Copyright Statement

The digital copy of this thesis is protected by the Copyright Act 1994 (New Zealand).

This thesis may be consulted by you, provided you comply with the provisions of the Act and the following conditions of use:

- Any use you make of these documents or images must be for research or private study purposes only, and you may not make them available to any other person.
- Authors control the copyright of their thesis. You will recognize the author's right to be identified as the author of this thesis, and due acknowledgement will be made to the author where appropriate.
- You will obtain the author's permission before publishing any material from their thesis.

General copyright and disclaimer

In addition to the above conditions, authors give their consent for the digital copy of their work to be used subject to the conditions specified on the Library Thesis Consent Form and Deposit Licence.
Abstract

This thesis examines industrial demand-side management over a co-optimized energy and reserve market from the perspective of a strategic consumer with flexibility in demand and ability to offer interruptible load. Industrial demand response is the participation of large consumers in electricity markets, through active management of their consumption in response to prices. Demand response not only enables large consumers of electricity to reduce their energy costs, but it also improves the reliability of the system, facilitates increased capacity of distributed energy resources within the market, and may enable deferrals in generation and transmission investments. Another important aspect of industrial demand response is its contribution to maintaining grid stability through offering ancillary services. For instance, major consumers with demand flexibility may offer interruptible load reserve, which enables them to earn revenue through their ability to shed load thus protecting the system against potential outages.

What differentiates industrial demand-side participation from other forms of demand response is that a large manufacturer’s decision on its consumption level can affect the energy and reserve prices. Therefore, we model price-making major consumers who anticipate the impacts of their actions on the market. Through a bi-level optimization model, we study the strategic behavior of large consumers in both energy and reserve markets under uncertainty. This optimization problem maximizes the expected utility of the major consumer, yielding admissible energy bid stacks and reserve offer stacks for each trading period. In order to apply this to real-world problems, our model is reformulated to a mixed-integer program, for which we offer tailor-made solution methods.

In addition to presenting utility-driven co-optimization of energy and reserve, we propose stochastic cost-minimizing demand scheduling models, where finding the optimal load level for each trading period depends on the consumption in other time periods. We present a unique and novel stochastic multistage optimization method to compute optimal strategies using Lagrangian decomposition methods. This model is developed for major manufacturers who are required to achieve a total production level over a short- or long-term time horizon and can shift consumption of electricity by utilizing flexibility in their production schedules.

Finally, we demonstrate implementation of our single- and multi-stage models for a large industrial consumer of energy in New Zealand. Through utilizing our proposed solution methodologies, we simulate our policies with historical data and compare the results with the cost of policies that are used in practice.
For my family and friends
I would like to thank my supervisors Golbon Zakeri and Anthony Downward. I deeply appreciate your patience, support and guidance throughout this project. I would also like to thank Prof. Miguel F. Anjos and Prof. Michael Ferris for their collaboration. I am grateful to you for all the time you dedicated to this work. Furthermore, I would like to thank the Electric Power Optimization Center (EPOC), University of Auckland and the MBIE grant who supported me financially during this project.

I would like to thank my parents, my sister and my brother for all their unconditional love and support. I am grateful to you all for giving me the opportunity to be ambitious, and pursue my dreams.

To Ahmad, Amir, Bahareh, Farzaneh, Maryam, Parisa and Rasoul, I truly appreciate your friendship and emotional support throughout my PhD. I would like to thank my friend Hasti, whom I have always gone to for advice in the past eight years. I also would like to thank my friend Maria for sharing early morning runs with me, which certainly helped me keep sane throughout this process. And to Derek, thank you for always being there for me in all ups and downs, and being patient with my unusual PhD life schedule.

And lastly, to my dearest friends of many years, Mina, Parinaz, Pegah, Saeedeh and Simin, who supported me throughout my journey. You encouraged me, believed in me and supported all my decisions, without you none of this would even be possible.
## Contents

Abstract ii

Acknowledgements iv

List of Figures viii

List of Tables x

Abbreviations xi

Terms xiii

Preface 1

I Background 4

1 Electricity Markets 5

1.1 Wholesale Electricity Market 6

1.2 Forward Markets 9

1.3 Reserve Markets 10

1.4 Demand Response 11

1.5 New Zealand Electricity Market 13

1.5.1 Wholesale Energy Market 14

1.5.2 Reserve Procurement 17

1.5.3 Demand Response 20

2 Mathematical Background Review 25

2.1 Motivation and Structure 25

2.2 Auctions 26

2.3 Equilibrium Constraints 27

2.3.1 Competitive Equilibrium 28

2.3.2 Bi-level Optimization 29

2.4 Dynamic Programming 30

2.5 Lagrangian Methods 31
## Contents

### II Single-Stage Co-Optimization of Energy and ILR

#### 3 Price-taking Agents

3.1 Introduction .................................................. 35
3.2 Co-optimized Energy and Reserve Optimal Power Flow Problem .... 37
3.3 Competitive Equilibrium ..................................... 40
3.4 Multi-item Auctions .......................................... 43
   3.4.1 A major consumer’s electricity and ILR joint value-cost function 44
3.5 Conclusions ................................................. 50

