} A_{e_2 k_2} \Sigma_{k_1 k_2} \tag{4.10d}$$

$$\text{CoVar}[\boldsymbol{\delta_{e_1}^y}, \boldsymbol{\delta_{e_2}^y}] = \pi_{e_1} \pi_{e_2} \sigma_\Delta^2 - \pi_{e_1} \sigma_{e_2}^2 - \pi_{e_2} \sigma_{e_1}^2 + \sigma_{e_1 e_2}^2 \tag{4.10e}$$

where $\sigma_\Delta^2, \sigma_{e_1}^2$, and $\sigma_{e_1 e_2}^2$ are all pre-computable parameters given by the following equations

$$\sigma_\Delta^2 = \sum_{k_1 \in \mathcal{M}} \sum_{k_2 \in \mathcal{M}} \Sigma_{k_1, k_2} \tag{4.11a}$$

$$\sigma_e^2 = \sum_{k_1 \in \mathcal{M}} \sum_{k_2 \in \mathcal{M}} A_{e k_1} \Sigma_{k_1, k_2} \quad \forall e \in \mathcal{E} \tag{4.11b}$$

$$\sigma_{e_1 e_2}^2 = \sum_{k_1 \in \mathcal{M}} \sum_{k_2 \in \mathcal{M}} A_{e_1 k_1} A_{e_2 k_2} \Sigma_{k_1, k_2} \quad \forall e_1, e_2 \in \mathcal{E}. \tag{4.11c}$$

There are two primary parts to the variance in line flow, that from the uncertain random injects and that from the aggregate deviation from forecast at the slack distributions response. The variance of branch flow on line e is described as follows

$$\text{Var}[\boldsymbol{\delta_e^y}] = \pi_e^2 \sigma_\Delta^2 - 2\pi_e \sigma_e^2 + \sigma_{ee}^2 \quad \forall e \in \mathcal{E} \tag{4.12}$$

When the net injections are uncertain, the branch flows are also uncertain and represented by $\boldsymbol{y}_e$ for branch e, where the bold letter denotes a random variable throughout this chapter. Let ϵ_l be the percentage of time we allow the constraint to be violated for each line l, so that the following constraint

$$\mathbb{P}_\Omega \left[-U_{\leq} \boldsymbol{y}_e \leq U \right] \geq 1 - \epsilon_l \quad \forall e \in \mathcal{E} \tag{4.13}$$

represents the chance constraint version of the nominal line capacity limit. Assuming $\boldsymbol{y}_e$ is a Gaussian random variable with mean μ_e^y and standard deviation σ_e^y. Given a line risk preference ϵ_l, we have

$$\eta_l = \Phi^{-1} \left(1 - \epsilon_l \right) \tag{4.14}$$

where $\Phi^{-1}(\cdot)$ is the inverse of the standard normal distribution. The following constraints

$$\mu_e^y + \eta_l \sigma_e^y \leq U_e \quad \forall e \in \mathcal{E}$$

$$\mu_e^y - \eta_l \sigma_e^y \geq -U_e \quad \forall e \in \mathcal{E}$$

are deterministic constraints for Equation (4.13)

In addition to chance constraints on branch flows, the output of generators may be uncertain. Let $\boldsymbol{\Delta}$ represent the aggregate deviation from forecast that the generators must respond to. The slack distribution is a subset of generators who are able to respond to deviations in forecast in a fast time scale. They typically follow a signal such as the area control error (ACE). Let β_j define this slack distribution, where β_j is the portion of $\boldsymbol{\Delta}$ that generator j compensates for. We know that $\sum_{j \in \mathcal{J}} \beta_j = 1$ for demand to be satisfied exactly. Additional constraints can be put on the slack variables β depending on generator characteristics and their ability to respond to variations in load. The total output for generator j is $x_j + \beta_j \boldsymbol{\Delta}$ and we enforce its constraints probabilistically since

it is a random variable. Assuming that the individual variations in wind and load are Gaussian, their aggregate is also Gaussian. Let ϵ_j be the probability that the constraint can be violated for generator j and we would like a constraint to enforce

$$\mathbb{P}_\Omega \left[G_e^{min} \leq x_j + \beta_j \mathbf{\Delta} \leq G_e^{max} \right] \geq 1 - \epsilon_j \quad \forall j \in \mathcal{J} \tag{4.15}$$

where G_e^{min}, G_e^{max} represent the minimum and maximum output of the generator over the given time interval and accounts for ramping constraints and already made commitment decisions. Since $\mathbb{E}_\Omega [\mathbf{\Delta}] = 0$ and the standard deviation of $\mathbf{\Delta}$ is known, which takes the value σ_Δ, the following constraints to enforce Equation (4.15) are

$$x_j + \beta_j \sigma_\Delta \eta_g \leq G_j^{max} \quad \forall j \in \mathcal{J}$$
$$x_j - \beta_j \sigma_\Delta \eta_g \geq G_j^{min} \quad \forall j \in \mathcal{J}.$$

By allowing the slack distribution to be variables in the optimization procedure, the cost to meet load is now a random variable. Assuming we would like to minimize the expected cost of generation, we would like to minimize

$$\mathbb{E}_\Omega \left[\sum_{j \in \mathcal{J}} \left[c_2^j (x_j + \beta_j \mathbf{\Delta})^2 + c_1^j (x_j + \beta_j \mathbf{\Delta}) + c_0^j \right] \right]$$

Since $\mathbf{\Delta}$ is a Gaussian and $\mathbb{E}_\Omega [\mathbf{\Delta}] = 0$, we have that $\mathbb{E}_\Omega [\mathbf{\Delta}] = \sigma_\Delta^2$. Our new objective is now

$$\sum_{j \in \mathcal{J}} \left[c_2^j \left(x_j^2 + \sigma_\Delta^2 \beta_j^2 \right) + c_1^j x_j + c_0^j \right]$$

We defer derivation of the standard deviation of branch flow, σ_e, Equations (4.16h) and (4.16i)

in the following optimization problem, to the following section with a complete discussion on random power flows assuming a multivariate Gaussian distribution on net injects. The following optimization problems highlight the difference between OPF hard line thresholds for branch flow limits and generator constraints and the probabilistic versions in the CC version. The CC models make use of both mean and standard deviation for the random variables in question. The parameters ϵ_l and ϵ_g can be changed to reflect the preference or aversion of a constraint being violated. Of primary note is that with $\epsilon_l = \epsilon_g = .5$ the standard OPF model is recovered. As these epsilon parameters are reduced, constraints are tightened so that the optimal value of OPF Equation (4.4) is less than or equal to the optimal value of CC Equation (4.16).

$$\textbf{CC:=} \min_{(x,\beta,;\theta,y)} \sum_{j\in\mathcal{J}} \left[c_2\left(x_j^2 + \beta_j^2\sigma_\Delta^2\right) + c_1 x_j + c_0\right] \tag{}$$

$$\sum_{j\in\mathcal{J}} c_{ij}^g x_j - \sum_{j\in\mathcal{J}} c_{ie}^b y_e = d_i \qquad \forall i \in \mathcal{I} \tag{4.16b}$$

$$y_e - b_e \sum_{i\in\mathcal{I}} c_{ie}^b \theta_i = 0 \qquad \forall e \in \mathcal{E} \tag{4.16c}$$

$$\mu_e^y + \eta_l \sigma_e^y \leq U \qquad \forall e \in \mathcal{E} \tag{4.16d}$$

$$\mu_e^y - \eta_l \sigma_e^y \geq -U \qquad \forall e \in \mathcal{E} \tag{4.16e}$$

$$x_j + \beta_j \sigma_\Delta \eta_g \leq G_j^{max} \qquad \forall j \in \mathcal{J} \tag{4.16f}$$

$$x_j - \beta_j \sigma_\Delta \eta_g \geq G_j^{min} \qquad \forall j \in \mathcal{J} \tag{4.16g}$$

$$\pi_e - \sum_{j\in\mathcal{J}} A_{ej}\beta_j = 0 \qquad \forall e \in \mathcal{E} \tag{4.16h}$$

$$s_e^2 - \pi_e^2\sigma_\Delta^2 + 2\pi_e\sigma_{e_1}^2 \geq \sigma_{e_1 e_1}^2 \qquad \forall e \in \mathcal{E} \tag{4.16i}$$

$$\sum_{j\in\mathcal{J}} \beta_j = 1 \tag{4.16j}$$

4.4 Joint Chance Constraint for System Risk Measure

In order to ensure reliability of the bulk power system, the system risk needs to be quantified and constrained. In this chapter, we define system risk as the probability that no lines fail. We are primarily concerned with what we term endogeneous system risk, which is the likelihood of failure induced by the branch flows. We start with our system risk analysis under a fixed generation and demand scenario. We can solve this problem exactly with nonlinear programming. Following, we perform the analysis when the demand and generation are a multivariate Gaussian distribution in which the mean and covariance are known. In this system we make a linearization to approximate our system risk function.

We start with the situation in which generation and demand are known with certainty. Let $h(y) : \mathbb{R}_+^M \to [0, 1]$ be the risk function dependent on line flows y, let ϵ be the risk tolerance and define the system risk constraint as

$$h(y) = P_{\Xi}\left[\text{no line fails}|\text{line flows } y\right] \leq \epsilon \tag{4.17}$$

where the probability space Ξ can be thought of as an effective capacity. That is, the line fails if it has flow above its effective capacity. The space Ξ represents the transmission elements' effective capacity distribution. This risk measure Equation (4.17) defines a Bernoulli random variable that takes the value 1 with probability $h(y)$ if no lines fail, the complement of all lines succeeding. Assuming that the failure probabilities of individual lines are independent given the flow, we can find $h(y)$ by multiplying the probabilities that individual lines succeed $(1 - g_e(y_e))$, we have the probability that all lines succeed

$$h(y) = \prod_{e \in \mathcal{E}} \left(1 - g_e(y_e)\right) \tag{4.18}$$

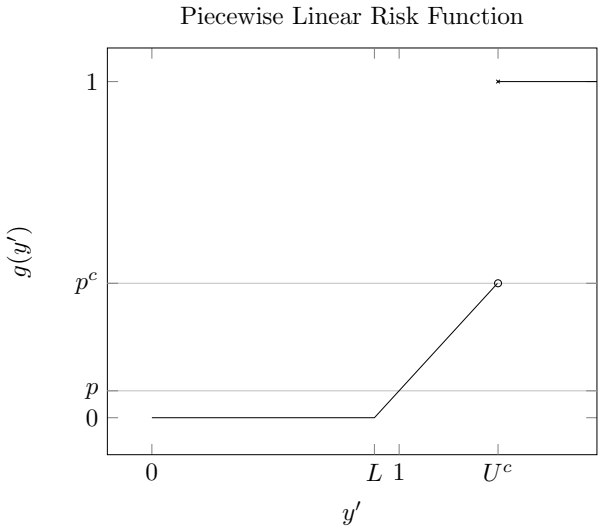


Figure 4.1: Line failure density function for normalized line flow y'

where $g_e(y_e)$ is the probability that line e fails given line flow y_e. The line risk function $g : \mathbb{R}^+ \to [0, 1]$ takes the line flow y_e and returns the probability $g_e(y_e)$ that the line will fail.

$$g_e(y_e) = \mathbb{P}_\Xi \left[\text{Line } e \text{ fails} | y_e\right] \qquad \forall e \in \mathcal{E} \tag{4.19}$$

We make note that the probability a line fails is only dependent on the flow y_e. While the flow y_e certainly covaries with the other line flows y, we assume the failure process is independent of other line flows.

A piecewise linear function captures the important features of the endogenous risk of the transmission element associated with loading. Up to a certain point, L_e, the power flow does not add any risk above that of its normal outage rate. Above that point, the additional risk is proportional to the additional flow on the line. Once a critical capacity U_e^c is reached, a protective

element is tripped and the line fails with certainty (ignoring hidden failures).

$$g_e(y_e) = \begin{cases} 0 & y_e \leq L_e \\ a_e + b_e y_e & L_e \leq y_e < U_e^c \\ 1 & U_e^c \leq y_e \end{cases} \tag{4.20}$$

Using as reference the probability p_e that the line fails at nominal capacity U_e, the piecewise linear paramters a_e and b_e can be calculated as $a_e = -p_e L_e (1 - L_e)^{-1}$ and $b_e = p_e (1 - L_e)^{-1}$. This piecewise linear failure density function is shown in Figure 4.1.

Lemma 4.4.1 *$h(y)$ is log-concave in domain $\left\{ y_e \in \mathbb{R}^{\mathcal{E}} | 0 \leq y_e < U_e^c \ \forall e \in \mathcal{E} \right\}$*

Proof First we note that for $\{y_e | y_e < U_e^c \forall e\}$, $g_e(y_e) \in [0, 1)$ and is the max of two convex functions and is thus convex. To show that $\ln h(y)$ is concave, we take the log of both sides of Equation (4.18), giving

$$\begin{aligned} \ln h(y) &= \ln \left[\prod_{e \in \mathcal{E}} (1 - g_e(y_e)) \right] \\ &= \sum_e \ln \left[(1 - g_e(y_e)) \right] \end{aligned}$$

For $x_e = g_e(y_e)$, let $H(x) = \ln h(x) = \sum_e \ln [1 - x_e]$, we have

$$\begin{aligned} \frac{\partial H(x)}{\partial x_e} &= -\frac{1}{1 - x_e} \\ \frac{\partial^2 H(x)}{\partial x_n^2} &= -\frac{1}{(1 - x_e)^2} \\ \frac{\partial^2 H(x)}{\partial x_n \partial x_e} &= 0 \end{aligned}$$

which shows that $H(x)$ is a decreasing and concave function on $\{x_e | x_e \in [0, 1) \, \forall e\}$, thus $-H(x)$ is

convex and non-decreasing. Composing a convex, non-decreasing function with a convex function is convex, so that $-\ln h(y)$ is convex and $\ln h(y)$ is concave. Thus, $h(y)$ is log concave. ∎

This function can be solved exactly for fixed demand using nonlinear programming by taking a log transform of the line risk. The system risk constraint Equation (4.18) can be written

$$\prod_{e\in\mathcal{E}}(1-g_e(y_e)) \geq 1-\epsilon$$

where $1-g_e(y_e)$ is the probability that line e doesn't fail given flow y_e. Taking the product of all these events gives the probability that no lines fail, we have

$$\ln\left(\prod_{e\in\mathcal{E}}(1-g_e(y_e))\right) \geq \ln(1-\epsilon) \qquad g_e(y_e)\in[0,1)\forall e\in\mathcal{E} \Leftrightarrow$$

$$\sum_{e\in\mathcal{E}}\ln(1-g_e(y_e)) \geq \ln(1-\epsilon) \qquad g_e(y_e)\in[0,1)\forall e\in\mathcal{E}.$$

If we enforce $w_e \leq \ln(1-g_e(y_e))\forall e\in\mathcal{E}$, then we have our no line failure system risk constraint Equation (4.18) as

$$\sum_{e\in\mathcal{E}} w_e \geq \ln(1-\epsilon) \tag{4.21}$$

To enforce $w_e \leq \ln(1-g_e(y_e))$ we manipulate to

$$w_e \leq \ln(1-g_e(y_e)) \qquad \forall e\in\mathcal{E}$$

$$\exp(w_e) \leq 1-g_e(y_e) \qquad \forall e\in\mathcal{E}$$

$$g_e(x_e)+\exp(w_e) \leq 1 \qquad \forall e\in\mathcal{E}.$$

Equation (4.21) implies that there are fixed line limits related to the risk tolerance ϵ. Since $\ln(1-x) \leq 0$ for all $x\in[0,1)$ shown in Figure 4.2, this implies that each term in the sum

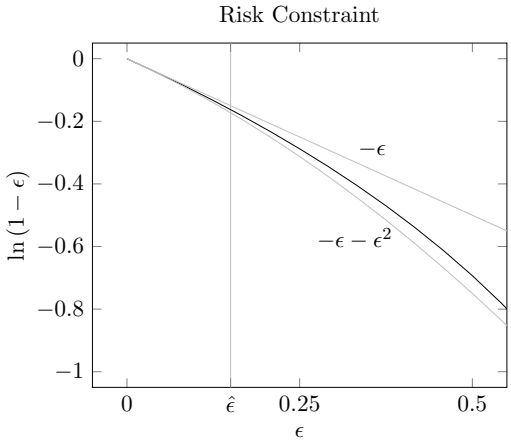


Figure 4.2: Line risk substitution for individual branches and possible approximations

also satisfies the constraint $\ln(1 - g_e(y_e)) \geq \ln(1 - \epsilon)$, giving the upper limit on flow y_e defined by

$$U_e^\epsilon = g_e^{-1}(y_e) \quad \forall e \in \mathcal{E} \tag{4.22}$$

4.4.1 System Risk for Multivariate Gaussian Branch Flows

Now we combine the uncertainty in the effective capacity Ξ with the uncertainty in generation and demand $\mathbb{D}$. We assume that the uncertainty spaces Ξ and $\mathbb{D}$ are orthogonal to each other. This may not be the case, due to geographically correlated weather which can affect both the effective capacity distribution and the random generation and demand. Applying the line risk function Equation (4.20) to the system risk measure Equation (4.18) and taking the expectation over the random injection space $\mathbb{D}$, we have

$$\mathbb{E}_\mathbb{D}[h(\boldsymbol{y})] = \mathbb{E}_\mathbb{D}[P_\Xi[\text{no lines fail}|\text{line flows } \boldsymbol{y}]] \tag{4.23}$$