#### 4 Price-making Major Consumer

4.1 Introduction .................................................. 53
4.2 Bi-level Optimization Formulation .......................... 57
4.3 Standard MIP Reformulation ................................ 58
   4.3.1 Linearizing the Objective Function .................... 60
4.4 Strategic Co-optimization Effects .......................... 61
4.5 Strategic Consumer with Contracts for Differences .......... 64
4.6 Conclusions ................................................. 67

#### 5 Stochastic Price-making Major Consumer

5.1 Introduction .................................................. 70
5.2 Stochastic Co-optimization Formulation .................... 72
   5.2.1 Admissible Bids and Offers ............................. 75
5.3 Stochastic Co-optimization Effects ........................ 78
   5.3.1 3-Node Network ........................................ 79
   5.3.2 Co-optimized Bid and ILR Curve Example ............. 80
   5.3.3 BOOMER-Consumer Comparison ....................... 82
   5.3.4 Stochastic Co-optimization Solution Time ........... 84
5.4 Reformulation Methods ...................................... 85
   5.4.1 Piece-wise Linear Reformulation ....................... 86
   5.4.2 Improved MIP Method ................................ 88
   5.4.3 Disjunctive Reformulation ............................. 88
   5.4.4 Alternative Reformulations ............................ 90
   5.4.5 Bi-parametric Sensitivity Analysis Reformulation ... 91
      5.4.5.1 General Format Algorithm ....................... 91
      5.4.5.2 Bi-parametric Reformulation Stochastic Application 93
   5.4.6 Reformulation Methods Comparison .................... 96
5.5 Case Study ................................................. 96
   5.5.1 Scenario Selection Examples .......................... 98
   5.5.2 Full-Scale Model Solution Time ....................... 99
   5.5.3 Stochastic Strategic Policy Simulation ............... 100
   5.5.4 Policy Performance Comparison ....................... 101
5.6 Conclusions ................................................. 104

### III Multi-stage Co-Optimization of Energy and ILR

#### 6 Strategic Consumption over a Finite Time Horizon

---

---
6.1 Introduction .................................................. 107
6.2 Problem Structure ........................................... 114
6.3 Dynamic Programming Solution Methods .................. 115
  6.3.1 Standard DP ........................................... 115
6.4 Lagrangian Decomposition .................................. 117
  6.4.1 Price Taker Consumer ................................. 117
  6.4.2 Price Maker Consumer ................................. 118
6.5 Utility Consumption Curves ................................. 119
6.6 Heuristics .................................................... 121
6.7 Bounds ....................................................... 122
6.8 U-C Algorithm ............................................. 123
6.9 Multistage Co-optimization Example ...................... 124
  6.9.1 Multistage Co-optimization with CfD .................. 127
6.10 Conclusions ............................................... 129

7 Stochastic Strategic Consumption over a Finite Time Horizon 131
  7.1 Introduction ................................................ 131
  7.2 Problem Structure ......................................... 135
  7.3 Stochastic Decomposition ................................. 136
    7.3.1 Price Taker Consumer ............................... 136
    7.3.2 Price Maker Consumer ............................... 138
  7.4 Expected Utility Consumption Curves ..................... 139
    7.4.1 Stage-wise dependency ................................ 141
    7.4.2 Heuristics ........................................... 142
    7.4.3 Stochastic U-C Policy Implementation ............... 143
  7.5 Modelling Uncertainty ................................... 144
    7.5.1 Scenario Clusters .................................. 144
    7.5.2 Scenario Realizations ................................ 145
  7.6 Case Study ............................................... 148
    7.6.1 Scenario Clusters for NZAS ......................... 148
    7.6.2 Expected U-C Curves for NZAS ...................... 150
    7.6.3 Policy Simulation Algorithm ......................... 151
    7.6.4 Daily Scheduling .................................... 152
    7.6.5 Weekly Scheduling ................................... 154
    7.6.6 Monthly Scheduling .................................. 156
  7.7 Conclusions ............................................... 158

8 Concluding Remarks .......................................... 161

IV Appendices .................................................... 164
  A Gurobipy: Bi-parametric Sensitivity Analysis Region Construction 165
  B Gurobipy: Construction of Optimal Monotone Bid and Offer Curves 187