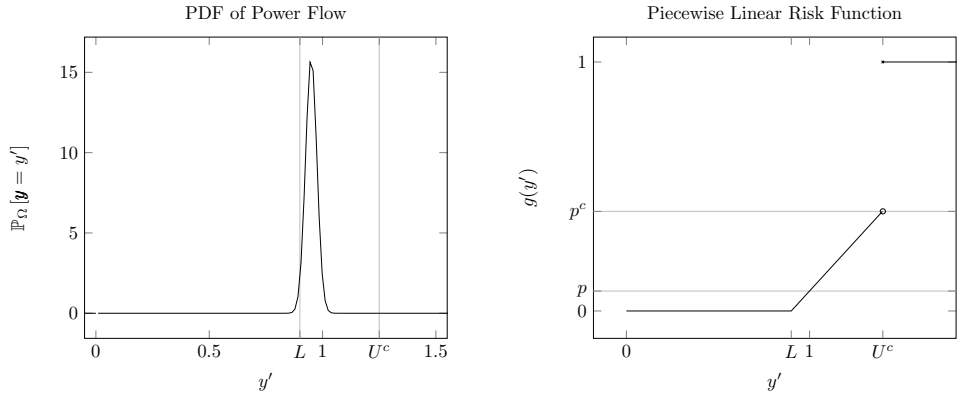
$$= \mathbb{E}_\mathbb{D}\left[\prod_{e\in\mathcal{E}}(1 - g_e(\boldsymbol{y}_e))\right] \tag{4.24}$$

since $1 - g_e(\boldsymbol{y}_e) = \mathbb{P}_{\Xi}\left[\text{Line } e \text{ succeeds}|\boldsymbol{y}_e\right]$ and multiplying over all lines to find the probability that no line fails. The function Equation (4.23) can be approximated by using a linearization of $h(\boldsymbol{y})$. Let r be the expectated value of the probability that one or more lines fail (the complement of $\mathbb{E}_{\mathbb{D}}\left[h(\boldsymbol{y})\right]$). Then, we have

$$
\begin{aligned}
r &= \mathbb{E}_{\mathbb{D}}\left[1 - h(\boldsymbol{y})\right] \\
&= 1 - \mathbb{E}_{\mathbb{D}}\left[\prod_{e\in\mathcal{E}}\left(1 - g_e(\boldsymbol{y}_e)\right)\right] \\
&= 1 - \mathbb{E}_{\mathbb{D}}\left[1 - \sum_{e} g_e(\boldsymbol{y}_e) + \sum_{e_1,e_2|e_1\neq e_2} g_{e_1}(\boldsymbol{y}_{e_1})g_{e_2}(\boldsymbol{y}_{e_2}) - \sum_{e_1,e_2,e_3|e_1\neq e_2\neq e_3} g_{e_1}(\boldsymbol{y}_{e_1})g_{e_2}(\boldsymbol{y}_{e_2})g_{e_3}(\boldsymbol{y}_{e_3}) + \cdots\right] \\
&\approx 1 - \mathbb{E}_{\mathbb{D}}\left[1 - \sum_{e} g_e(\boldsymbol{y}_e) + \frac{1}{2}\sum_{e_1,e_2|e_1\neq e_2} g_{e_1}(\boldsymbol{y}_{e_1})g_{e_2}(\boldsymbol{y}_{e_2}) - \cdots\right] \\
&\approx \sum_{e\in\mathcal{E}} \mathbb{E}_{\mathbb{D}}\left[g_e(\boldsymbol{y}_e)\right]
\end{aligned}
$$

where the approximation derived above comes from taking a Taylor expansions of $h(\cdot)$ at $y = 0$ and second order and higher terms. This approximation is good for small risk values of $h(y)$. The system risk contributed by line e is integrated over the space $\mathbb{D}$, which is orthogonal to effective capacity, or likelihood of failure due to loading, Ξ. This system risk measure r captures the endogenous system risk of line failures due to loading under uncertainty in branch flows.

Let $z_e = \mathbb{E}_{\mathbb{D}}\left[g_e(\boldsymbol{y}_e)\right]$ be the individual line risk. We can break z_e into three segments based upon the value of $\boldsymbol{y}_e$ in piecewise linear risk function Equation (4.20). The first segment for $\boldsymbol{y}_e \leq L_e$ has $g_e(\boldsymbol{y}_e) = 0$. In the second segment, for $L_e \leq \boldsymbol{y}_e \leq U_e^c$, $g_e(\boldsymbol{y}_e)$takes an expected value of a truncated normal distribution. In the last segment, $U_e^c \leq \boldsymbol{y}_e$, $g_e(\boldsymbol{y}_e)$ takes one and $\boldsymbol{y}_e$ is in that segment with the probability calculated from the cumulative distribution function (CDF) of



(a) Probability density of power flow (b) Line failure density function

Figure 4.3: Random Power Flows and the Failure Density Function

a normal distribution. That is,

$$z_e = 0 + \mathbb{E}_{\mathbb{D}}\left[a + b\boldsymbol{y}_e | L \leq \boldsymbol{y}_e \leq U^c\right] \mathbb{P}_{\mathbb{D}}\left[L \leq \boldsymbol{y}_e \leq U^c\right] + 1 * \mathbb{P}_{\mathbb{D}}\left[U^c \leq \boldsymbol{y}_e\right]$$

The truncation from the critical capacity U^c, the level at which the line fails with certainty, is approximately 0 for choices of ϵ around a few percent. If the mean flow $\boldsymbol{y}_e$ is at the hard limit U^ϵ defined by risk tolarence ϵ, the following point $(U^c - U^\epsilon)/\sigma$ is the corresponding point in a standard normal distribution and $\Phi\left((U^c - U^\epsilon)/\sigma\right)$ is the probability of being above the critical capacity U^c. When solving this problem, the assumption that $\mathbb{P}_{\mathbb{D}}\left[U^c \leq \boldsymbol{y}_e\right] \approx 0$ can be checked numerically. Then, we have

$$z_e \approx \mathbb{E}_{\mathbb{D}}\left[a + b\boldsymbol{y}_e | L \leq \boldsymbol{y}_e\right] \mathbb{P}_{\mathbb{D}}\left[L \leq \boldsymbol{y}_e\right]$$

The branch flows are truncated Gaussian, which have known mean and variances. Truncated Gaussian's expectation and tail probability are

$$\mathbb{E}_{\mathbb{D}}\left[\boldsymbol{y}_e | L \leq \boldsymbol{y}_e\right] = \mu_e^y + \frac{\phi(\alpha_L)}{1 - \Phi(\alpha_L)}\sigma_e^y$$

$$\mathbb{P}_{\mathbb{D}}\left[L \leq \boldsymbol{y}_e\right] = 1 - \Phi(\alpha_L)$$

where $\alpha_L = \frac{L-\mu_e^y}{\sigma_e^y}$, the PDF $\phi(\cdot)$, and the CDF $\Phi(\cdot)$ are for the standard normal distribution. The line risk $\rho(\mu_e^y, \sigma_e^y)$ is a function of the mean μ_e^y and standard deviation σ_e^y of the branch flows, given as

$$\rho(\mu_e^y, \sigma_e^y) = \mathbb{E}_{\mathbb{D}}\left[a + b\boldsymbol{y}_e | L \leq \boldsymbol{y}_e\right] \mathbb{P}_{\mathbb{D}}\left[L \leq \boldsymbol{y}_e\right] \tag{4.25}$$

$$= (a + b\mu_e^y)\left[1 - \Phi(\alpha_L)\right] + b\sigma_e^y \phi(\alpha_L). \tag{4.26}$$

Given the approximation to z_e of the truncation of the tail of the distribution, we have

$$z_e \approx \rho(\mu_e^y, \sigma_e^y) \tag{4.27}$$

4.4.2 Joint Chance Constraint Model

The JCC model departs from traditional power flow models in that it allows for a trade-off between line risk, system risk, and cost. Since the cost and risk are both related to the slack distribution, the model is allowed to vary the slack distribution. In this section, the full joint chance constraint model follows with a brief explanation of some of the constraints. The relevant parameters derived from the injection covariance matrix is also given. Then a cutting plane algorithm is shown to solve the full JCC model.

This convex program is minimizing a quadratic cost function of generation x_j for each generator $j \in \mathcal{J}$ as well as the expected cost from its participation in the slack distribution β_j. The cost coefficients for generator j are c_j^2, c_j^1, and c_j^0. The incidence matrix c_{ij}^g is 1 if generator j is connected to bus i. The incidence matrix c_{ie}^b is 1 if branch e is from bus i and -1 if branch e is to bus

i. The parameter b_e is the susceptance of branch e and U_e^ϵ is the hard limit on branch flow defined by the given risk level in Equation (4.22). Each generator j has a risk level ϵ_j and an associated η_j from Equation (4.14). The standard deviation of aggregate generation and demand uncertain is σ_Δ^2. The variable π_e represents how the slack distribution affects branch e in repsonse to aggregate changes in generation and demand. The variable s_e represents the standard deviation of branch flow on branch e and z_e represents the probability that line will fail. The sum of all branch failure probabilities is constrained to be less than the given risk level ϵ.

$$\textbf{JCC:=} \min_{(x,\beta;\theta,y,\pi,s,z)} \sum_{j\in\mathcal{J}} \left[c_j^2\left(x_j^2+\beta_j^2\sigma_\Delta^2\right)+c_j^1 x_j + c_j^0\right] \tag{4.28a}$$

$$\sum_{j\in\mathcal{J}} c_{ij}^g x_j - \sum_{j\in\mathcal{J}} c_{ie}^b y_e = d_i \qquad \forall i\in\mathcal{I} \tag{4.28b}$$

$$y_e - b_e \sum_{i\in\mathcal{I}} c_{ie}^b \theta_i = 0 \qquad \forall e\in\mathcal{E} \tag{4.28c}$$

$$y_e \in [-U_e^\epsilon, U_e^\epsilon] \forall e\in\mathcal{E} \tag{4.28d}$$

$$x_j + \beta_j\sigma_\Delta\eta_j \le G_j^{max} \qquad \forall j\in\mathcal{J} \tag{4.28e}$$

$$x_j - \beta_j\sigma_\Delta\eta_j \ge G_j^{min} \qquad \forall j\in\mathcal{J} \tag{4.28f}$$

$$\sum_{j\in\mathcal{J}} \beta_j = 1 \tag{4.28g}$$

$$\pi_e - \sum_{j\in\mathcal{J}} A_{ej}\beta_j = 0 \qquad \forall e\in\mathcal{E} \tag{4.28h}$$

$$s_e^2 - \pi_e^2\sigma_\Delta^2 + 2\pi_e\sigma_{e_1}^2 \ge \sigma_{e_1e_1}^2 \qquad \forall e\in\mathcal{E} \tag{4.28i}$$

$$z_e - g_e(|y_e|, s_e) \ge 0 \qquad \forall e\in\mathcal{E} \tag{4.28j}$$

$$\sum_e z_e \le \epsilon \tag{4.28k}$$

The objective Equation (4.28a) for the JCC model is the typical quadratic objective for the

OPF model plus a contribution from the slack distribution due to the uncertainty in the random injects. This objective is the expected cost of meeting the realized demand. We use individual chance constraints to deal with the uncertainty in load for Equations (4.28e) and (4.28f) with $\eta_g = \Phi^{-1}(1-\epsilon_g)$ being its tolerance. The upper and lower bounds on generator levels $[G^{min}, G^{max}]$ are dependent on many things, most importantly the time frame used. The levels are affected by the status of the generator (on,off,starting up,shutting down), its ramping rate, and any other physical limits it may have.

The first equations Equations (4.28a) to (4.28d) and (4.28g) are your typical DC power flow, Equations (4.28e) and (4.28f) are chance constraints on generators, and the last set Equations (4.28h) to (4.28k) describe system risk. The branch variance Equation (4.28i) are a second order cone and the line risk Equation (4.28j) involve the CDF of a normal distribution. These equations are solved via a cutting plane approach so that the individual subproblems are linear programs.

4.4.3 Solution Methodology

Since the risk function is convex, we choose cutting planes to approximate the line risk constraints Equation (4.28j). In addition, there were many second order cone constraints (number of lines). Instead of using these constraints explicitly, they were added through cutting planes as well. This kept the program to a manageable size and allowed for fast solve times.

Lemma 4.4.2 *The line risk function Equation* (4.25) *is convex with respect to* μ_e^y *and* σ_e^y

Proof Starting with the line risk function Equation (4.25)

$$\rho_e(\mu_e^y, \sigma_e^y) = (a_e + b\mu_e^y)\left[1 - \Phi(\alpha_L)\right] + b\sigma_e^y\phi(\alpha_L)$$

First, we calculate $\frac{\partial \Phi(\alpha_L)}{\partial \mu_e^y} = -\frac{1}{\sigma_e^y}\phi(\alpha_L)$ and $\frac{\partial \phi(\alpha_L)}{\partial \mu_e^y} = \frac{1}{\sigma_e^y}\alpha_L\phi(\alpha_L)$ using the chain rule. Now note that from our piecewise linear function Equation (4.20), we have the identity $L = -\frac{a_e}{b_e}$. Then, taking the derivative of ρ with respect to μ_e^y, we get

$$\begin{aligned}
\frac{\partial \rho_e}{\partial \mu}(\mu_e^y, \sigma_e^y) &= (a_e + b_e\mu_e^y)\frac{1}{\sigma_e^y}\phi(\alpha_L) + b_e\left[1 - \Phi(\alpha_L)\right] + b_e\alpha_L\phi(\alpha_L) \\
&= \frac{a_e + b_e\mu_e^y}{\sigma_e^y}\phi(\alpha_L) + b_e\left[1 - \Phi(\alpha_L)\right] + \frac{b_eL_e - b_e\mu_e^y}{\sigma_e^y}\phi(\alpha_L) \\
&= \frac{a_e + b_e\mu_e^y}{\sigma_e^y}\phi(\alpha_L) + b_e\left[1 - \Phi(\alpha_L)\right] - \frac{a_e + b_e\mu_e^y}{\sigma_e^y}\phi(\alpha_L) \\
&= b_e\left[1 - \Phi(\alpha_L)\right]
\end{aligned}$$

For our derivative with respect to σ_e^y, we calculate $\frac{\partial \Phi(\alpha_L)}{\partial \sigma_e^y} = -\frac{1}{\sigma_e^y}\alpha_L\phi(\alpha_L)$ and $\frac{\partial \phi(\alpha_L)}{\partial \sigma_e^y} = \frac{1}{\sigma_e^y}\alpha_L^2\phi(\alpha_L)$. Then, taking the derivative of ρ with respect to σ_e^y, we have

$$\begin{aligned}
\frac{\partial \rho_e}{\partial \sigma}(\mu_e^y, \sigma_e^y) &= (a_e + b_e\mu_e^y)\frac{1}{\sigma_e^y}\alpha_L\phi(\alpha_L) + b_e\sigma_e^y\frac{1}{\sigma_e^y}\alpha_L^2\phi(\alpha_L) + b_e\phi(\alpha_L) \\
&= -b_e\frac{L - \mu_e^y}{\sigma_e^y}\alpha_L\phi(\alpha_L) + b_e\alpha_L^2\phi(\alpha_L) + b_e\phi(\alpha_L) \\
&= b_e\phi(\alpha_L)
\end{aligned}$$

where the identities $a_e = -b_eL_e$ and $\alpha_L = \frac{L_e - \mu_e^y}{\sigma_e^y}$ were used.

The Hessian is found with the second derivatives and is given by

$$\bigtriangledown^2 \rho_e(\mu_e^y, \sigma_e^y) = \frac{b_e\phi(\alpha_L)}{\sigma}\begin{bmatrix} 1 & \alpha_L \\ \alpha_L & \alpha_L^2 \end{bmatrix} \tag{4.29}$$

Then, we note that the determinant is 0 and the diagonal elements are positive so that there is a positive eigenvalue and a zero eigenvalue, thus the line risk is convex with respect to μ_e^y, σ_e^y. Since system risk is approximated by the sum of line risks, this system risk measure is convex. ∎

Since the inequality Equation (4.28j) is convex, we can solve this using a cutting plane approach to describe the line risk in terms of the mean branch flow y_e and the standard deviation of branch flow s_e. While there are infinitely many cuts, this program can be solved for a given error tolerance with a finite, and typically small, set of cuts.

$$z_e \geq \rho_e(\hat{y}_e, \hat{s}_e) + \frac{\partial \rho}{\partial y_e}(\hat{y}_e, \hat{s}_e)\,(y_e - \hat{y}_e) + \frac{\partial \rho}{\partial s_e}(\hat{y}_e, \hat{s}_e)\,(s_e - \hat{s}_e) \tag{4.30a}$$

$$\rho_e(y_e, s_e) = (a_e + b_e y_e)\,[1 - \Phi(\alpha_L)] + b_e s_e \phi(\alpha_L) \tag{4.30b}$$

$$\frac{\partial}{\partial y_e}\rho_e(y_e, s_e) = b_e\,[1 - \Phi(\alpha_L)] \tag{4.30c}$$

$$\frac{\partial}{\partial s_e}\rho_e(y_e, s_e) = b_e \phi(\alpha_L) \tag{4.30d}$$

The standard deviation of branch flow can be formulated as a second order cone with respect to the slack distribution variables. While this could be solved with a commercial solver, we proceed with a cutting plane approach to speed up solve times. As the solution approach is already iteratively improving system risk, it adds only a small amount of work to add these cuts as well and reduces the subproblem to a linear program. The inequallities required to approximate the nonlinear constraints Equation (4.28i) are:

$$s_e \geq f_e\left(\hat{\beta}\right) + \sum_{j \in \mathcal{J}} \frac{\partial f_e\left(\hat{\beta}\right)}{\partial \beta_j}\left(\beta_j - \hat{\beta}_j\right) \tag{4.31a}$$

$$f_e\left(\beta\right) = \sqrt{\pi_e^2 \sigma_\Delta^2 - 2\pi_e \sigma_{e_1}^2 + \sigma_{e_1 e_1}^2} \tag{4.31b}$$

$$\frac{\partial}{\partial \beta_j} f_e\left(\beta\right) = \frac{A_{ej}\left(\pi_e \sigma_\Delta^2 - \sigma_{e_1}^2\right)}{\sqrt{\pi_e^2 \sigma_\Delta^2 - 2\pi_e \sigma_{e_1}^2 + \sigma_{e_1 e_1}^2}} \tag{4.31c}$$

Cutting Plane Algorithm

Now, we can describe the cutting plane algorithm for JCC at a high level. Algorithm 3 is pseudo code for the implementation. The main subproblem the algorithm solves is the standard DC power flow, which is defined by Equations (4.28a) to (4.28d) and (4.28g) . After solving, the generator injects x, slack distribution β, and branch flows y are used to calculate risk information about the dispatch point. To get the risk information, the branch standard deviations s need to be calculated. With the mean flow y and standard deviation s, the line risk z can be calculated. The sum of line risk is the system risk r. If r is less than the required system risk ϵ, the problem is solved. Otherwise, the algorithm adds cuts for all lines with a positive risk z. The cuts describe how risk z is related to branch flows y,standard deviation s and how s is related to β. Then the power flow subproblem is solved with the addition of the cuts and this repeats until it is infeasible or the risk constraint is satisfied.

Algorithm 3 This cutting plane algorithm solves JCC Equation (4.28) via linear programs and cutting planes

procedure JCC(d,Σ^m,ϵ,ϵ_g,L,p)
  $L \leftarrow \emptyset$ (Set of Lines with potential risk)
  $S \leftarrow \emptyset$ (Set of Cuts)
  $r \leftarrow 0$ (Risk)
solve:
  $(\hat{x}, \hat{\beta}, \hat{y}) \leftarrow$Solve DC Power Flow, Equations (4.28a) to (4.28g), with cuts S, risk $r \leq \epsilon$
  if Infeasible **then return** Problem Infeasible
  Calculate $\hat{s}, \hat{z}, \hat{r}$ using $(\hat{x}, \hat{\beta}, \hat{y})$ and Equation (4.9) and **??**
  if $\hat{r} \leq \epsilon + tol$ **then return** Optimal $(\hat{x}, \hat{\beta}, \hat{y}, \hat{s}, \hat{z}, \hat{r})$
  for $\forall e$ **do**
    if $\hat{z_e} \geq tol$ **then**
      if $e \notin L$ **then**
        $L \leftarrow \{L, e\}$
        Initialize s_e, z_e
        $r \leftarrow r + z_e$
      $S \leftarrow$ line risk cuts Equation (4.30) for z_e, y_e, s_e dependent on $\hat{z}_e, \hat{y}_e, \hat{s}_e$
      $S \leftarrow$ branch variance cuts Equation (4.31) for s_e, β_e dependent on $\hat{s}_e, \hat{\beta}_e$
  goto *solve*

4.5 Computational Experiments

This section explains the computation setup, the test cases, the experiments performed, and the intuition to gain from the comparison of OPF, CC, and JCC models.

4.5.1 Implementation of the JCC Model

All of the experiments are run on a laptop with an Intel i7-3537U processor with 2 cores,4 threads at 2.00GHz. The laptop has 8GB of memory, but even with the large instances (2383 buses), memory is not an issue. The laptop is running Linux Mint Petra 16. The OPF, CC, and JCC models are all solved within a C++ program. The program uses Armadillo[41] for linear algebra computation and Concert and CPLEX to solve the linear programs. CPLEX is run with default settings and the dual simplex algorithm is used to solve the linear programs (quadratic objective for the 30 bus test case). All time comparisons are using the same system environment with a fixed clocked speed and few programs in background. The primary DC OPF solver in the C++ program has been developed to output nearly identical results to Matpower[56].

4.5.2 Single Instance

The test cases are all taken from Matpower test cases. The two test cases used are the 30 bus test case as well as the 2383wp test case. The small test case is used to show properties of the different dispatch points and the cost-risk frontier. The large test case is used for time trials as well as a cost-risk scatter plot.

	OPF	CC	JCC
Cost	567.1	574.0	565.2
r	0.0171	0.0179	0.0080

Table 4.1: Cost and risk results for OPF,CC, and JCC models on the small test case

		Generator					
	Mod	1	2	3	4	5	6
$[g_{min}, g_{max}]$ $\{c_2, c_1\}$		[0,80] $\{0.02, 2\}$	[0,80] $\{0.0175, 1.75\}$	[0,50] $\{0.0625, 1\}$	[0,55] $\{0.00834, 3.25\}$	[0,30] $\{0.025, 3\}$	[0,40] $\{0.025, 3\}$
x_g	OPF	39.7	52.3	24.2	35.7	19	18.3
x_g	CC	35.9	47.9	25.7	37.2	19.3	23.1
x_g	JCC	41.9	55.0	23.2	34.0	18.6	16.5
β_g	OPF	0.1548	0.1769	0.0495	0.3712	0.1238	0.1238
β_g	CC	0	0	0.4411	0.2986	0	0.2602
β_g	JCC	0.2456	0.2795	0.0846	0.0646	0.0597	0.2659

Table 4.2: Generator results using OPF, CC, and JCC models on the small test case.

30 Bus Case

This case from Matpower has 30 buses, 41 branches, and 6 generators. The branches have a single capacity rating and in the given demand scenario, none of the line capacity constraints are active. A capacity factor M is used to uniformly scale the branch capacities. All of the generators are active and have quadratic cost functions. Ramping constraints for generators are not considered and the generators are allowed to take any value in its given range. In addition, all generators are allowed to participate in the slack distribution. Each bus with a demand is considered random, with mean equal to the demand. For this example, the injections are independent of each other but this need not be the case. The variance of each injection is equal to 5% of the demand at the node times a budget factor B used to tune the model uncertainty level.

The first example has risk parameters $L = 0.9$, that is a line begins taking on risk after it is at 90% of its rated capacity. In the risk function, we set the parameter $p_e = 0.005$ for all branches e, which means that if flow is at nominal capacity, the line has a 0.5% chance of failing. The line capacities are scaled by $M = 0.745$. The system risk constraint is $\epsilon = 0.008$, that is we

would like to enforce that the risk of one or more lines failing is less than or equal to 0.8%. The chance constrained version of OPF is defined by $\eta_L = 0.05$, that is the line capacities can not be violated more than 5% of the time. Finally, the variance of the injects is scaled by $B = 0.025$ so that the standard deviation of aggregate demand is 0.6, a small fraction of the total demand, 189.2, of this instance. These values were chosen to get the objective values close and highlight how the uncertainty in generation and demand affect the models differently. The solution values for generation and slack distribution (x, b) are tabulated in Tables 4.1 and 4.2. Increasing B to 0.25 by 10 fold led to the standard deviation of aggregate demand to be 1.345, around 1% of total load. The cost of OPF and JCC had negligible change, whereas CC cost increased to \$597, an increase of \$30 or an increase of 5% over the OPF solution. When the OPF model has transmission congestion, the cost of the CC model is highly sensitive to the uncertainty in demand and often infeasible solutions.

The CC model is always more conservative than the OPF model as it tightens the line constraints. This means that the CC version is always at least as expensive as the OPF model. In this case, the total cost rose by \$7 (to \$574, or a 1.2 % increase) to ensure the line constraints were met probabilistically (95% of the time). The JCC model was able to lower the cost because it removed the branch capacity constraints (or increased them by 6% to match the system risk level). In addition, JCC knows and constrains the system risk measure so that it is able to find a cheaper point with a system risk level half that of the OPF and CC models.

Another important note is that the slack would like to be distributed as much as possible to reduce cost. This can be seen from the objective Equation (4.28a) due to the squared beta term and the OPF results show it spreading when there is no chance constraints for lines or system risk constraints. The CC model slightly changes its generator position as well as removing 3 generators from the slack distribution. It is removing generators which has an effect on the probabilistically

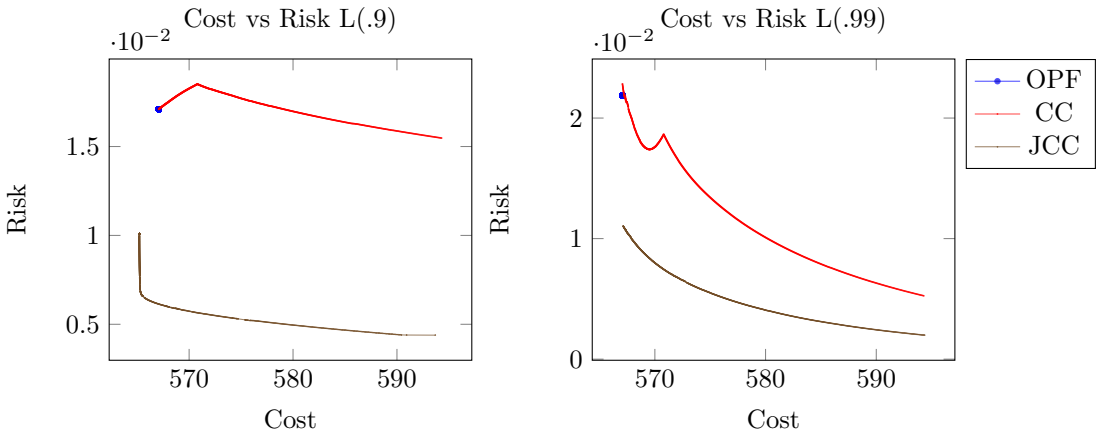


Figure 4.4: Reliability frontier for the small test case

constrained transmission lines in order to ensure that the constraints are met 95% of the time. The JCC model has moved in almost exactly the opposite direction and has kept a distributed slack (as seen in Table 4.2).

Cost-Risk Frontier

Now we use this same example case to draw a cost-risk frontier. An efficient operating point would be at that boundary of the feasible points, that is, neither cost nor risk can be improved without a loss to the other. Since neither the OPF or CC model constrains our risk measure, it would be unlikely for them to be on the boundary. In Figure 4.4 we see that this is the case. The OPF model is in the interior and is equivalent to the CC model when $\eta_l = .5$. Since the branch flows are Gaussian, if the mean flow is at its threshold, it has a 50% chance of being over its threshold. As η_l is reduced, the CC line is drawn and is stopped once the system becomes infeasible. The JCC line is started by finding the point when the system risk level is not constrained, for this case $r = .01$. As r is decreased, the system risk becomes constrained and the cost begins to rise as the risk level is reduced and the line stops when the system becomes infeasible. We used parameters $M = 0.775$ for line capacity scaling and $B = 0.4$ with standard deviation of aggregate load being around 1% of

total demand. The left plot uses $L = .9$ for calculating the system risk measure whereas the right plot uses $L = .99$ to calculate the system risk measure. In practice, it was found that as L gets closer to 1, the model behaves more like the CC model.

It is important to note that the cost-risk frontier is entirely dependent on how system risk is measured. In our case, we use a piecewise linear failure density function with parameters L and p. As L and p are varied, the shape of these frontiers changes. Holding p fixed and increasing L towards 1, the CC model, while still on the interior of the frontier, does a better job of reducing system risk. With L around .98, the CC and the JCC behave similarly. Both programs try to reduce the flow of lines that are at their capacity, with the exception that JCC allows a small number (typically one), of flow on these lines to increase. More discussion on this behavior is given in the sensitivity analysis Section 4.5.3.

2383wp Bus Case

This case from Matpower has 2383 buses, 2896 branches, and 327 generators. The case is similar to the small case in many respects, such as only a single line rating capacity being given. The generators have the biggest difference in that there are many which are nearly fixed and have no cost. The larger flexible generators are only given a linear cost so that there is no quadratic objective. The covariance matrix is developed the same as in the small case, i.e. with no covariance in random injects and the variance depending on the total demand of each node

Similar to the 30 bus case, we created a random instance using parameters $M = 1.03, \epsilon = 0.03, \eta_L = 0.05, L = 0.85, p = 0.005$, and $B = 1$ and recording the standard cost and risk information as well as solve time. The case was repeated 10 times and the mean time is reported in Table 4.3. CC and JCC take a similar amount of time, the majority of the work at each step being the

Time (η_L, ϵ)	OPF	CC	JCC
Time (0.05,0.03)	0.37	8.7	7.5
Time (0.20,0.07)	0.37	8.4	8.0
Time (0.40,0.20)	0.37	8.3	6.1
Time (0.48,0.30)	0.37	7.7	6.1
Avg Time (100 trials)	0.34	2.2	2.3

Table 4.3: Time comparison, in seconds, for OPF,CC, and JCC on the large test instances.

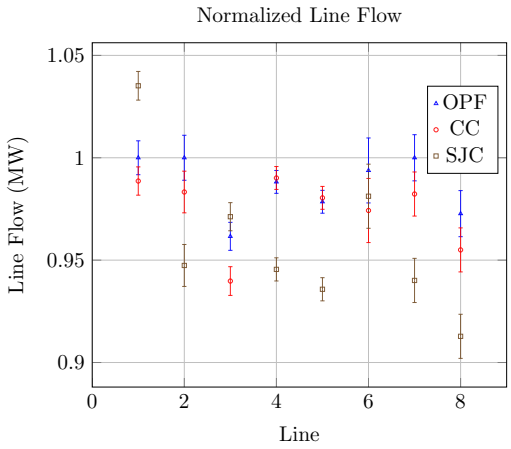
calculation of the branch covariance matrix. The total time is largely dependent on how many iterations the algorithms take, which is typically around 5 or 7 until convergence.

In addition, we solved 100 instances with random demand and the same covariance matrix to show the speed-up you can achieve with repeated solves. By having the same covariance matrix, all of the cuts for previous solves are still valid. So, in addition to using a warm start on the LP, after the first few solves the algorithm typically needs no additional cuts. In practice, this could mean that the important lines and scenarios from the risk perspective are known before hand and cuts are added based on the assumed covariance matrix. Instead of taking up to 5 to 7 iterations, it may solve in only one and perform a simple check to ensure that any lines not included in the analysis are not violated. The last row in Table 4.3 shows that the solve time was reduced from on around 8 seconds to around 2 seconds.

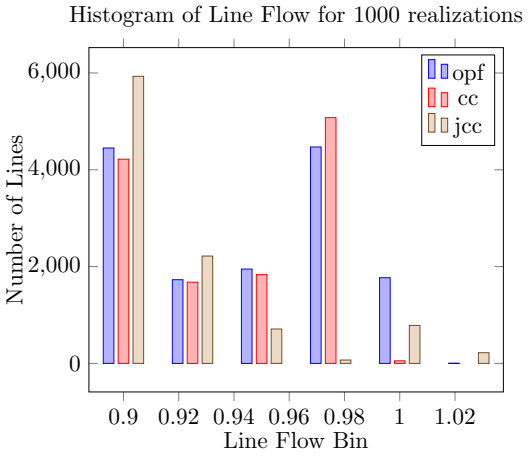
Line Threshold Comparison

Here we explore what the JCC model is doing to lower its cost or risk compared to the traditional models. For this experiment, the parameters are as follows, $M = 1.03, \epsilon = 0.015, \eta_L = 0.05, L = 0.95, p = 0.005$, and $B = 1$. Solving this program, we see that the majority of the lines are less than 50% utilized. There are a small number of lines that are at or near their nominal capacity.

Now, we look at the trade-off that the JCC model is able to make due to it increasing the nominal line limit by 10% to match the given system risk level. In Figure 4.5a, 8 lines are shown,



(a) All lines that have mean flow (in OPF) above 95% of nominal capacity



(b) Histogram to compare high flow lines of OPF,CC, and JCC

which are the lines above 95% of their nominal capacity in the OPF model. The mean line flows are plotted as well as the standard deviation. There are 8 lines above 95% of their capacity in the OPF model, indexed as lines 23,291,320,1380,1381,1815,2108, and 2109. In the OPF model, there are 3 lines at the nominal capacity. Line 23 (index 1 in figure) has a dual price of 872, line 291 (index 2) has a dual price of 32, and line 2108 (index 8) has a dual price of 294. Ideally, line 23 constraint should be relaxed due to the higher shadow price and perhaps other lines further constrained so that the system risk level is not increased. However, in the CC model, these constraints are tightened further to ensure that these thresholds are met at least 95% of the time. The JCC model on the other hand relaxes these constraints and imposes a system risk constraint instead. The final solution for the JCC model has line 23 exceed its threshold with certainty but is rewarded with savings related to the high shadow price. On the other hand, other lines with positive line risk are be restricted to ensure the system risk levels are met.

To give another perspective, the line flows were sampled and binned to create a histogram of line flows to show the trade-off between models. For the JCC model, the line risk parameter L was chosen to be 0.9. The results can be seen in Figure 4.5b. We can see that the normalized line

	OPF	CC .01	JCC .98
Cost	565.2	590.2	589.6
r	0.00504	0.00150	0.00031
Ex	0.3875	0.0225	0.0125

Table 4.4: Risk comparison for OPF,CC, and JCC on the small test instance.

flows pile up in bin 0.975 for the CC model, and there are a number of lines in OPF that are in bin $1 - 1.025$, thus exceeding their capacity a significant amount of the time. The JCC also has a line that exceeds its capacity, almost with certainty. However, the JCC has far fewer lines that are at or near their capacity.