Bibliography ...................................................... 194
# List of Figures

1.1 Energy offer stack ............................................. 15  
1.2 Wholesale electricity prices ............................... 16  
1.3 SIR prices in 2017 ...................................... 18  
1.4 Yearly FIR prices ........................................ 18  
1.5 Aggregated supply function .............................. 21  
1.6 Alleviating price spikes at Tiwai point in SI with change in demand .................. 21  
1.7 Alleviating price spikes at Glenbrook point in NI with change in demand ....... 22  
1.8 Demand bid stack ....................................... 22  

3.1 Inverse bathtub constraints ............................. 38  
3.2 Generation offer stack .................................. 45  
3.3 Generator’s reserve offer stack ......................... 46  
3.4 Demand bid stack ....................................... 47  
3.5 Optimal cleared energy quantities for consumer’s offer and bid tranches in [J-EROPF] .............................................................. 48  
3.6 Optimal cleared energy quantities for consumer’s offer and bid tranches in [S-EROPF] .............................................................. 49  
3.7 Welfare under various demand realizations. ......... 50  

4.1 The single-node network .................................. 62  
4.2 Profit maximization objective function ............... 63  
4.3 Strategic consumer decision variables ................ 63  
4.4 Electricity prices ......................................... 64  
4.5 Reserve prices ............................................ 64  
4.6 CfD model’s profit maximization objective function ......... 66  
4.7 CfD model’s strategic consumer decision variables ............ 66  
4.8 CfD model’s electricity prices ........................... 67  
4.9 CfD model’s reserve prices .............................. 67  

5.1 Optimal wait-and-see bids. ............................. 75  
5.2 Optimal monotone consumer bid ......................... 75  
5.3 Monotone demand bid with \( m = 3 \) tranches .......... 76  
5.4 The 4-node network ...................................... 79  
5.5 Stochastic co-optimization effect ....................... 81  
5.6 Optimal wait-and-see bids. ............................. 83  
5.7 Energy price regions .................................... 94  
5.8 Reserve price regions ................................... 95  
5.9 Sample set comparison .................................. 98
## List of Figures

5.10 Similar scenarios .................................................. 98  
5.11 Optimal energy bid stack - winter peak .......................... 101  
5.12 Optimal stack policy out-of-sample simulation algorithm .......... 101  
5.13 Winter peak profit vs value of electricity $u$ ....................... 103  
6.1 U-C for one TP ........................................................ 119  
6.2 Aggregate U-C ....................................................... 119  
6.3 U-C curve demonstration for [HEU] ................................. 121  
6.4 Cost curve ............................................................ 123  
6.5 Cost curve - bounded ............................................... 123  
6.6 The 3-node network. .................................................. 125  
6.7 Aggregated U-C curves for v1 and v2 .............................. 126  
6.8 Energy and reserve prices in v1 and v2 ............................ 127  
7.1 Nodes in the tree $n \in \mathcal{N}_t$ ...................................... 136  
7.2 Paths of the tree $m \in \mathcal{N}_T$ ...................................... 136  
7.3 Vertical expectation step of constructing aggregated expected U-C curves 140  
7.4 Horizontal aggregation step of constructing aggregated expected U-C curves 140  
7.5 Wait-and-See ......................................................... 146  
7.6 Here-and-Now ........................................................ 146  
7.7 Hybrid ................................................................. 147  
7.8 Optimal stochastic policy out-of-sample simulation algorithm ........ 151  
7.9 Policy comparison .................................................. 153  
7.10 Stage-wise dependent/independent policy comparison ................ 154  
7.11 Weekly and daily policy comparison .............................. 156
## List of Tables

1.1 Generators offers ................................................. 19
3.1 Separate versus joint cost function results .......................... 49
4.1 Parameter values of the single node example ......................... 62
4.2 Energy and reserve offer stack data. ................................ 63
5.1 Energy and reserve offer stack data. ................................ 79
5.2 Inelastic demand and reserve levels ................................. 80
5.3 Co-optimized vs energy-only bidding comparison .................. 81
5.4 BOOMER optimal reserve stack ..................................... 83
5.5 Profit comparison .................................................. 83
5.6 Non-monotone and monotone solution-time comparison .............. 85
5.7 Reformulation comparisons ........................................ 97
5.8 Out-of-sample performance comparison ............................. 98
5.9 Profit vs policy ..................................................... 99
5.10 Average bound gaps (%) versus number of scenarios. .............. 100
5.11 Number of integers versus number of scenarios. ................... 100
5.12 Policy performance- winter peak .................................. 102
5.13 Winter policy performance ......................................... 102
5.14 Summer policy performance (with $u = 90$/MWh) .................. 104
6.1 Energy and reserve offer stack data. ................................ 125
6.2 Inelastic demand and reserve levels ................................ 125
6.3 Consumption levels for normal U-C policy vs heuristic policy ...... 126
6.4 Total cost comparison .............................................. 127
6.5 Consumption levels for normal policy vs CfD policy ................. 128
7.1 Price ranges ......................................................... 145
7.2 Price sensitivity analysis ............................................ 149
7.3 Price range clusters ................................................ 149
7.4 Daily scheduling comparison ....................................... 152
7.5 Weekly scheduling comparison ..................................... 155
7.6 Weekly scheduling comparison ..................................... 155
7.7 Bisection search $u$ intervals for monthly scheduling .............. 157
7.8 Monthly scheduling comparison ..................................... 158
Abbreviations

AC Alternating Current
BOOMER Bid-offer Optimization Model for Electricity Resources
CfD Contracts for Differences
DP Dynamic Programming
EFV Expected Future Value
FTR Financial Transmission Right
GIP Grid Injection Point
GXP Grid Exit Point
HVDC High Voltage Direct Current
ILR Interruptible Load Reserve
ISO Independent System Operator
KKT Karush Kuhn Tucker
LP Linear Program
LMP Locational Marginal Pricing
MIP Mixed Integer Program
MOPEC Multiple Mathematical Programs with Equilibrium Constraints
MPEC Mathematical Program with Equilibrium Constraints
MW Mega Watt
MWh Mega Watt-hour
NI North Island of New Zealand
NZ New Zealand
NZAS New Zealand Aluminum Smelter
NZEM New Zealand Electricity Market
OPF Optimal Power Flow
PH Progressive Hedging
<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SDDP</td>
<td>Stochastic Dual Dynamic Programming</td>
</tr>
<tr>
<td>SDP</td>
<td>Stochastic Dynamic Programming</td>
</tr>
<tr>
<td>SPD</td>
<td>Scheduling Pricing and Dispatch</td>
</tr>
<tr>
<td>SI</td>
<td>South Island of New Zealand</td>
</tr>
<tr>
<td>TP</td>
<td>Time Period</td>
</tr>
<tr>
<td>UFE</td>
<td>Under Frequency Event</td>
</tr>
<tr>
<td>vSPD</td>
<td>Vectorized Scheduling Pricing and Dispatch</td>
</tr>
</tbody>
</table>
## Terms

<table>
<thead>
<tr>
<th>Term</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Ancillary Service</strong></td>
<td>A service that ensures the stability of the power system such as regulating, operating reserve and voltage support.</td>
</tr>
<tr>
<td><strong>Demand Response</strong></td>
<td>Participation of consumers in electricity market through price-responsive bidding strategies.</td>
</tr>
<tr>
<td><strong>Energy</strong></td>
<td>Used interchangeably with electricity throughout this thesis.</td>
</tr>
<tr>
<td><strong>Load</strong></td>
<td>Used interchangeably with demand throughout this thesis.</td>
</tr>
<tr>
<td><strong>Nodal Price</strong></td>
<td>A pricing scheme in which each node in the grid has a separate marginal price. This is also referred to as Locational Marginal Pricing (LMP)</td>
</tr>
<tr>
<td><strong>Optimal Power Flow</strong></td>
<td>The mathematical model that the system operator solves in order to obtain a feasible and optimal grid dispatch. This optimization problem is subject to supply, demand and network constraints.</td>
</tr>
<tr>
<td><strong>Uniform Pricing</strong></td>
<td>A form of auction where all the dispatched generators are paid the marginal clearing price for the trading period.</td>
</tr>
</tbody>
</table>
Preface

Over half of the world’s annual energy consumption is by the industrial sector, which also has the largest growth in demand in the power sector. Among all end-use forms of energy, electricity is the rising force, making up 40% of the rise in total consumption anticipated for 2040. This pattern is similar to the share of growth in the oil industry in the past 25 years [1]. In the industrialized countries, a high percentage of generated electricity is allocated to the industrial sector. For instance, around 70% of China’s national demand is consumed by industry.

In the past century, fossil fuels have been the main energy resource for on-going industrialization. However, since the 1990s, the increase in the demand together with depleting gas and coal resources have resulted in higher prices of energy, so that efficiency has become a necessity for many manufacturing industries. Furthermore, environmental concerns regarding the use of fossil fuels, such as the impact of the greenhouse gas emissions on climate change has incentivized investment in renewable energy resources for both suppliers of energy and the major consumers.