The primary strength of the CC model is to meet the line threshold constraints probabilistically when the demands are not known with certainty. Let's look at how well it does from the system perspective, that is the expected number of lines to be over their nominal capacity, shown in Table 4.4. First we note the costs, both CC and JCC are around 5% higher and in return for the higher cost, have better risk characteristics. In the standard OPF model, the expected number of lines over their threshold is 20 times that of the CC model. The JCC halves the CC model for roughly the same cost. The CC model directly constraints these probabilities, but does so on the individual line level. By having a system constraint, the JCC model is better suited to optimize risk characteristics related to system level transmission utilization.

4.5.3 Sensitivity Analysis

Now, we want to look at the sensitivity of the JCC model to its input parameters and demand uncertainty. First we look at how the risk of the different dispatch points from the OPF, CC, and JCC models respond to changes in the failure density function parameters L and p. To this end, we solved the OPF, CC ($\eta_L = .01$), and JCC ($\epsilon = 0.0003$, $L = .98$). These three dispatch points are used to find the risk of the branch flows for varying risk parameters L, p. The cost of the dispatch

	OPF	CC	JCC
Cost	567.1	574.0	572.2

Table 4.5: Table of cost for the three dispatch points used in the sensitivity analysis

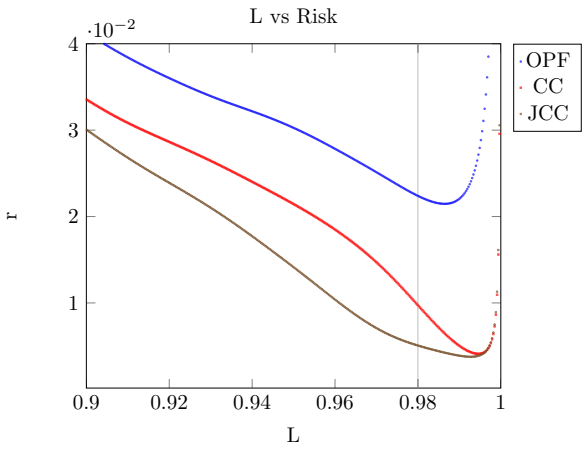


Figure 4.6: A comparison of the three dispatch points created by fixing p and varying L

points are shown in Table 4.5 and the sensitivity analysis is shown in Figure 4.6. This figure shows that for the given JCC solution, the system risk stays under that of the CC model. The OPF model has a lower cost so it is reasonable that the risk is much higher. Even when changing to other polynomial risk functions, such as x^2, x^3, and x^4, the JCC model performed better in terms of this alternative risk function than the OPF and CC models. These parameters are very difficult to estimate in the real world and as such it is very important that our model has shown to be robust to changing both the parameters in the piecewise linear model as well as testing against different polynomial models for failure density functions. This sensitivity analysis shows that while our parameters may be off, this model still does well for the assumptions that at some point, lines begin to take on additional risk due to congestion and that risk is monotonically increasing.

4.6 Conclusion

The primary strength of the JCC model is the addition of a system risk constraint and the relaxation of line thresholds. Line thresholds are hard constraints that subject the system to price spikes. The hard line constraints are somewhat arbitrary as the line does not fail when it is exceeded by a small amount. Instead, an economic trade-off should be made between individual lines to ensure the system risk is constrained at an adequate level. The system risk measure allows for direct comparison of different dispatch points with respect to risk.

The JCC model is computationally efficient in its full form. It allows for creation of a cost-risk frontier that finds dispatch points that are better in both terms of risk and cost. Under uncertainty, this model should be compared against the CC model, which is a probabilistic interpretation of line thresholds. In the computational section, we saw the extremely high sensitivity of cost to uncertainty in load when the transmission system is congested. In both of these models, the variable slack distribution plays a small direct role in cost through the objective, however also plays a role in ensuring the line and system risk constraints are met (and a larger indirect cost contribution). These system risk constraints capture the risk of a heavily loaded transmission system that the traditional OPF and CC miss that is related to cascading power failure risk in literature.

Chapter 5

Reducing Cascading Risk Through Real-Time Dispatch

Large scale load shedding events caused by cascading power failures have an extreme impact on society. As seen in Chapter 1, the costs of these single events can be in the billions, put a halt on commercial and industrial activities, and even cause the loss of life. In this Chapter, we extend our joint chance-constrained model to more effectively control line failures that may lead to large load-shedding events.

The downside of a chance constrained model is that it does not account for the impact of failure events. Power system networks are comprised of many different classes of assets. The most straightforward example is that the failure of a small distribution line in a radial tree has minimal impact on the reliability of the bulk power system and the high voltage network that carries the majority of the power over long distances. However, even among the bulk power system, the failure of individual lines may have disproportionate impact on the probability of a large scale cascading event. This may have to do with many things such as the connectivity of a neighboring node, the

current environmental conditions in and around a region, or even the relay settings on neighboring transmission lines. Here we take an empirical approach using the OPA cascading simulation from Chapter 3 to evaluate the impact of losing a line on the resulting cascade simulation and the output of the load shed distribution. We then incorporate this measure of impact into our JCC model by giving it an OPA weighting scheme, denoted JCC-OW (Joint Chance Constraint - OPA Weighted). This OPA weighting scheme can be thought of as a surrogate model for OPA to be used in a real time dispatch model. This problem is solved with a cutting plane algorithm similar to JCC. Finally, we explore the trade-off between cost of dispatch and rare event risk through the computational section.

5.1 Cascading Failure Risk Model

We would like to take into account the impact of losing a transmission element on the load shed distribution from OPA. Instead of being concerned with only whether or not a line fails, we would also like to know how much does this line contribute to cascading power failure risk. Due to reliability requirements such as $N-1$ constraints, the base system is typically stable and has low cascading power failure risk. Since line failures can be caused by things outside of system control, we perform our risk analysis under $N-1$ exogenous contingencies corresponding to each individual line failure. After these initial line failures the system is sometimes moved to a state where additional failures may begin a cascading process leading to a large load shed event. We use these $N-1$ contingencies as well as our JCC risk model to sample initial contingencies for the OPA cascading process. The initial contingency includes the line that failed with certainty due to an exogenous event as well as probable line failures due to current flow on transmission elements. We use the distribution of load shed after the OPA process to develop a weighting for the JCC

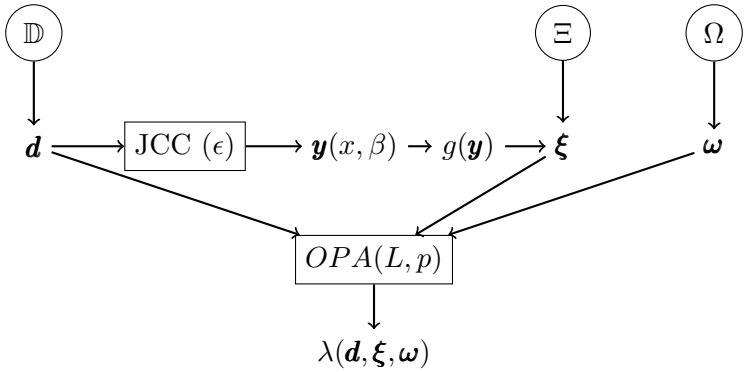


Figure 5.1: Random variable relationships and sources of uncertainty

model to take into account the effects of the particular line on the cascading process.

5.1.1 Overview of Sources of Uncertainty

We are capturing three separate sources of uncertainty and evaluating the stress on the power system by using the OPA cascading model as a surrogate process for the risk of a cascading power failure. The power system is affected by the uncertainty in demand and generation, denoted by random variable $\boldsymbol{d} = d + C_m\boldsymbol{\delta^m}$ with probability space associated with $\mathbb{D}$. We built this uncertainty into our model by assuming the net injection uncertainty is multivariate Gaussian, we calculate a system risk measure approximating the probability that one or more lines fail. This ties into the next source of uncertainty, encapsulated by the random variable $\boldsymbol{\xi}$ for probability space Ξ, which models the initial line failures that could initiate a cascading sequence. This random variable, $\boldsymbol{\xi}$, provides the connection between the JCC model of Chapter 4 with the OPA model from Chapters 2 and 3. The primary source of uncertainty modeled in Chapters 2 and 3 was encapsulated in the random variable $\boldsymbol{\omega}$ from a sample space Ω which governs the evolution of the cascading model. This cascading model can be seen as a surrogate for the stress on the power system in relation to rare event failures.

The JCC model incorporates uncertainty from random demand, $\boldsymbol{\delta^m}$. The outcome of the

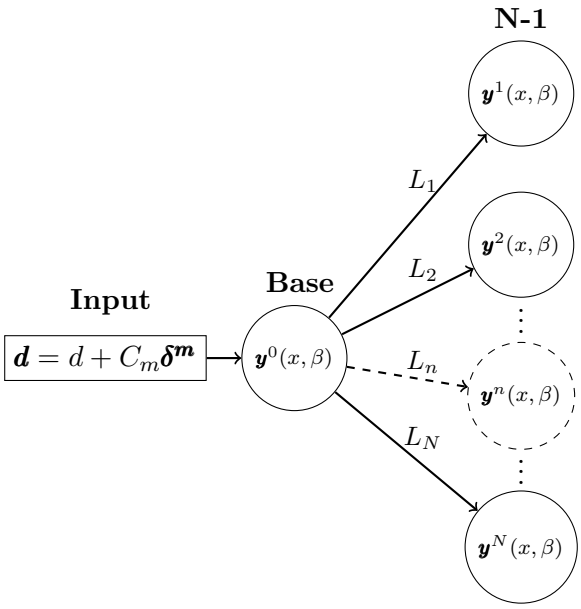


Figure 5.2: $N-1$ Exogenous Contingencies

JCC model are random variables for branch flows. Using a failure density function from the probability space associated with Ξ, we find a distribution of initial contingencies ξ. The OPA model incorporates uncertainty from all three sources, the demand $\mathbb{D}$, the initial contingencies Ξ, and the cascade evolution Ω. Let $\mathcal{X} = \mathbb{D} \times \Xi \times \Omega$ represent the whole sample space underlying the probability space for the OPA model, where $\boldsymbol{\chi} = (\boldsymbol{d}, \boldsymbol{\xi}, \boldsymbol{\omega})$ is an event in the larger sample space. The random variables' dependencies are laid out in Figure 5.1 that describe the relationships between variables.

5.1.2 $N-1$ Exogenous Contingencies

Reliability standards in power systems include $N-1$ contingency constraints. In this section, we describe how to extend the JCC model from Chapter 4 to the "security-cosntrained" or $N-1$ setting. The system must be robust to failures of each individual component due to reasons outside of the control of the system. The branch flows for line e given a contingency n, representing line n

failing, can be described with the following relationship

$$\boldsymbol{y}_e^n = \boldsymbol{y}_e + L_{en}\boldsymbol{y}_n \tag{5.1}$$

with L_{en} being a line outage factor for failure of line n's effect on line e. In Chapter 4 we saw the injection shift factor matrix A that gave the change in branch flows depending on changes in net injection. Here, we use the branch shift factor A^B where the individual entry $A^b_{e_1e_2} = \frac{dy_{e_1}}{dy_{e_2}}$ give the change in branch flow of branch e_1 when branch e_2 flow is changed and is detailed in [56]. We can calculate the branch shift factor by applying our branch incidence matrix to the injection shift factor in matrix form as

$$A^B = AC^T. \tag{5.2}$$

The line outage factor can be found by applying a scale factor so that when the line outage factor is multiplied by the branch flow, the response to all other branches is found. Specifically, we have

$$L_{en} = \begin{cases} -1 & \text{if } e = n \\ A^B_{en}(1 - A^B_{nn})^{-1} & \text{if } A^B_{nn} \neq 1 \\ \text{NaN} & \text{o/w} \end{cases} \tag{5.3}$$

Shift factors for multiple lines can be found provided certain conditions are met, as explained in [22]. However, line outage factors are typically only used in case of a small number of line failures. Note that lines do not have line outage factors when $A^B_{nn} = 1$. In our computational results, we filter out these scenarios for the $N-1$ analysis.

The mean flow on line e in contingency n can be found by taking the expectation over the uncertainty in net injections and using Equation (5.1) that describes flow for branch e in contingency

n.

$$\mathbb{E}_{\mathbb{D}}\left[\boldsymbol{y}_e^n\right] = \mathbb{E}_{\mathbb{D}}\left[\boldsymbol{y}_e\right] + L_{en}\mathbb{E}_{\mathbb{D}}\left[\boldsymbol{y}_n\right]. \tag{5.4}$$

We also need to understand the standard deviation of the branch flows $\boldsymbol{y}$ in the $N-1$ contingencies in order to form the JCC constraint for each contingency. Calculating the variance of $\boldsymbol{y}_e^n$ and expanding out using Equation (5.1), we see that the variance of branch flow is

$$\begin{aligned}
\operatorname{Var}[\boldsymbol{y}_e^n] &= \operatorname{Var}[\boldsymbol{y}_e] + L_{en}^2 \operatorname{Var}[\boldsymbol{y}_n] + 2L_{en} \operatorname{CoVar}[\boldsymbol{y}_e, \boldsymbol{y}_n] && \text{(5.5a)}\\
&= \pi_e^2\sigma_\Delta^2 - 2\pi_e\sigma_e^2 + \sigma_{ee}^2 \\
&\quad + L_{en}^2\left[\pi_n^2\sigma_\Delta^2 - 2\pi_n\sigma_n^2 + \sigma_{nn}^2\right] \\
&\quad + 2L_{en}\left[\pi_e\pi_n\sigma_\Delta^2 - \pi_e\sigma_{e_2}^2 - \pi_n\sigma_{e_1}^2 + \sigma_{e_1e_2}^2\right] && \text{(5.5b)}\\
&= \left[\pi_e^2 + 2L_{en}\pi_e\pi_n + L_{en}^2\right]\sigma_\Delta^2 \\
&\quad - 2\left[\pi_e\sigma_e^2 + L_{en}\pi_e\sigma_n^2 + L_{en}\pi_n\sigma_e^2 + L_{en}^2\pi_n\sigma_n^2\right] \\
&\quad + \sigma_{ee}^2 + 2L_{en}\sigma_{en}^2 + L_{en}^2\sigma_{nn}^2 && \text{(5.5c)}
\end{aligned}$$

by using equation 4.10 for the covariance between two branches. We can simplify the equation by substituting

$$\psi_{en} = \pi_e + L_{en}\pi_n \tag{5.6}$$

which captures the slack distribution response to aggregate demand for branch e in contingency n. Addditionally, we can pre-compute a new parameter $\sigma_{\psi_{en}}^2$, which is used in the branch covariance matrix and cutting plane algorithms. We again use the set of random generation and demand $\mathcal{M}$ and sum over every combination of $k_1, k_2 \in \mathcal{M}$.

$$\sigma_{\psi_{en}}^2 = \sum_{k_1\in\mathcal{M}}\sum_{k_2\in\mathcal{M}} \left(A_{ek_1} + L_{en}A_{nk_2}\right)^2 \Sigma_{k_1,k_2} \tag{5.7}$$

Then, we have

$$\text{Var}[\boldsymbol{y}_e^n] = \psi_{en}^2\sigma_\Delta^2 - 2\psi_{en}\left(\sigma_e^2 + L_{en}\sigma_n^2\right) + \sigma_{\psi_{en}}^2. \tag{5.8}$$

It is important to note that variance of line flow e in contingency n depends not only on the variance of the individual line flows but also on the covariance between branch flows e and n.

The following equations are added to the JCC model Equation (4.28) to describe the mean branch flows in the contingencies as well as the standard deviation of branch flows. Each contingency has separate line risk variables z_{en} and the system risk is constrained according to that contingencies specific risk level ϵ_n

$$\textbf{JCC N1:=}\ \min \sum_j \left[c_2\left(x_j^2 + \beta_j^2\sigma_\Delta^2\right) + c_1 x_j + c_0\right] \tag{5.9a}$$

Equations (4.28b) to (4.28k)

$$y_{en}^+ - y_e - L_{en}y_n \geq 0 \qquad \forall e, n \in \mathcal{E} \tag{5.9b}$$

$$y_{en}^+ + y_e + L_{en}y_n \geq 0 \qquad \forall e, n \in \mathcal{E} \tag{5.9c}$$

$$s_{en}^2 - \psi_{en}^2\sigma_\Delta^2 + 2\psi_{en}\left(\sigma_e^2 + L_{en}\sigma_n^2\right) \geq \sigma_{\psi_{en}}^2 \qquad \forall e, n \in \mathcal{E} \tag{5.9d}$$

$$z_{en} - \rho_e(y_{en}^+, s_{en}) \geq 0 \qquad \forall e, n \in \mathcal{E} \tag{5.9e}$$

$$\sum_e z_{en} \leq \epsilon_n \qquad \forall n \in \mathcal{E} \tag{5.9f}$$

In Equations (5.9b) and (5.9c), we enforce variable y_{en}^+ to be the absolute value of the power flow in branch e for contingency n. The standard deviation of branch flow e in contingency n is described by Equation (5.9d). Additionally, we have Equation (5.9e) which describes the individual line risk for every line in every contingency and Equation (5.9f) enforces that the system risk level for each contingency is below the chosen risk level ϵ_n.

Finally, in order to describe the convex, non-analytic risk function for the $N-1$ contingencies

Equation (5.9e), we use the same cutting planes from JCC Equation (4.30). The cutting planes are calculated at a specific value for slack distribution β and the line risk function is underestimated via the gradient. The cutting planes for the standard deviation of branch flow have changed due to the new covariance calculation taking into account the $N-1$ contingencies. These equations are as follows.

$$s_{en} \geq f_{en}(\beta) + \sum_j \frac{\partial f_{en}(\beta)}{\partial \beta_j}\left(\beta_j - \hat{\beta}_j\right) \tag{5.10a}$$

$$f_{en}(\beta) = \sqrt{\psi_{en}^2 \sigma_\Delta^2 - 2\psi_{en}\left(\sigma_e^2 + L_{en}\sigma_n^2\right) + \sigma_{\psi_{en}}^2} \tag{5.10b}$$

$$\frac{\partial f_{en}(\beta)}{\partial \beta_j} = \frac{\left(A_{ej} + L_{en}A_{nj}\right)\left(\psi_{en}\sigma_\Delta^2 - \left(\sigma_e^2 + L_{en}\sigma_n^2\right)\right)}{\sqrt{\psi_{en}^2 \sigma_\Delta^2 - 2\psi_{en}\left(\sigma_e^2 + L_{en}\sigma_n^2\right) + \sigma_{\psi_{en}}^2}} \tag{5.10c}$$

These equations are used in the cutting plane algorithm, where Equation (5.10a) gives an underestimator for the standard deviation of line e in contingency n. To form that equation, we calculate the standard deviation for a fixed slack distribution β in Equation (5.10b) and also the gradient of the standard deviation evaluated at a fixed β in Equation (5.10c).

5.1.3 Random Initial Contingencies for OPA

Ideally, given the random branch flow $\boldsymbol{y}_e$, the failure density function $g_e(\boldsymbol{y}_e)$ represents the probability that each individual line may fail given the flow $\boldsymbol{y}_e$. In order to make the connection to OPA, we use this sample space to seed the random initial contingencies for the OPA cascading model. In order to also capture the effects of events outside of our control, we perform this analysis under the standard $N-1$ contingencies. For each $N-1$ contingency, we have the probabilities that other lines may fail, given by $g_e(y_e^n)$ for branch e in contingency n. This means that our OPA cascading model can be seeded by contingencies with one or more initial line failures, governed by the probability distributions of branch flows and uncertainty coming from exogenous failures.

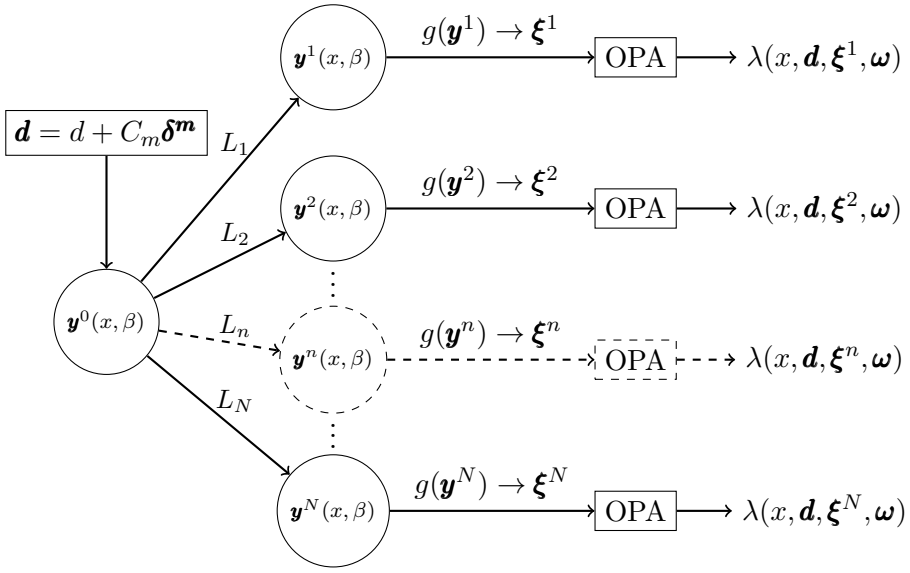


Figure 5.3: JCC $N-1$ Risk Model to seed random initial contingencies for OPA

The high-level sampling process is depicted in Figure 5.3. A brief explanation is given for the sampling algorithm used to seed the random initial contingencies for the OPA model. The initial contingencies depend on the uncertainty in demand $\boldsymbol{d}$ as well as the controllable injects and slack distribution (x, β). Using the DC assumption, this gives us random branch flows that follow a multivariate Gaussian distribution. We sample from this distribution to get a realization of branch flows, y_e, that wetransform via our failure density function $g(\cdot)$. Once we have $g_e(y_e)$, we can sample from our second source of uncertainty, Ξ, to get ξ_e, a vector of Bernoulli random variables taking 1 with the probabilities $g_e(y_e)$. This distribution seeds the OPA simulation and then the sampling of the cascade evolution Ω governs how the cascade evolves, as described in Chapter 3.

The OPA cascade in this sense is used as a surrogate function to represent rare event risk for a given topology. The following are the inputs to OPA model that has multiple stages and is solved via multiple LP programs. The first is the controllable generation x, which also indirectly controls the line failure distributions through the intermediate variable $\boldsymbol{y}$. The second input for the OPA model is $\boldsymbol{d}$, which is also used directly in our JCC model. The third input is the initial contingencies

$\boldsymbol{\xi}$, which depends on $\boldsymbol{y}$, and thus on x and $\boldsymbol{d}$. Finally, we have $\boldsymbol{\omega}$, which governs the evolution of the cascade process. The load shed for a particular realization of an OPA run is represented by λ and is the difference between the nominal load and the load at the last stage of the cascade in the OPA simulation.

$$\lambda(x, \boldsymbol{d}, \boldsymbol{\xi}, \boldsymbol{\omega}) \tag{5.11}$$

We are using the expected load shed as a weighting factor to measure the impact of a particular line failure on the OPA cascading process. For each contingency n, we have a different set of random initial contingencies due to the choice of generation x, slack distribution β and the resulting random flows $\boldsymbol{y}^n$ for each contingency. The expected load shed for contingency n is

$$f_n = \mathbb{E}_{\mathbb{D},\Xi,\Omega}\left[\lambda\left(x, \boldsymbol{d}, \boldsymbol{\xi}^n, \boldsymbol{\omega}\right)\right].$$

Algorithm 4 Contingency Sampling Algorithm for OPA. Given a dispatch (x, β), random demand $\boldsymbol{d} = d_0 + C_m\boldsymbol{\delta^m}$, sample $\boldsymbol{\xi}^n$ for T trials for each $N-1$ Contingency Figure 5.3

---

procedure SAMPLE($x, \beta, \boldsymbol{\delta^m}, T$)
  for $\forall t = 1, 2, ..., T^{\mathbb{D}}$ **do**
    Sample $a = (a_1, a_2, ..., a_N)^T$ from independent standard normal distributions.
    $HH^T = \Sigma^m$ using Cholesky decomposition
    $\delta^m = H * a$, which has the desired distribution due to affine transformation
    $\Delta = \sum_m \delta_m$
    $x = C_g(x_0 + \beta\Delta) - (d_0 + C_m\delta^m)$ using Equation (4.5)
    $y = y_0 + AC_G\beta\Delta - AC_m\delta^m$ using Equation (4.7)
    for $\forall n \in \mathcal{E}$s.t.$A^B_{nn} \neq 1$ **do**
      $L_{en} = A^B_{en}(1 - A^B_{nn})^{-1}$ using Equation (5.3)
      $y^n_e = y_e + L_{en}y_n$ using Equation (5.1)
      $z^n_e = g_e(y^n_e)$
      $\mathbb{O}^n \leftarrow \emptyset$
      for $\forall t = 1, 2, ..., T^{\Xi}$ **do**
        for $\forall e \in \mathcal{E}$ **do**

$$\xi_e = \begin{cases} 1 & \text{w/ prob. } z^n_e \\ 0 & \text{o/w} \end{cases}$$

        $\mathbb{O}^n \leftarrow \xi$
  The set $\mathbb{O}^n$ of initial contingencies for each $N-1$ contingency, representing distribution $\boldsymbol{\xi}^n$

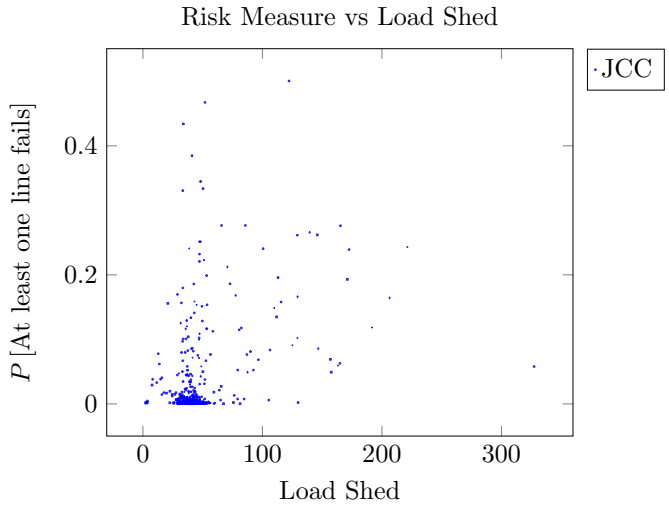


Figure 5.4: Line failure risk and OPA not correlated

5.1.4 OPA Weighting for JCC

While it is important that no lines fail, it may be overly restrictive while not actually reducing the risk of large load shedding events, which is a main concern from a system reliability perspective. It would certainly be more beneficial to keep the large high voltage lines in operation that are critical to system stability versus a few small distribution feeders, which may cause some small load shedding but would keep the bad events contained. To highlight the downside of chance constrained programming of not capturing the impact of the events, a scatter plot in Figure 5.4 of the system risk measure of the JCC model and the expected load shed of the OPA model seeded by the random initial contingencies from Algorithm 4, we see there is not a good correlation. We want to develop a system constraint that is correlated with the expected load shed from the cascading process.

For each line e, we construct a risk-weighting η_e such that the expected value of the load

shed distribution after losing that line is equal to η_e.

$$\sum_e \eta_e \mathbb{E}_{\mathbb{D}} \left[g(\boldsymbol{y}_e^n(x, \beta)) \right] \approx \mathbb{E}_{\mathbb{D}, \Xi^n, \Omega} \left[\lambda(x, \boldsymbol{d}, \boldsymbol{\xi}^n, \boldsymbol{\omega}) \right] \tag{5.12}$$

$$\eta^T z^n \approx f_n \tag{5.13}$$

There are many ways to find a weighting scheme to represent the impact from losing particular lines. Here is one example that attempts to capture the impact of losing a line with respect to the effect on the expected load shed of the resulting OPA simulations. For each contingency, a vector of risk levels, z^n, were recorded, which represent the expected probability of line e failing under contingency n (for contingency n, the probability of line n failing is $z_n^n = 1$). After performing the OPA simulations, we record the expected load shed $f_n(x)$ for that set of initial contingencies $\boldsymbol{\xi}^n$. We formulate the following linear system Section 5.1.4 and solve for η.

$$\begin{bmatrix} z_1^1 & \cdots & z_{N_l}^1 \\ \vdots & \vdots & \vdots \\ z_1^{N_l} & \cdots & z_{N_l}^{N_l} \end{bmatrix} \begin{bmatrix} \eta_1 \\ \vdots \\ \eta_{N_l} \end{bmatrix} = \begin{bmatrix} f_1 \\ \vdots \\ f_{N_l} \end{bmatrix}$$

After applying these new linear weights η to the line risks z, we find that our new system constraint $\sum_e \eta_e z_e$ is now well correlated with the resulting expected value of load shed of the cascading process as shown in Figure 5.5. In order to solve problems using this weighting scheme, it is nearly identical to JCC $N-1$ modelEquation (5.9), except with the system risk constraint Equation (5.9e) representing probability of one or more line failures with the new risk constraint representing the expected value of load shed Equation (5.12).

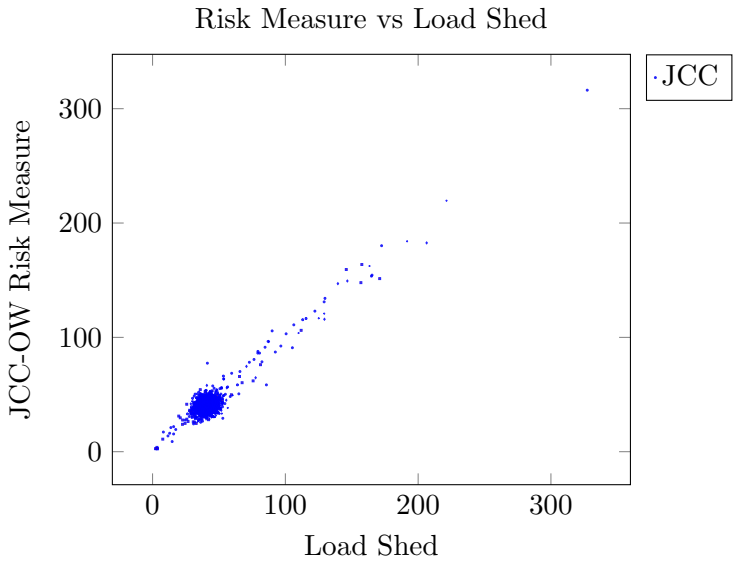


Figure 5.5: JCC $N-1$ OW correlated with OPA

$$\textbf{JCC } N-1 \textbf{ OW:=} \min_{(x,\beta,;\theta,y,\pi,s,z)} \sum_j \left[c_2\left(x_j^2+\beta_j^2\sigma_\Delta^2\right)+c_1x_j+c_0\right] \tag{5.14a}$$

$$\textit{Equations } (4.28b) \textit{to } (4.28k)$$

$$\textit{Equations } (5.9b) \textit{to } (5.9e)$$

$$\sum_e \eta_e z_{en} \leq \zeta_n \qquad \forall n \in \mathcal{E} \tag{5.14b}$$

5.1.5 Cutting Plane Algorithm for JCC $N-1$ with OPA Weighting

Now, we describe the algorithm used to solve JCC $N-1$ Equation (5.9) and the new JCC OW Equation (5.14). The algorithm for JCC OW is shown explicitly in Algorithm 5 and the JCC $N-1$ is nearly identical except for the difference in system risk constraints.

Algorithm 5 This cutting plane algorithm solves JCC Equation (4.28) with $N-1$ contingencies Equation (5.9) using the OPA weighting scheme Equation (5.12) via linear programs and cutting planes

procedure JCC $N-1$ OW(d,Σ^m,ζ,ζ_n,L,p)
  $L^n \leftarrow \emptyset$ (Set of Lines with potential risk for contingency n)
  $S \leftarrow \emptyset$ (Set of Cuts)
  $r \leftarrow 0$ (Risk)
solve:
  $(\hat{x}, \hat{\beta}, \hat{y}) \leftarrow$Solve DC Power Flow, Equations (4.28a) to (4.28g), with cuts S, risk $r \leq \epsilon$
  if Infeasible **then return** Problem Infeasible
  $N_f \leftarrow 0$
  for $\forall n$ **do**
    Calculate branch flows $\hat{y^n}$ for contingency n using Equation (5.1)
    Calculate $\hat{s^n}, \hat{z^n}, \hat{r^n}$ using $(\hat{x}, \hat{\beta}, \hat{y})$ and Equation (5.8)
    if $\hat{r} \geq \epsilon + tol$ **then** $N_f = N_f + 1$
    for $\forall e$ **do**
      if $\hat{z_e^n} \geq tol$ **then**
        if $e \notin L^n$ **then**
          $L^n \leftarrow \{L^n, e\}$
          Initialize s_e^n, z_e^n
          $r^n \leftarrow r^n + z_e^n$
        $S \leftarrow$ line risk cuts Equation (4.30) for z_e^n, y_e^n, s_e^n dependent on $\hat{z}_e^n, \hat{y}_e^n, \hat{s}_e^n$
        $S \leftarrow$ branch variance cuts Equation (5.10) for s_e^n, β_e dependent on $\hat{s}_e^n, \hat{\beta}_e$
  if $N_t = 1$ **then return** Optimal $(\hat{x}, \hat{\beta}, \hat{y})$
  else
    goto *solve*

5.2 Computational Experiments

This computational section shows an example of using the OPA weighting on top of the JCC model enchanced with $N-1$ contingency considerations. We began by running JCC $N-1$ and used it to initiate the $N-1$ OPA sampling process and ran OPA on each initial contingency. We created an OPA weighting vector as described in Section 5.1.4. Then we solved JCC OW Equation (5.14) using cutting plane algorithm Algorithm 5. Finally, we ran the $N-1$ OPA process again with the two models to determine load shed distribution associated with the two dispatch points. The goal for the JCC OW model is to modify the distribution of load shed and reduce the risk of rare event failures that are large.

To highlight the differences of the JCC OW model, this trial gave a higher budget to the JCC OW model and constrained the new system risk measure so that the cost of dispatch was 4% higher than the standard JCC model. In return, it was able to reduce the expected value of load shed by 4.7%, and reduce the 99 percentile of load shed by 28.5%. Some statistical measures of the load shed distribution are shown in Table 5.1 and plotted in Figure 5.6. In the plot Figure 5.6, we have the load shed distribution shown on the left where each picture is zoomed in to see the tail of the distribution. On the right side, we have a log plot of the number of trials that are in each bin. In Table 5.1, we have the cost of the dispatch point as well as statistics on the resulting load shed distribution. We have tabulated the mean, the number of trials with no load shed, the 95th, 99th, and 99.8th percentiles. Additionally, we have the maximum over all trials as well as the conditional value at risk of the 95th percentile.