Utilizing renewable energy resources, however, comes with the disadvantage of uncertainty. The intermittent nature of wind and solar energy results in volatility in wholesale prices, that may discourage a risk-averse major consumer to participate in the spot market. On the other hand, the elasticity of demand that is a result of demand-side participation, reduces the risk of unforeseen disruptions in supply that may be caused by high penetration of intermittent energy resources. Demand response also alleviates large fluctuations in price. For instance, if there is a sufficient amount of price-sensitive load available at times of low renewable energy supply, expensive and polluting generators need not be activated, that would otherwise, lead to a spike in the price of electricity. In addition, using on-site generation for major consumers is a highly effective method to avoid purchase of energy at peak trading periods and prevent a further increase in spot prices.

Elasticity of demand can best be achieved through participation of large consumers in the wholesale market. However, fluctuations in dispatched quantities for manufacturers
who submit consumption bids instead of fixed demand values, have the risk of disruptions in their production process. Such risks can be minimized through implementing technologies to adapt flexible manufacturing with regard to energy consumption. In addition, in order to facilitate efficient manufacturing, the use of power can be optimized with respect to the electricity price. [2–4] are examples of production plans that employ technologies and optimization models in order to promote electricity price-sensitive manufacturing scheduling.

Among large manufacturers, there are some whose production processes depend heavily on electricity, such as aluminum smelters and steel manufacturers. The cost of energy for these consumers forms a high percentage of their total production cost. These large consumers have the incentive to participate in the electricity markets and exploit the competition that is enabled through deregulation of electricity markets. Since the load of these plants is high in volume, the effect of their decisions not only impacts on their own costs, it also alters the market spot prices.

In this thesis we explore demand-side participation for industrial consumers of energy and present stochastic policies that maximize the expected profit for the manufacturer. Unlike previous work on energy efficient manufacturing, our approach is focused on a price-maker consumer or in other words, endogenous price processes. Therefore, we are able to address the impacts of demand response in the market as well as the consumer’s cost reductions comprehensively. By utilizing novel methodologies for solving bi-level optimization programs and multistage stochastic programs, we present optimal consumption scheduling for long time-horizons. We also probe the participation of consumers in reserve markets, particularly in co-optimized markets, such as New Zealand.

### Thesis Outline

We start the thesis with Part I in which we lay out a background review of electricity markets and the mathematical models that we will use throughout this thesis. In Chapter 1, we discuss the properties of deregulated electricity markets, and how the generators and consumers interact in the market, in particular in the New Zealand market, which is a co-optimized electricity and reserve market. We continue with Chapter 2, in which we lay out a review of the mathematical foundations of our proposed modelling and solution approaches.

In part II we discuss large consumers’ co-optimization of energy and reserve. We start modeling the major consumer’s participation in the energy market using a price taker
approach, in Chapter 3. We set out a competitive equilibrium in which multiple generators and major consumers simultaneously maximize their profit. The analysis of the competitive equilibrium is followed by a more realistic and elaborate model in Chapter 4, where we consider the major consumer to be a price maker. The strategic consumer solves a bi-level optimization problem, with the upper-level objective of maximizing its profit, while the market clearing problem is embedded inside it as the lower-level problem. Solving a bi-level optimization problem is cumbersome, so we reformulate our bi-level optimization problem to a mixed-integer program (MIP).

In order to address uncertainty within the energy market, such as fluctuations in demand from other consumers, the major consumer should reflect here-and-now decisions in its bids and offers. In Chapter 5, we present a stochastic model, in which the strategic agent maximizes expected profit over a set of scenarios. Our aim is to present practical solution methods, as well as building a realistic model for demand response; accordingly, we examine several reformulation methods, in order to solve a full-scale model in a reasonable time frame. In addition, we outline the results of a real-world case study for the Tiwai Aluminum Smelter in New Zealand.

Large manufacturers need to comply with their contracts that determine their amount of production and the corresponding energy consumption through a time horizon. Accordingly, we dedicate Part III of this thesis to the multistage consumption planning. In Chapter 6 we co-optimize the consumption bid andInterruptible Load Reserve (ILR) offer of the strategic consumer over a finite time horizon.

Decision making for a long time horizon is subjected to higher risk of uncertainty than a short-term plan. In order to present policies that account for different possible realizations of future, in Chapter 7, we build a stochastic model in which we schedule consumption for a long-time horizon, for a given total consumption and over a set of scenarios. In order to solve this stochastic multi-stage problem, we introduce a heuristic stochastic dynamic programming approach, and simulate our proposed policies in a case study.
Co-Authorship Form

This form is to accompany the submission of any PhD that contains published or unpublished co-authored work. Please include one copy of this form for each co-authored work. Completed forms should be included in all copies of your thesis submitted for examination and library deposit (including digital deposit), following your thesis Acknowledgements. Co-authored works may be included in a thesis if the candidate has written all or the majority of the text and had their contribution confirmed by all co-authors as not less than 65%.

Please indicate the chapter/section/pages of this thesis that are extracted from a co-authored work and give the title and publication details or details of submission of the co-authored work.

<table>
<thead>
<tr>
<th>Nature of contribution by PhD candidate</th>
<th>Modeling, Intuition, Coding, Writing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Extent of contribution by PhD candidate (%)</td>
<td>65</td>
</tr>
</tbody>
</table>

CO-AUTHORS

<table>
<thead>
<tr>
<th>Name</th>
<th>Nature of Contribution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Anthony Downward</td>
<td>Editing, Discussion, Modelling</td>
</tr>
<tr>
<td>Golbon Zakeri</td>
<td>Editing, Discussion</td>
</tr>
</tbody>
</table>

Certification by Co-Authors

The undersigned hereby certify that:

- the above statement correctly reflects the nature and extent of the PhD candidate’s contribution to this work, and the nature of the contribution of each of the co-authors; and
- that the candidate wrote all or the majority of the text.

<table>
<thead>
<tr>
<th>Name</th>
<th>Signature</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Anthony Downward</td>
<td></td>
<td>29/3/2018</td>
</tr>
<tr>
<td>Golbon Zakeri</td>
<td></td>
<td>29/3/2018</td>
</tr>
<tr>
<td>Mahbubeh Habibian</td>
<td></td>
<td>29/3/2018</td>
</tr>
</tbody>
</table>

Last updated: 28 November 2017
This form is to accompany the submission of any PhD that contains published or unpublished co-authored work. **Please include one copy of this form for each co-authored work.** Completed forms should be included in all copies of your thesis submitted for examination and library deposit (including digital deposit), following your thesis Acknowledgements. Co-authored works may be included in a thesis if the candidate has written all or the majority of the text and had their contribution confirmed by all co-authors as not less than 65%.

Please indicate the chapter/section/pages of this thesis that are extracted from a co-authored work and give the title and publication details or details of submission of the co-authored work.

Chapter 4 and 5 are based on a paper that is submitted to Energy Systems Journal and is currently undergoing peer review. The paper is titled: Co-optimization of Demand Response and Interruptible Load Reserve Offers for a Price-Making Major Consumer

<table>
<thead>
<tr>
<th>Nature of contribution by PhD candidate</th>
<th>Modeling, Intuition, Coding, Analysis, Writing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Extent of contribution by PhD candidate (%)</td>
<td>80</td>
</tr>
</tbody>
</table>

### CO-AUTHORS

<table>
<thead>
<tr>
<th>Name</th>
<th>Nature of Contribution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Anthony Downward</td>
<td>Modelling, Editing</td>
</tr>
<tr>
<td>Golbon Zakeri</td>
<td>Modelling, Editing</td>
</tr>
<tr>
<td>Miguel F. Anjos</td>
<td>Modelling, Editing</td>
</tr>
<tr>
<td>Michael Ferris</td>
<td>Modelling, Editing</td>
</tr>
</tbody>
</table>

### Certification by Co-Authors

The undersigned hereby certify that:
- the above statement correctly reflects the nature and extent of the PhD candidate's contribution to this work, and the nature of the contribution of each of the co-authors; and
- that the candidate wrote all or the majority of the text.

<table>
<thead>
<tr>
<th>Name</th>
<th>Signature</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Anthony Downward</td>
<td></td>
<td>29/8/2018</td>
</tr>
<tr>
<td>Golbon Zakeri</td>
<td></td>
<td>29/8/2018</td>
</tr>
<tr>
<td>Miguel F. Anjos</td>
<td></td>
<td>20 August 2018</td>
</tr>
<tr>
<td>Michael Ferris</td>
<td>Michael C. Ferris</td>
<td>20 August 2018</td>
</tr>
<tr>
<td>Mahbubeh Habibian</td>
<td></td>
<td>29/8/2018</td>
</tr>
</tbody>
</table>

Last updated: 28 November 2017
Co-Authorship Form

This form is to accompany the submission of any PhD that contains published or unpublished co-authored work. Please include one copy of this form for each co-authored work. Completed forms should be included in all copies of your thesis submitted for examination and library deposit (including digital deposit), following your thesis Acknowledgements. Co-authored works may be included in a thesis if the candidate has written all or the majority of the text and had their contribution confirmed by all co-authors as not less than 65%.