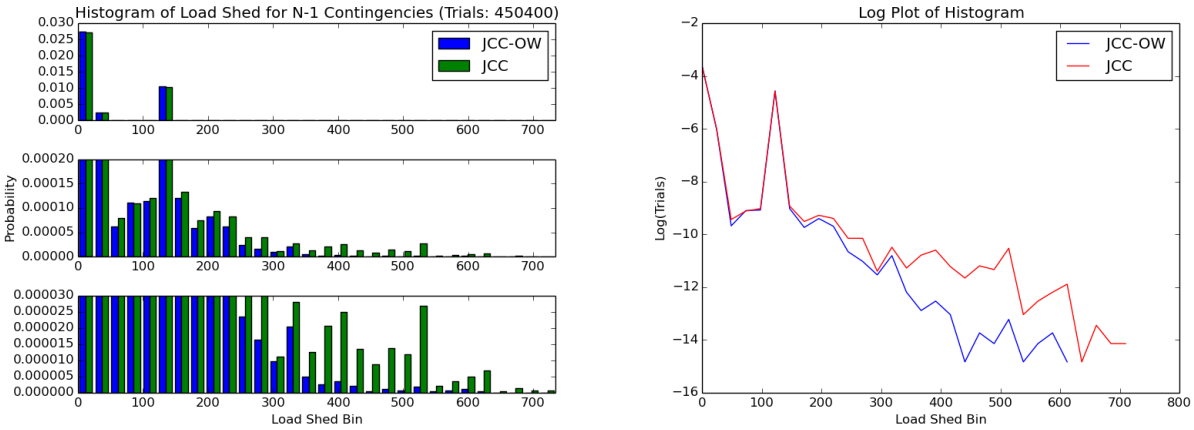


Figure 5.6: Load shed distribution

Trials **450400**

Stat	JCC	JCC-OW	diff (%)
cost	1.78e6	1.85e6	-3.9
mean	44.0	41.9	4.8
No Load Shed	199444	201984	1.3
P95	138.81	138.81	0
P99	205.77	147.12456	28.5
P99.8	441.21	251.76	43
max	734.29	612.9	16.5
CVaR95	182.5	155.6	14.7

Table 5.1: Rare event risk metrics

5.3 Conclusion

In this Chapter, we extended the JCC model to include the security constrained, $N-1$ contingencies. Additionally, we take into account the approximate impact of losing subsets of lines on the resulting OPA cascading process. We use the OPA cascading process as an empirical model to represent how stressed the grid is with respect to rare event failures. The OPA process gives us a distribution of load sheds and we use this distribution to develop risk-weighting factors for the lines. The JCC OW model, which incorporates this linear weighting scheme, was able to change the distribution (for an increased cost) of the resulting load shed by reducing the mean and, even more so, reducing the tail events with large load shed.

Chapter 6

Conclusion

In this thesis we explored rare event risk on the bulk power system. From the recent literature of cascading power failures, we built upon the OPA cascading power failure simulation. We started by modeling this process as a multi-stage stochastic program with integer variables in Chapter 2. We introduced the concept of effective capacity to model the decision-dependent uncertain inherent in this cascading process. This model allows for the flexibility of using the cascading process as a sub-problem in long term design problems. We then turn to Monte Carlo simulation in order to parallelize the computational process to get better resolution of the cascade evolution uncertainty in Chapter 3. We look at the effects of transmission expansion on the OPA simulation and optimize this using derivative free optimization techniques and large computational resources through HTCondor. In the second half of the thesis, we switch to real-time dispatch models. In Chapter 4 we develop a system risk measure that constrains the probability of one or more lines failing. Also, we model uncertainty in generation and demand and translate this to uncertainty in branch flows by calculating the branch covariance matrix. Using these two models, we approximate this system risk measure under uncertainty and use a cutting plane algorithm to solve this convex

problem. Finally, in Chapter 5 we combine the rare event risk model of the OPA simulation with the real-time line failure dispatch model. We extend the joint chance constraint to the N-1 contingencies and use these contingencies to seed the OPA model. Based off of these results, we develop a linear weighting system to approximate how important each line is with respect to the OPA cascade simulation. We use this linear weighting system and a cutting plane algorithm to solve this convex optimization problem.

6.1 Contributions

Modeling Casacading Power Failures

In Chapter 2, we explore the models of power flow over topological networks, cascading risk models, and a system equilibrium model (OPA) in which there is a balance found between economic efficiency and reliability. For economic reasons, power systems move towards critical points, which are points of maximum throughput for the given infrastructure and are characterized by power flow being limited by either transmission constraints or generation capacity limits. A critical point primarily due to transmission constraints lead to larger blackouts, however infrequently, and a critical point primarily due to generation capacity lead to smaller blackouts that are more frequent. In the OPA model, there is an engineering response to these blackouts to improve the infrastructure. An equilibrium is found in the distribution of blackout size such that the load shed for these events follow a power tail distribution. This equilibrium has been seen in the real world, the distribution of load shed events follow a power tail distribution with exponent $-1.3 \pm .2$ and has stayed the same for 30 years. We model the short term cascading process as a multi-stage stochastic program that can be embedded in design problems.

The primary contribution was to model the existing cascading power failure simulation as a

multi-stage stochastic program using mixed integer variables. The primary difficulty was overcoming the decision dependent uncertainty, which was done by using the concept of effective capacity distribution and a priori sampling. While the computational difficulty made practical use of this impossible for realistically-sized power systems. The analytical modeling of the simulation process and decision dependent uncertainty will hopefully find application in other power system problems.

Using the OPA Simulation for Transmission Expansion

In Chapter 3, we decompose the MSIP model by scenario and use a fast Monte Carlo simulation approach that is parallelizable to evaluate the load shed distribution given demand and contingency scenarios. Here we aim to improve the system before an event occurs. As we saw in the Northeast Blackout of 2003, once things go wrong, events that the system is typically robust to can compound and eventually lead to large scale blackout. For example, after losing the Stuart-Atlanta 345 kV line, the fact that MISO was unaware of this event, led to the inability to provide support in dealing with this problem. One way to reduce the likelihood of these rare events is to reduce the likelihood of the initiating contingencies. We proceeded to explore this load shed distribution by looking at the design problem of increasing capacity on transmission elements and the effects it has on the OPA simulation. This OPA simulation has a natural connection with our risk model that we explore in this chapter.

The primary contribution was to optimize over the functional landscape defined by OPA simulation efficiently. We looked at the design problem of transmission expansion and explored the characteristics of this risk function. We developed an efficient implementation with a common random number scheme to reduce the variance of function estimates and effective software engineering to allow for massively parallel solution methodology. In addition, DFO techniques were used to filter the high frequency noise and exploratory steps using accessory information from the

OPA simulation enhanced the solution time. A computational experiment demonstrating how the system could effectively find system configurations improving load shed was performed.

Line Failure Risk Models for Real-Time Dispatch

In Chapter 4, we developed a model to constrain the probability that one or more lines fail to be small. This is in line with the idea in Chapter 3 where we would like to reduce the likelihood of initiating events occuring. To start, we noted that due to the increased penetration in renewables and modern technologies of electric vehicles and energy storage, it is vital to include uncertainty of net injections into the model. We then look at the chance constraint models in literature that deal with this uncertainty in net injections. Another interesting class of models in literature approached the reliability problem from a system perspective. Since we don't want any lines to fail, we develop a joint chance constraint on the probability that one or more lines fail. In order to model this, we assume that the failure density function of an individual line is a piecewise linear function and that line failure probabilities are independent of each other given the branch flow. However, the branch flow is certainly correlated, as it is the flows on fixed topological structure with fixed power flow parameters. For the DC approximation of power flows, the branch flows follow a linear relationship with net injections. If we assume that the net injections are a multivariate Gaussian with a known covariance matrix it follows that the branch flows are multivariate Gaussian and we can calculate its covariance matrix. We now make our first approximation by taking a linearization ofthesystem risk measure using a Taylor expansion and dropping higher order terms. This is a very good approximation for small failure probabilities and it allows us to pass this multivariate Gaussian through our piecewise linear failure density function. We finish this model by calculating the expected probability of a line failing for each branch and then sum over all branches to get our system risk measure, which is constrained.

The primary contribution was to develop a dispatch model that included a system risk constraint to control endogenous risk from line loading. This system risk measure was modeled as a joint chance constraint that we can solve exactly when there is no uncertainty in generation and demand. When there is net injection uncertainty, we approximate the system risk measure and solve the model approximately using a cutting plane algorithm. The cost-risk frontier was explored and the new model was compared to the standard chance constrained model used in recent literature.

Reducing Cascading Risk Through Real-Time Dispatch

In Chapter 5, we extended our endogenous system risk model to the N-1 contingencies. We also evaluate the N-1 contingencies using OPA to evaluate rare event risk. We then develop a linear weighting system to account for this rare event risk and constrain it in our real time dispatch model. Computational experiments suggest that the JCC N-1 with OPA weighting model may have more desirable load-shed distribution.

6.2 Future Work

The Chapters 2 and 3 can be improved by incorporating a more detailed simulation of cascading power failures including risk from voltage collapse, power transients, and other physical risk leading to large load shed events. Chapters 4 and 5 can be improved by incorporating covariance between effective capacities and demand uncertainty. Geographically correlated weather has an effect on both transmission capacities as well as generation and demand. Additionally, the real time model should include contingencies of generator failures as well. The model is dependent on the chosen line failure density model and it will be useful to better understand this function. Determining approximate values for the function parameters that represent real world conditions will be useful.

Bibliography

[1] U.S. Energy Information Agency. `http://www.eia.gov/electricity/data/eia826/`. Accessed: 2014-11-30.

[2] U.S. Energy Information Agency. *Independent Statistics and Analysis: www.eia.gov*. U.S. Department of Energy, 2013.

[3] Reka Albert, Istvan Albert, and Gary L. Nakarado. Structural vulnerability of the North American power grid. *Physical Review E*, 69:1–4, 2004.

[4] Patrick L. Anderson and Ilhan K. Geckil. Northeast Blackout Likely to Reduce U.S. Earnings by $6.4 Billion. *Anderson Economic Group*, 2003.

[5] A. R. Bergen. *Power System Analysis*. Prentice-Hall, 1986.

[6] Gerald Berman. Minimization by successive approximation. *SIAM Journal on Numerical Analysis*, 3(1):123–133, 1966.

[7] Gerald Berman. Lattice approximations to the minima of functions of several variables. *Journal of the ACM (JACM)*, 16(2):286–294, 1969.

[8] D. Bienstock, M. Chertkov, and S. Harnett. Chance Constrained Optimal Power Flow: Risk-Aware Network Control under Uncertainty. *ArXiv e-prints*, September 2012.

[9] Daniel Bienstock. Optimal control of cascading power grid failures. *50th IEEE Conference on Decision and Control and European Control Conference*, pages 2166–2173, 2011.

[10] M. Bucher, M. Vrakopoulou, and G. Andersson. Probabilistic n-1 security assessment incorporating dynamic line ratings, pes 2013. In *IEEE Power and Energy Society General Meeting*, 2013.

[11] B. A. Carreras, V. E. Lynch, I. Dobson, and D. E. Newman. Critical points and transitions in an electric power transmission model for cascading failure blackouts. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 12(4):985–994, 2002.

[12] B. A. Carreras, V.E. Lynch, I. Dobson, and D. E. Newman. Complex dynamics of blackouts in power transmission systems. *Chaos*, 14(3):643,652, September 2004.

[13] Jie Chen, James S. Thorp, and Ian Dobson. Cascading dynamics and mitigation assessment in power system disturbances via a hidden failure model. *International Journal of Electrical Power and Energy Systems*, 27(4):318 – 326, 2005.

[14] Andrew R. Conn, Katya Scheinberg, and Luis N. Vicente. *Introduction to Derivative-Free Optimization*. SIAM and MPS, Philadelphia, PA, 2009.

[15] Eduardo Cotilla-Sanchez, Paul D. H. Hines, Clayton Barrows, and Seth Blumsack. Comparing the topological and electrical structure of the North American electric power infrastructure. *IEEE Systems Journal*, 6:616–626, 2012.

[16] North American Electric Reliability Council. *Disturbance Analysis Working Group Database: www.nerc.com*. NERC, 2013.

[17] Jian Ding, Xiaomin Bai, Wei Zhao, Zhu Fang, Zaihua Li, and Min Liu. The improvement of the small-world network model and its application research in bulk power system. *Power System Technology*, pages 1–5, 2006.

[18] I. Dobson, B. A. Carreras, V. E. Lynch, and D. E. Newman. An initial model for complex dynamics in electric power system blackouts. In *34th Hawaii International Conference on System Science*, pages 705,709, January 2001.

[19] Ian Dobson, Benjamin A. Carreras, Vickie E. Lynch, and David E. Newman. Complex systems analysis of series of blackouts: Cascading failure, critical points, and self-organization. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 17(2):–, 2007.

[20] Leonardo Duenas-Osorio and Srivishnu Mohan Vemuru. Cascading failures in complex infrastructure systems. *Structural Safety*, 31:157–167, 2009.

[21] D.H.J. Epema, M. Livny, R. van Dantzig, X. Evers, and J. Pruyne. A worldwide flock of Condors: Load sharing among workstation clusters. *Future Generation Computer Systems*, 12:53–65, 1996.

[22] T. Guler, G. Gross, and Minghai Liu. Generalized line outage distribution factors. *Power Systems, IEEE Transactions on*, 22(2):879–881, May 2007.

[23] Paul Hines, Jay Apt, and Sarosh Talukdar. Trends in the history of large blackouts in the United States. *Power and Energy Society General Meeting*, pages 1–8, 2008.

[24] Paul Hines, Jay Apt, and Sarosh Talukday. Large blackouts in North America: Historical trends and policy implications. *Energy Policy*, 37:5249–5259, 2009.

[25] Paul Hines, Eduardo Cotilla-Sanchez, and Seth Blumsack. Do topological models provide good information about electricity infrastructure vulnerability? *Chaos*, 20:1–5, 2010.

[26] Paul Hines, Eduardo Cotilla-Sanchez, and Seth Blumsack. Topological models and critical slowing down: Two approaches to power system blackout risk analysis. In *Proceedings of the Annual Hawaii International Conference on System Sciences*, pages University of Hawai'i at Manoa, Shidler College of Business –, Koloa, Kauai, HI, United states, 2011.

[27] Robert Hooke and To A Jeeves. "direct search"solution of numerical and statistical problems. *Journal of the ACM (JACM)*, 8(2):212–229, 1961.

[28] R. Kinney, P. Crucitti, R. Albert, and V. Latora. Modeling cascading failures in the North American Power Grid. *European Physics Journal*, 46:101–107, 2005.

[29] T. Kolda, R. Lewis, and V. Torczon. Optimization by direct search: New perspectives on some classical and modern methods. *SIAM Review*, 45(3):385–482, 2003.

[30] Kristina Hamachi LaCommare and Joseph H. Eto. Cost of power interruptions to electricity consumers in the United States. *Energy*, 31:1845–1855, 2006.

[31] Averill M. Law. *Simulation Modeling and Analysis*. McGraw-Hill, New York, NY, 2007.

[32] Robert Michael Lewis, Virginia Torczon, and Michael W. Trosset. Direct search methods: then and now. *Journal of Computational and Applied Mathematics*, 124:191–207, 2000.

[33] S. Mei, Yixin Ni, Gang Wang, and Shengyu Wu. A study of self-organized criticality of power system under cascading failures based on ac-opf with voltage stability margin. *IEEE Trans. on Power Systems*, pages 1719–1726, 2008.

[34] S. Mei, Yadana, Yadana, Xiao feng Weng, and An cheng Xue. Blackout model based on opf and its self-organized criticality. *Chinese Control Conference*, pages 1673–1678, 2006.

[35] Shengwei Mei, Fei He, Xuemin Zhang, Shengyu Wu, and Gang Wang. An improved opa model and blackout risk assessment. *IEEE Transactions on Power Systems*, 24:814–823, 2009.

[36] Dusko P. Nedic, Ian Dobson, Daniel S. Kirschen, Benjamin A. Carreras, and Vickie E. Lynch. Criticality in a cascading failure blackout model. *Electric Power and Energy Systems*, 28:626–633, 2006.

[37] D.E. Newman, B.A. Carreras, N.S. Degala, and I. Dobson. Risk metrics for dynamic complex infrastructure systems such as the power transmission grid. In *Proceedings of the Annual Hawaii International Conference on System Sciences*, pages 2082 – 2090, Maui, HI, United states, 2011.

[38] Junjian Qi, Shengwei Mei, and Feng Liu. Blackout model considering slow process. Accepted, 2013.

[39] L. Roald, F. Oldewurtel, T. Krause, and G. Andersson. Analytical reformulation of security constrained optimal power flow with probabilistic constraints. In *PowerTech (POWERTECH), 2013 IEEE Grenoble*, pages 1–6, June 2013.

[40] Jian-Wei Wang Li-Li Rong. A model for cascading failures in scale-free networks with a breakdown probability. *Physica A*, 388:1289–1298, 2009.

[41] Conrad Sanderson. Armadillo: An open source c++ linear algebra library for fast prototyping and computationally intensive experiments. *Technical Report, NICTA*, 2010.

[42] Tapani O Seppa. Reliability and real time transmission line ratings. 2007.

[43] H.J. Sun, H. Zhao, and J.J. Wu. A robust matching model of capacity to defense cascading fialure on complex networks. *Physics A*, 387:6431–6435, 2008.

[44] Todd Tannenbaum, Derek Wright, Karen Miller, and Miron Livny. Condor – a distributed job scheduler. In Thomas Sterling, editor, *Beowulf Cluster Computing with Linux*. MIT Press, October 2001.

[45] U.S.-Canada Power System Outage task Force. *Final Report on the August 14, 2003 Blackout in the United States and Canada: Causes and Recommendations*. U.S. Department of Energy and Natural Resources Canada, 2004.

[46] Virginia Torczon. On the convergence of pattern search algorithms. *SIAM Journal on optimization*, 7(1):1–25, 1997.

[47] M. Vrakopoulou, S. Chatzivasileiadis, and G. Andersson. Probabilistic security-constrained optimal power flow including the controllability of hvdc lines. In *Innovative Smart Grid Technologies Europe (ISGT EUROPE), 2013 4th IEEE/PES*, pages 1–5, Oct 2013.