<table>
<thead>
<tr>
<th>Nature of contribution by PhD candidate</th>
<th>Modeling, Intuition, Coding, Data Analysis, Writing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Extent of contribution by PhD candidate (%)</td>
<td>85 %</td>
</tr>
</tbody>
</table>

![Image](image)

**CO-AUTHORS**

<table>
<thead>
<tr>
<th>Name</th>
<th>Nature of Contribution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Anthony Downward</td>
<td>Editing, Discussion, Modelling</td>
</tr>
<tr>
<td>Golbon Zakeri</td>
<td>Editing, Discussion</td>
</tr>
</tbody>
</table>

**Certification by Co-Authors**

The undersigned hereby certify that:

- the above statement correctly reflects the nature and extent of the PhD candidate’s contribution to this work, and the nature of the contribution of each of the co-authors; and
- that the candidate wrote all or the majority of the text.

<table>
<thead>
<tr>
<th>Name</th>
<th>Signature</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Anthony Downward</td>
<td>[Signature]</td>
<td>29.8.2018</td>
</tr>
<tr>
<td>Golbon Zakeri</td>
<td>[Signature]</td>
<td>29.8.2018</td>
</tr>
<tr>
<td>Mahbubeh Habibian</td>
<td>[Signature]</td>
<td>29.8.2018</td>
</tr>
</tbody>
</table>

Last updated: 28 November 2017
Part I

Background
Chapter 1

Electricity Markets

In recent decades, many countries have moved towards deregulated electricity markets. The aim was to increase both short- and long-term efficiency, by promoting competition in order to share the risk between consumers and generators, and possibly to integrate capacity markets, to ensure a reliable supply of power. There has been extensive research undertaken on optimization of the generation side [5–8], and on improving market regulations and structure [9–12].

The complex nature of electricity networks is a significant concern in designing energy markets. Physical constraints such as thermal capacities of transmission lines, make it nearly impossible to build a reliable electricity market without a market operator. Within the past two decades, as the energy markets have moved towards deregulation, independent market operators have been ensuring a balanced dispatch of electricity.

Modern societies are heavily dependent on electricity, so that the failure to retain demand-supply equilibrium can result in jeopardizing safety and potentially major dysfunctions in day-to-day tasks. Therefore, ensuring the reliability of sustained electricity procurement is vital. In order to cover the risk of disruptions in energy dispatch, all electricity markets operate alongside ancillary services, known as the reserve market.

The growing dominance of technologies that require electricity, necessitates expanding the capacity of the transmission grid and generation units. Infrastructure of energy markets (e.g. transmission networks) is not shared with any other market, so that the cost of capacity expansion and maintenance is incorporated within the cost of energy and plays a big role in decision making. On the other hand, such isolation can be regarded as an opportunity to give more independence to policy makers and investors.

Energy markets operate at different stages that target short-term and long-term efficiency and risk management. First and foremost is the wholesale electricity market,
which is aimed at ensuring short-run efficiency of dispatch. The wholesale market includes the spot market, to which real-time data are submitted, and the pre-dispatch market. The resulting price and dispatch quantities are perceived as signals to make long-term contracts or short-term plans for future trading periods. Hence, alongside the wholesale market, there are capacity and contract markets that are built to reduce the risk of spot price fluctuations.

In this section we discuss the basic concepts of deregulated energy markets. First, we outline the design and functions of a wholesale electricity market. Next, we discuss different types of markets that are formed alongside the wholesale market, in order to enhance and support a reliable clearing and settling of wholesale transactions. In Section 1.3 we discuss the evolution of ancillary services and different types of reserves, along with their integration with the electricity market. Section 1.4, is dedicated to exploring demand response, and the progress that has been made towards introducing demand elasticity in energy markets. Section 1.4 presents a literature review on demand response, in particular for large manufacturers, since the main focus of this thesis is on industrial consumers’ participation in the spot and reserve markets.

1.1 Wholesale Electricity Market

The market for trading electricity among producers, retailers and industrial consumers is called the wholesale electricity market. In addition, there exists another market for retail, where electricity is traded between retailers and consumers. The focus in this chapter is the wholesale market, where producers and retailers interact to determine production quantities and prices.

Market Design

An efficient market design provides reliable electricity at the lowest cost to consumers. The first objective of such a market design is the short-run efficiency that focuses on optimizing electricity dispatch, using the existing infrastructure. The electricity markets are designed to coordinate the physical and financial transactions between producers and consumers across a distributed network.

Like most markets, the wholesale electricity market consists of sellers and buyers of a “good”. The buyers of electricity in the market are retailers and major consumers who buy electricity directly from the spot market, and the sellers are the generators.
Chapter 1 Electricity Markets

The major part of the wholesale market is the spot market, which is a real time commodity market for instant sale and delivery of energy. The goal of the spot market is to facilitate a least-cost dispatch of generation units subject to technical constraints. The time scale for electricity markets can be as small as a few minutes. The spot market allows producers of surplus energy to instantly locate available buyers for this energy, and deliver actual energy to the customers.