[48] M. Vrakopoulou, S. Chatzivasileiadis, E. Iggland, M. Imhof, T. Krause, O. Makela, J.L. Mathieu, L. Roald, R. Wiget, and G. Andersson. A unified analysis of security-constrained opf formulations considering uncertainty, risk, and controllability in single and multi-area systems. In *Bulk Power System Dynamics and Control - IX Optimization, Security and Control of the Emerging Power Grid (IREP), 2013 IREP Symposium*, pages 1–19, Aug 2013.

[49] M. Vrakopoulou, K. Margellos, J. Lygeros, and G. Andersson. A probabilistic framework for reserve scheduling and n-1 security assessment of systems with high wind power penetration. *Power Systems, IEEE Transactions on*, 28(4):3885–3896, Nov 2013.

[50] Kongsen Wang, Gehao Sheng, and Xiuchen Jiang. Risk assessment of transmission dynamic line rating based on monte carlo. In *Power Engineering and Automation Conference (PEAM), 2011 IEEE*, volume 2, pages 398–402, Sept 2011.

[51] Q. Wang, G. Zhang, J.D. McCalley, T. Zheng, and E. Litvinov. Risk-based locational marginal pricing and congestion management. *Power Systems, IEEE Transactions on*, PP(99):1–11, 2014.

[52] Qin Wang, J.D. McCalley, Tongxin Zheng, and E. Litvinov. A computational strategy to solve preventive risk-based security-constrained opf. *Power Systems, IEEE Transactions on*, 28(2):1666–1675, May 2013.

[53] T. Yip, Chang An, G.J. Lloyd, M. Aten, and B. Ferri. Dynamic line rating protection for wind farm connections. In *Integration of Wide-Scale Renewable Resources Into the Power Delivery System, 2009 CIGRE/IEEE PES Joint Symposium*, pages 1–5, July 2009.

[54] Jun Zhang, Jian Pu, J.D. McCalley, H. Stern, and Jr. Gallus, W.A. A bayesian approach for short-term transmission line thermal overload risk assessment. *Power Delivery, IEEE Transactions on*, 17(3):770–778, Jul 2002.

[55] Liang Zhao, Kwangho Park, and Ying-Cheng Lai. Attack vulnerability of scale-free networks due to cascading breakdown. *Physical Review*, 70:1–4, 2004.

[56] Ray D. Zimmerman and Carlos E. Murillo-Snchez. Matpower 4.1 user's manual, 2011.

Appendices

Appendix A

Code for Computational Models

A.1 Multi Stage Stochastic Program

This code is publicly available on GitHub at `https://github.com/eanderson4/msip`

A.2 HTCondor Parallelization Code

```
#!/progs/gurobi/linux64/bin/python2.7

import sys


if __name__=="__main__":

    if len(sys.argv) > 1:
        #Get Log File
        flog = open(sys.argv[1],'r')
        log = flog.readlines()
        flog.close()
        #Use last line as arguments
        args = log[-1].split(" ")[1:]
        runit_args = [ arg.split("\n")[0] for arg in args ]

        if 'start' in runit_args:
            #Check Condor Queue
            fcon = open(sys.argv[2],'r')
            con = fcon.readlines()
            fcon.close()
            jobsrunning= con[-1].split(";")[1].split(", ")[3]
            jobsidle= con[-1].split(";")[1].split(", ")[2]
            jobsheld= con[-1].split(";")[1].split(", ")[4]
            sumo = int(jobsidle.split(" ")[0]) + int(jobsrunning.split(" ")[0]) + int(
    jobsheld.split(" ")[0])
```

```
            #If no jobs are running, idle, or held, output -notrunning
            if sumo == 0:
                fout = open(sys.argv[1],'a')
                fout.write('-notrunning\n')
                fout.close()


        if 'check_queue' in runit_args:
            #Check Condor Queue
            fcon = open(sys.argv[2],'r')
            con = fcon.readlines()
            fcon.close()
            jobsrunning= con[-1].split(";")[1].split(", ")[3]
            jobsidle= con[-1].split(";")[1].split(", ")[2]
            jobsheld= con[-1].split(";")[1].split(", ")[4]
            print jobsidle,jobsrunning,jobsheld
            sumo = int(jobsidle.split(" ")[0]) + int(jobsrunning.split(" ")[0]) + int(
    jobsheld.split(" ")[0])
            #If no jobs are running, idle, or held, output -dosomething
            if sumo == 0:
                fout = open(sys.argv[1],'a')
                fout.write('-dosomething\n')
                fout.close()

        if 'iteration' in runit_args:
            fcheck = open(sys.argv[2],'r')
            check = fcheck.readlines()
            fcheck.close()
            fout = open(sys.argv[1],'a')
            maxim = 0
            if check:
                for c in check:
                    print c.split('step')[1].split('\n')[0]
                    test = int(c.split('step')[1].split('\n')[0])
                    if test > maxim:
                        maxim=test
                fout.write(str(maxim))
            else:
                print "HERE"
                print check
                fout.write(str(-1))
            fout.close()


    else:
        "Usage: Proc.py <log input> <condor_q in>"
```

Listing A.1: proc.py: HTCondor Queue and Log File Reader

```
#! /bin/sh
if [ -n "$1" ]
then
    echo "Some input entered"
    echo $1
    FILE=$1
else
    echo "no input entered"
    FILE="runitlog"
fi


if [ -f $FILE ];
then
    echo "File $FILE exists, need different name"
else
    echo "Using $FILE for runit log"
fi

```

```
isNotDone=true

#Run proc.py to find if starting fresh or resuming old run
echo "-proc.py start" > $FILE
condor_q eanderson > condor_q_output
Proc.py $FILE condor_q_output
command=$(tail -1 $FILE)
echo $command
iteration=0
ls -d step* > tempfile
echo "-proc.py iteration" > $FILE
Proc.py $FILE tempfile

#Set iteration based on proc.py output, a new run (0) or resuming old run at iteration (n)
iteration=$(tail -1 $FILE)
echo $iteration
sleep 3

#If starting a new run, initialize and run algorithm
if [ "-notrunning" = "$command" ] && [ $iteration = -1 ]
then
cd Powerin
echo "consub_1d"
./consub_1d.py -c grid2.gr LINESIN
sleep 5
cd ../
mkdag
sleep 3
mkdag --data=Powerin --outputdir=step0 --cmdtorun=arrive.py --pattern=want.lao --pattern=
    want.lsa --type=Other
cd step0
sleep 5
condor_submit_dag mydag.dag
sleep 7
cd ../
((iteration++))
fi


loop=0
#Loop forever
while $isNotDone; do
    let loop+=1

    #Check condor queue and run proc.py to develop command
    echo "LOOP: $loop" >> $FILE
    echo "-condor_q" >> $FILE
    condor_q eanderson > condor_q_output
    echo "-proc.py check_queue" >> $FILE
    Proc.py $FILE condor_q_output
    command=$(tail -1 $FILE)

    #If nothing is running in condor, do something
    if [ "-dosomething" = "$command" ]
    then

  #Move files if iteration 0 is done and clean folder directory
  if [ $iteration = 0 ]
  then
      cd Powerin
      consub_1d.py -p grid2.gr LINESIN
      sleep 7
      cp 1d*dat ../step0/
      cp subset.lines ../step0/
      sleep 7
      consub_1d.py -clean grid2.gr LINESIN
      rm ./1d_*dat
      rm LIST_*
```

```
  #Move files and clean folder director for nth iteration
  else
      cd Powerin
      allocate.py -p allocate.lines ../step$iteration
      sleep 5
      transposeFile.py allocate.lines ../step$iteration/
      sleep 7
      cp n*dat ../step$iteration/
      cp allocate.lines ../step$iteration
      cp allocate.picks ../step$iteration
      cp search.lines ../step$iteration
      cp point.cap ../step$iteration
      cp point.lsa ../step$iteration
      cp point.lao ../step$iteration
      sleep 5
      allocate.py -clean allocate.lines
      rm n*dat
  fi


  #Develop pattern for pattern search
  if [ $iteration = 0 ]
  then
      allocate.py -picks subset.lines ../step0/1d_
      allocate.py -c subset.lines ../step0/1d_
  else
      allocate.py -point allocate.lines ../step$iteration/
      sleep 10
      allocate.py -step point.cap search.lines
      sleep 10

  fi
  sleep 25

  #Increment iteration and run parallization through dag
  ((iteration++))
  cd ..
  mkdag --data=Powerin --outputdir=step$iteration --cmdtorun=arrive.py --pattern=want.lao --
    pattern=want.lsa --type=Other --maxidle=500
  cd step$iteration
  sleep 7
  condor_submit_dag mydag.dag
  cd ..
  sleep 5
    fi
    sleep 7


done
```

Listing A.2: runit: Main Process Flow

```
#Power System Tools

class Bus:
    """Stores bus information"""
    def __init__(self, ID, name, pmin, pmax):
        self.ID = ID
        self.name = name
        self.pmin = pmin
        self.pmax = pmax
    def __str__(self):
        return self.ID + ": " + self.name + "  ( "+ self.pmin +" , "+ self.pmax + " ) "

class Branch:
    """Stores branch information"""
    def __init__(self, ID, bus1, bus2, limit, X):
        self.ID = ID
```

```
17          self.bus1 = bus1
18          self.bus2 = bus2
19          self.limit = limit
20          self.X = X
21
22      def __str__(self):
23          return self.ID + ": " + self.bus1 +" -> " +self.bus2 + " , " +self.limit+ " ("+self.
    X+")"
24
25  class Grid:
26      """Stores grid information"""
27      def printBranch(self, ID):
28          print  self.branches[ ID ]
29      def printBus(self, ID):
30          print  self.buses[ ID ]
31
32      def getBranchLimit(self, ID):
33          return self.branches[ ID ].limit
34
35      def readData(self,fin):
36          f = open(fin,'r')
37          l = f.readline()
38          while l:
39              if 'GRIDDATA' in l:
40                  print 'Grid data ID: #Buses #Branches'
41                  #read in grid information
42                  l = f.readline()
43                  temp = l.split(" ")
44                  self.ID=temp[1]
45                  self.Nbus=temp[2]
46                  self.Nbranch=temp[3].split("\r")[0]
47                  print self.ID,": ",self.Nbus, self.Nbranch
48
49
50              if 'BUSDATA' in l:
51                  print 'Bus data  ID: name  (pmin, pmax)'
52                  #read in bus information and store in dictionary
53                  l = f.readline()
54                  temp = l.split(" ")
55                  b = Bus(-1,0,0,0)
56                  self.buses = { b.ID : b }
57                  while temp[0]=='d':
58                      #parse bus string
59                      bus = Bus(temp[1], temp[3], temp[4], temp[5].split("\r")[0] )
60                      self.buses[ bus.ID ] = bus
61                      print bus
62                      temp = f.readline().split(" ")
63
64
65              if 'BRANCHDATA' in l:
66                  print 'Branch data  ID: bus1 -> bus2 , capacity (X)'
67                  #read in branch information
68                  l = f.readline()
69                  temp = l.split(" ")
70                  b = Branch( -1, 0, 0, 0, 0 )
71                  self.branches = { b.ID : b}
72                  while temp[0]=='d':
73                      #parse branch string
74                      branch = Branch(temp[1], temp[2], temp[3], temp[4], temp[5].split("\r")
    [0] )
75                      self.branches[branch.ID] = branch
76                      print branch
77                      temp = f.readline().split(" ")
78
79
80              l = f.readline()
81
82          print 'File read in complete'
```

```
        f.close()


    def __init__(self):
        self.data = []
```

Listing A.3: power.py: Power Class

```
#Power System Tools

import random
import os
from power import *

def makeInputFile(fo,scenarios, stages, outcomes):  #Make input file for msip
    fout = open(fo,"w")
    fout.write("name=sim\n\n")
    fout.write("lineoutage=yes\n")

    fout.write("Problem and Solve Method\n")
    fout.write("solvemethod=cplex\n")
    fout.write("problemtype=sim\n")

    fout.write("Power Grid\n")
    fout.write("gridfile=input/grid2.gr\n\n")

    fout.write("Line Failure Distribution\n")
    fout.write("p=.5\n")
    fout.write("L=.98\n\n")

    fout.write("Scenario Tree\n")
    fout.write("stages="+str(stages)+"\n")
    fout.write("outcomesV=")
    print outcomes
    for out in outcomes:
        fout.write(str(out)+" ")
    fout.write("\n")
    fout.write("scenarios="+str(scenarios)+"\n\n")

    fout.write("Outages\n")
    fout.write("a=12 14 34 11\n")
    fout.write("b=12 2 21\n")
    fout.write("c=82 5 25 34\n")
    fout.write("d=13 24\n\n")

    i = random.randint(12045,1205015)
    fout.write("Random Numbers\n")
    fout.write("seed="+str(i)+"\n\n")

    fout.close()
    print "Wrote to datafile "+ fo


def count(fi, fo, scale):  # "Usage: count.py <line outage file> <output file> <scale>"
    x={'0':0}
    for i in range(186):
        x[str(i)]=0
    fn = open(fo,'w')
    fn.write('line outages\n')
    bigdog=0.1
    f = open(fi,'r')

    lines = f.readline()

    [n,p,v] = lines.partition(' ')
    x[n]=1+x.get(n,0)

    while (v):
```

```
61          [n,p,v] = v.partition(' ')
62          x[n]=1+x.get(n,0)
63          if (bigdog<x[n]):
64              bigdog=x[n]
65
66      print x
67      sorted_x = sorted(x.items(), key=lambda x: x[1])
68      sorted_x.reverse()
69      print sorted_x
70
71      bigdog=float(scale)
72
73      for i in sorted_x:
74          fn.write(i[0]+" "+str(i[1])+"\n")
75
76      print "BIG: "+str(bigdog)
77      fn.close()
78      f.close()
79
80  def dumb(fi, fout, MW, k):   # Read in line outage file and output capacity file
81
82      fin = open(fi,'r')
83      l = fin.readline()
84      print l
85      lines = fin.readlines()
86
87      x={'0':0}
88      for i in range(186):
89          x[str(i)]=0
90
91      rank=['0']
92
93      for l in lines:
94          [ln,p,e] = l.partition(" ")
95          [hz,p,e] = e.partition("\n")
96          x[ln]=hz
97          rank.append(ln)
98          print ln,hz
99
100     rank.pop(0)
101
102     MWb = MW * 29600/100
103     print "Per Line: ",MWb/k,", Lines: ",k
104
105     rank = rank[0:k]
106     print rank
107
108     f = open(fout,'w')
109     f.write("Design Capacity -individual\n")
110     for line in rank:
111         f.write(str(line))
112         f.write(' ')
113         f.write(str(MWb/k))
114         f.write('\n')
115     f.close()
116
117 def loadshedanalysis( fi, fo, nom ):  #Read in <.dem> file and preform analysis and output
        to <.lsa>
118     f = open(fi,'r')
119     for line in f:
120         len(line)
121     ls = line.split("  ")
122     size = len(ls) -1
123     ls = ls[0:size]
124     ls.sort()
125 #     print "\nDemand Served"
126 #     print ls
127     total=0
```

```
    ls1 = []
    for item in ls:
        ls1.append(nom-float(item))
        total = total + nom - float(item)
    sq=0
    avg=total/size
    i = int(.05*size)
    count=0
    c2=0
    t2=0
    for item in ls1:
        count=count+1
        if(count<=i):
            c2=c2+1
            t2=t2+item
        sq = (item-avg)*(item-avg)/size + sq
    se=sq**.5/size**.5
    avg=total/size

    fout = open( fo, 'w')
    fout.write( "Samples: " +str(size) )
    fout.write( "\nAverage: " + str(avg)+ "  CI 95% [ " + str(avg-se*1.96) + ", " + str(avg+
    se*1.96) +" ]" )
    fout.write( "\nStandard Deviation: " + str(sq**.5) )
    fout.write( "\nStandard Error: " + str(se) )
    fout.write( "\nMin: " + str(min(ls1)) )
    fout.write( "\nMax: " + str(max(ls1)) )
    fout.write( "\n95% V@R: " + str(ls1[i]) )
    fout.write( "\n95% CV@R: " + str((t2/c2)) )

    f.close()
    fout.close()

def readlsa(fi):   #  Read load shed analysis file and return data
    data = [ 0, 0, 0, 0, 0, 0, 0, 0 ]
    #      [ Samples, Expect, St Dev, St Err, Min, Max, Var, CVar ]
    f = open(fi,'r')

    samples = f.readline()
    [a,samples] = samples.split(' ')
    [samples, a] = samples.split('\n')
    data[0] = samples

    mean = f.readline()
    mean =  mean.split(' ')[1]
    data[1] = mean

    stdv = f.readline()
    stdv = stdv.split(' ')[2].split('\n')[0]
    data[2] = stdv

    sterr = f.readline()
    sterr = sterr.split(' ')[2].split('\n')[0]
    data[3] = sterr

    mn = f.readline()
    mn = mn.split(' ')[1].split('\n')[0]
    data[4] = mn

    mx = f.readline()
    mx = mx.split(' ')[1].split('\n')[0]
    data[5] = mx

    var = f.readline()
    var = var.split(' ')[2].split('\n')[0]
    data[6] = var

    cvar = f.readline()
```

```
    cvar = cvar.split(' ')[2].split('\n')[0]
    data[7] = cvar
    f.close()

    return data

def readlao(fi):   #  Read load shed analysis file and return data
    data = { '-1':0 }
    #      [ line:outages ]
    f = open(fi,'r')

    lines = f.readlines()


    for line in lines:
        if 'line' in line:
            testr=1
        else:
            [a, b] = line.split(' ')
            data[int(a)] = int(b.split('\n')[0])

    data.pop('-1')
    return data

def readcap(fi):   #  Read load shed analysis file and return data
    data = { '-1':0 }
    #      [ line:capacity addition ]
    f = open(fi,'r')
    f.readline()
    lines = f.readlines()


    for line in lines:
        data[ line.split(" ")[0] ] = line.split(" ")[1].split('\n')[0]


    data.pop('-1')

    return data


def stepLines( lines, ts, numPoints):
    budget = 500
    used = 0
    norm = (numPoints-1)/3
    for t in ts.keys():
        print t,ts[t]
        used = float(used) + float(ts[t])

    a = Grid()
    a.readData( 'grid2.gr' )

    flist = open('allocate.lines','w')
    linenumber=1
    for line in lines:
        a.printBranch( line )
        Limit = a.getBranchLimit( line )
        print Limit
        if float(Limit) > float(budget)/3:
            Limit = float(budget)/3
        print Limit
        cs = [i*float(Limit)/norm for i in range(numPoints)]
        cs_r = [ round(elem, 2) for elem in cs ]
        print cs_r

        basename='n'+str(linenumber)
        linenumber=linenumber+1
        for i in range(numPoints):
```

```
263             name = basename+'_'+str(i)
264             flist.write(name+', ')
265             if not os.path.exists( name ):
266                 os.mkdir( name )
267             capfile = name+'/additions.cap'
268             fout = open(capfile,'w')
269             fout.write('Design Capacity -individual\n')
270 
271             newbudget = budget - cs_r[i]
272             w = newbudget/budget
273             print w,newbudget, used, budget
274             for t in ts.keys():
275                 final = float(ts[t])*w
276                 fout.write(t+' '+str(final)+'\n')
277 
278             fout.write(line+' '+str(cs_r[i])+'\n')
279 
280             fout.close()
281 
282 def stepGroups(groups, ts):
283     budget = 500
284     used = 0
285     for t in ts.keys():
286         print t,ts[t]
287         used = float(used) + float(ts[t])
288 
289     a = Grid()
290     a.readData( 'grid2.gr' )
291 
292     flist = open('step.lines','w')
293     linenumber=1
294     for group in groups:
295         print group
296 
297         gs = group.split(", ")
298         groupcap = 0
299         for line in gs:
300             a.printBranch( line )
301             Limit = a.getBranchLimit( line )
302             print Limit
303             groupcap = groupcap + float(Limit)
304 
305 #         groupweights = [ float(a.getBranchLimit( line ))/groupcap for line in gs ]
306         groupweights = { '-1':0 }
307         for line in gs:
308             groupweights[ line ] = float(a.getBranchLimit( line ))/groupcap
309         groupweights.pop('-1')
310         print groupweights
311 
312         w1 = [float(i)/30 for i in range(31)]
313         w2 = [1-float(i)/30 for i in range(31)]
314 
315         ef1 = [w*budget for w in w1]
316         ef2 = [w*budget for w in w2]
317 
318         print ef1
319         print ef2
320 
321         basename='n'+str(linenumber)
322         linenumber=linenumber+1
323         for i in range(31):
324             name = basename+'_'+str(i)
325             flist.write(name+', ')
326             if not os.path.exists( name ):
327                 os.mkdir( name )
328             capfile = name+'/additions.cap'
329             fout = open(capfile,'w')
330             fout.write('Design Capacity -individual\n')
```

```
331
332             newbudget = ef2[i]
333             w = newbudget/used
334             for t in ts.keys():
335                 final = float(ts[t])*w
336                 fout.write(t+' '+str(final)+'\n')
337
338             for line in gs:
339                 final = ef1[i]*groupweights[line]
340                 fout.write(line+' '+str(final)+'\n')
341             fout.close()
342
343 """
344         a.printBranch( line )
345         Limit = a.getBranchLimit( line )
346         print Limit
347         cs = [i*float(Limit)/10 for i in range(31)]
348         cs_r = [ round(elem, 2) for elem in cs ]
349         print cs_r
350
351         for i in range(31):
352
353
354
355             w = newbudget/budget
356             print w,newbudget, used, budget
357             for t in ts.keys():
358                 final = float(ts[t])*w
359                 fout.write(t+' '+str(final)+'\n')
360
361             fout.write(line+' '+str(cs_r[i])+'\n')
362
363             fout.close()
364 """
```

Listing A.4: tools.py: Common Functions

```
1 #!/progs/gurobi/linux64/bin/python2.7
2
3 import sys
4 import os
5 import shutil
6 from cap import *
7 from tools import *
8 from power import *
9
10 if __name__=="__main__":
11     if len(sys.argv) > 2:
12         print sys.argv
13
14         outputLocal = '../mostOutages'
15
16         if sys.argv[1] =='-step':
17             pointin = sys.argv[2]
18             linesin = sys.argv[3]
19             print 'Searching for better points'
20             print 'Open '+pointin+' as initial point'
21             print 'and search lines from '+linesin
22
23             ts = readcap(pointin)
24
25             if '.lines' in linesin:
26                 fl = open( linesin, 'r')
27                 lines = fl.readline().split(", ")
28                 print lines
29                 if '' in lines:
30                     lines.remove('')
31                 stepLines( lines, ts, 31 )
```

```
32                  fl.close()
33              if '.group' in linesin:
34                  fl = open( linesin, 'r')
35                  groups = fl.readline().split("/ ")
36                  print groups
37                  stepGroups( groups, ts )
38                  fl.close
39
40          else:
41              if len(sys.argv) > 3:
42                  outputLocal = sys.argv[3]
43                  iteration = sys.argv[3].split('step')[1]
44                  print 'Iteration: ',iteration
45
46          if sys.argv[1] == '-point':
47              print "Find good point and output to point.cap"
48              print "Find lines to search from new point and output to search.lines"
49              test = Cap( sys.argv[2], outputLocal)
50              test.params(.1,.2)
51  #            ss=test.getStableSet()
52              budget=500
53              point=test.getGreedyPoint()
54              print point
55              name = outputLocal+point[0]+'_'+point[1]+'/add.cap'
56              ts = readcap( name )
57              f = open('point.cap','w')
58              f.write('Design Capacity -individual')
59
60              for t in ts.keys():
61                  f.write('\n' + t + ' ' + ts[t])
62              f.close()
63
64              namelao = outputLocal+point[0]+'_'+point[1]+'/want.lao'
65              #Find lines to search
66              problines = readlao( namelao )
67
68              flao = open('search.lines','w')
69              print  problines
70              for line in problines.keys():
71                  Threshold = 2500
72                  if problines[line] > Threshold:
73                      flao.write(str(line)+', ')
74                      print line
75              flao.close()
76
77              stub = outputLocal + point[0]+'_'+point[1]
78              shutil.copy( namelao, 'point.lao')
79              shutil.copy( stub+'/want.lsa','point.lsa')
80
81          if sys.argv[1] == '-c':
82              test = Cap( sys.argv[2], outputLocal)
83              test.params(.1,.2)
84              ss=test.getStableSet()
85              budget=500
86              picks=test.getGreedySet(budget)
87              print picks
88              test.createFiles(picks,budget)
89
90          if sys.argv[1] == '-picks':
91              test = Cap( sys.argv[2], outputLocal )
92              test.params(.1,.2)
93              ss=test.getStableSet()
94              budget=500
95              picks=test.getGreedySet(budget)
96              print picks
97              f = open('allocate.picks','w')
98              for p in picks:
99                  f.write(str(p)+' ')
```

```
100             f.close()
101
102 #CONDOR SUBMIT first allocation step
103         if sys.argv[1] == '-clean':
104             fin = open( sys.argv[2] , 'r' )
105             line = fin.readline()
106             print line
107             names = [ n for n in line.split(", ") ]
108             names.remove( '' )
109
110             print names
111             for name in names:
112 #                print name
113                 if os.path.exists( name ):
114                     shutil.rmtree( name )
115
116         if sys.argv[1] == '-p':
117             fin = open( sys.argv[2] , 'r' )
118             line = fin.readline()
119             print line
120             names = [ n for n in line.split(", ") ]
121             names.remove( '' )
122             stubs = set([ n.split('_')[0] for n in names ])
123             print stubs
124
125             for s in stubs:
126                 f = open( s+'.dat','w' )
127                 f.write('run smp ex sd se min max var cvar\n')
128                 for name in names:
129                     if s+'_' in name:
130 #                        print outputLocal+'/'+name
131                         ts = readlsa(outputLocal+'/'+name+'/want.lsa')
132                         f.write(name.split('_')[1]+'\t')
133                         for t in ts:
134                             f.write(t+'\t')
135                         f.write('\n')
136                 f.close()
137
138                 fl = open(s+'.ldat','w')
139                 fls = open(s+'.lsdat','w')
140                 fl.write('name ')
141                 for i in range(185):  # FOR GRID 2
142                     fl.write( str(i) +' ')
143                 fl.write('\n')
144                 for name in names:
145                     if s+'_' in name:
146                         ts = readlao(outputLocal+'/'+name+'/want.lao')
147                         fl.write(name.split('_')[1]+'\t')
148                         fls.write(name.split('_')[1]+'\t')
149                         for key in sorted(ts.iterkeys()):
150                             fl.write(str(ts[key])+'\t')
151
152                         for key in sorted( ts.iteritems(), key=operator.itemgetter(1),
    reverse=True ):
153                             fls.write(str(key[0])+'\t')
154                         fl.write('\n')
155                         fls.write('\n')
156                 fl.close()
157                 fls.close()
158
159                 fc = open(s+'.cdat','w')
160                 fc.write('name')
161
162                 beginning = True
163                 for name in names:
164                     if s+'_' in name:
165                         ts = readcap(outputLocal+'/'+name+'/add.cap')
166
```

```
                        if beginning:
                            for t in ts:
                                fc.write('\t'+t)
                            fc.write('\n')
                            beginning = False

                        fc.write( name.split('_')[1])
                        for t in ts.keys():
                            fc.write('\t'+ts[t])
                        fc.write('\n')
                fc.close()
            print stubs

    else:
        print "Usage:  allocate <operation> <line in files>"
        print "                -c (create files)"
        print "                -p (process files)"
        print "                -clean (clean files)"
        print "                -step <point.cap> <search.lines>"
```

Listing A.5: allocate.py: Direct Search Pattern

```
#!/progs/gurobi/linux64/bin/python2.7

import sys
import os
import operator
from tools import *
from power import *

def makeCapFile(line,cap1):
    f = open( 'LIST_'+line, 'w' )

    for c1 in cap1:
        name = line+'.'+str(c1)
        f.write(name+' ')
        if not os.path.exists( name ):
            os.mkdir( name )
        capfile = name +'/additions.cap'
        fout = open(capfile,'w')

        fout.write('Design Capacity -individual\n')
        fout.write(line+' '+str(c1))

    f.close()
    fout.close()

def outputData(line,cap1):
    inputlocation = '../step0/'
    f = open('1d_'+line+'.dat','w')
    f.write('run smp ex sd se min max var cvar\n')
    for c1 in cap1:
        name = line+'.'+str(c1)
        ts = readlsa(inputlocation + name+'/want.lsa')
        f.write(str(c1)+'\t')
        for t in ts:
            f.write(t+'\t')
        f.write('\n')
    f.close()

    fl = open('1d_'+line+'.ldat','w')
    fls = open('1d_'+line+'.lsdat','w')
    fl.write('name ')
    for i in range(185):  # FOR GRID 2
        fl.write( str(i) +' ')
    fl.write('\n')
    for c1 in cap1:
        name = line+'.'+str(c1)
```

```
47          ts = readlao(inputlocation+name+'/want.lao')
48          fl.write(str(c1)+'\t')
49          fls.write(str(c1)+'\t')
50
51          for key in sorted(ts.iterkeys()):
52              fl.write(str(ts[key])+'\t')
53
54          for key in sorted( ts.iteritems(), key=operator.itemgetter(1), reverse=True ):
55              fls.write(str(key[0])+'\t')
56
57          fl.write('\n')
58          fls.write('\n')
59
60      fl.close()
61      fls.close()
62
63
64      fc = open('1d_'+line+'.cdat','w')
65      fc.write('name')
66      beginning = True
67
68      for c1 in cap1:
69          name = line+'.'+str(c1)
70          ts = readcap(inputlocation+name+'/add.cap')
71          fc.write(str(c1)+'\t')
72
73          if beginning:
74              for t in ts:
75                  fc.write('\t'+t)
76              fc.write('\n')
77              beginning = False
78
79          fc.write(str(c1))
80          for t in ts.keys():
81              fc.write('\t'+ts[t])
82          fc.write('\n')
83      fc.close()
84
85  def cleanFiles(line,cap):
86      for c in cap:
87          name = line+'.'+str(c)
88          if os.path.exists(name):
89              os.remove(name+'/additions.cap')
90              os.rmdir(name)
91
92  if __name__=="__main__":
93      if len(sys.argv) > 2:
94
95          a = Grid()
96          a.readData( sys.argv[2] )
97
98  #        lines = [ '31', '48', '84', '85', '15', '97', '77', '98', '37','106', '181', '113',
        '141', '137' ]
99          if len(sys.argv) > 3:
100             fin = open( sys.argv[3] , 'r')
101             linest = fin.readline()
102             check = linest.split('\t')
103         else:
104             fin = open( "LINES" , 'r' )
105             linest = fin.readline()
106             check = linest.split(' ')
107
108         print linest
109         print check
110         fin.close()
111
112         subset = check[0 : 104 ]
113         fsubset = open('subset.lines','w')
```

```
        for line in subset:
            fsubset.write(line + ", " )
        fsubset.close()

        for line in subset:
            a.printBranch( line )
            Limit = a.getBranchLimit( line )
            print Limit

            cs = [i*float(Limit)/10 for i in range(31)]
            cs_r = [ round(elem, 2) for elem in cs ]
            print cs_r

            if sys.argv[1] == '-p':
                print "Process"
                outputData(line,cs_r)
            if sys.argv[1] == '-c':
                print "Make condor submit structure for dag"
                makeCapFile(line,cs_r)
            if sys.argv[1] == '-clean':
                print "Clean condor submit structure"
                cleanFiles(line,cs_r)

    else:
        print "Usage: consub_1d.py <operation> <gridfile>  (Interface with condor)"
        print "            -p process files"
        print "            -c make condor submit"
        print "            -l count line outages"
```

Listing A.6: consub.py: Build HTCondor Submit Structure

```
#!./python273/bin/python

import sys
from tools import *

if __name__=="__main__":
    if len(sys.argv) > 2:
        nominaldemand = 3668
        print "Analyzing "+argv[1]+" and outputing results to "+argv[2] )
        loadshedanalysis( argv[1], argv[2], nominaldemand )

    else:
        print "Usage: loadShed.py < ???.dem > < ???.lsa >  (Statistical analysis of load
    shed data)"
```

Listing A.7: loadshed.py: Take raw load shed and summarize

```
#!./python273/bin/python

import sys
from tools import *


if __name__=="__main__":

    if len(sys.argv) >2:
        "Processing "+argv[1]+ " into " +argv[2]
        trials = 10000
        count( argv[1], argv[2], trials )

    else:
        print "Usage: countLines.py < ???.lin > < ???.lao>  (Counts line outages)"
```

Listing A.8: countLines.py: Take raw line out info and summarize

```
line,outages
14,14557 12,14557 11,14557 34,14557 141,12774 140,12533 112,12226 137,12002 37,11639
    182,11629 15,11492 64,10682 31,10434 118,10251 106,9946 133,9912 42,9683 8,9438 6,9313
    63,8946 38,8526 139,7934 122,7381 154,7071 158,6802 98,6689 97,6665 90,6554 77,6473
    142,6261 89,6260 91,6201 54,6179 181,5863 126,5822 125,5521 88,4902 121,4857 138,4851
    86,4723 74,4630 75,4578 100,4111 148,4059 41,3574 107,3421 120,3397 185,3373 178,3189
    78,3058 156,3047 57,2992 61,2743 76,2652 52,2643 164,2629 87,2615 101,2568 45,2527
    136,2464 23,2304 173,2165 175,2048 62,2047 105,2018 159,1975 40,1955 30,1841 163,1821
    99,1803 72,1594 176,1569 96,1568 92,1542 93,1535 27,1529 165,1526 157,1525 81,1491
    67,1462 177,1349 128,1192 103,1152 94,1122 50,1004 183,983 56,978 115,871 166,839
    123,791 71,756 22,726 119,719 153,695 151,685 147,673 135,666 83,604 73,585 168,584
    172,575 49,568 167,509 149,502 162,397 143,396 80,385 161,334 85,333 113,305 59,285
    20,275 127,227 104,222 114,212 55,212 35,192 48,189 170,167 117,156 24,152 171,134
    60,108 155,96 66,96 25,93 65,86 144,65 16,61 47,44 53,44 69,41 169,40 95,37 84,35 134,29
     146,26 19,22 79,16 131,16 70,14 116,10 43,9 129,7 18,5 109,5 44,3 110,3 51,3 152,2 7,1
    145,0 9,0 5,0 1,0 46,0 150,0 17,0 13,0 10,0 184,0 2,0 180,0 174,0 68,0 36,0 32,0 33,0
    39,0 102,0 108,0 3,0 82,0 111,0 179,0 58,0 124,0 4,0 0,0 29,0 28,0 160,0 21,0 26,0 130,0
     132,0
```

Listing A.9: point.lao: Line outage file from chosen design

A.3 Joint Chance Constraint Model

This code is publicly available on GitHub at `https://github.com/eanderson4/pow-opt`

A.4 Joint Chance Constraint with OPA Weighting

This code is publicly available on GitHub at `https://github.com/eanderson4/opt-opa`