In order to facilitate the transfer of energy, the transmission grid connects the nodes in the network. The term nodes refers to points of significance related to injections or offtakes of energy or substations with voltage transitions. Mathematically transmission networks can be represented as network flow models [13].

**Independent System Operator**

The most common configurations used for the complicated trade of electricity is the centralized pool or an Independent System Operator (ISO). One of the main characteristics of reformed energy markets is that the various components of the electricity sector including production, transmission and distribution are no longer vertically integrated anymore. Hence the responsibility of economic dispatch and system security belongs to an ISO.

As an independent entity, the ISO is able to determine generation quantities and market prices. It should determine these values in such a way so as to make sure that the optimal dispatch is matched with total demand. The optimal dispatch should also be physically possible (i.e. it must be a feasible solution). For example, the optimal dispatch should comply with the thermal capacities of transmission lines. In addition, the ISO takes into account the loss of energy in the transmission lines. Other responsibilities of an ISO include determining tariffs on the transmission network and managing congestion of transmission lines [9].

The ISO determines the production quantity of each generator, and electricity prices at all nodes of the electricity network. There are three approaches to price electricity, locational Marginal Pricing (LMP) [14], single node dispatch and zonal pricing. Under LMP, the price of electricity is the marginal cost of generation at the node.

LMP determines the marginal cost of consumption or supply of energy at each node in the grid and theoretically improves investment decisions relating to the transmission grid. Moreover, LMP may introduce locational price risk for vertically integrated utilities that purchases power for their customers at a different node to that in which they
produce energy. Financial Transmission Rights (FTRs) [15] are designed to alleviate such financial risk.

In single-node dispatch, there is one clearing price for all the nodes. The transmission constraints are ignored and all generators and consumers are assumed to be located at a single node. Electricity markets such as England and Wales, NordPool, and Spanish electricity markets, are operated via this approach [16]. Single-node dispatch can lead to a simpler marketplace without complex financial contrast. On the other hand, it disables locational signals to place generation plants efficiently.

Zonal pricing is the approach that combines the first two pricing schemes. In his method, at each zone the clearing price is calculated similar to a single node dispatch, while different zones may have different zonal marginal prices. In this thesis, our focus is on the LMP form of an energy market.

**Optimal Power Flow Problem**

The optimization problem that is solved to determine the optimal price and dispatch quantities is called the OPF problem. There are two types of OPFs that are considered in the literature, based on the types of current flow in a circuit. One deals with the alternating current (AC) and is called ACOPF, and the other one assumes direct current (DC), called DCOPF. The supply and demand function parameters used in this optimization problem are derived from the offers and bids of generators and consumers at each node in the market.

The generators submit offer stacks that include a specific number of quantity-price pairs referred to as tranches. The offer stacks represent the willingness of generators to produce the specified quantity for the associated price or a higher price. Similarly, consumers submit non-increasing bid stacks to the market that consist of quantity-price pairs. The bid stack reveals the willingness to purchase of specific quantities of electricity at the given prices by that consumer. Under LMP, the intersection of the generation offer curve with the demand bid curve determines system electricity price and the dispatch quantity at each node. Given the clearing price, all the offers with a price lower than the market price are fully dispatched and offers with prices higher than clearing price are not dispatched.
Pre-dispatch Markets

In order to reduce uncertainty and facilitate the reliability of the dispatch, generators and consumers participate in forward markets before the spot market. The most common type of pre-dispatch market is the day-ahead market, which determines the status of generating units, the generation scheduling and the hourly spot price at each time period of the following day. Electricity markets in England and Wales, NordPool, Spain, California, and PJM\textsuperscript{1} all have a pre-dispatch day-ahead electricity market [16, 17]. There are also hour(s)-ahead markets to provide generators with an approximation of real-time dispatch. However, such scheduling is only an estimate of the optimal real-time dispatch. The pre-dispatch market cannot handle real-time fluctuations and possible contingencies.

1.2 Forward Markets

The contract markets and capacity markets that are described below are not a part of the wholesale electricity market, but perform alongside it to further ensure efficiency and reduce the risk of the stochastic nature of the spot market.

Contract Markets

Fluctuations in electricity spot prices is one of the challenging aspects of buying and selling energy. One of the instruments used by generators, major producers and retailers to hedge against this risk is through committing to contracts months before the time of dispatch. There are several different over-the-counter hedge types, such as contracts for difference (CfD), fixed-price fixed-volume and the fixed-price variable-volume. These contracts are formed to hedge against different types of risk [18].

One of the most commonly used forms of risk management contracts is the CfD. In CfDs a pre-agreed price (the strike price) is set between the buyer and seller for a fixed quantity of electricity at a specific time. The buyer compensates the seller when the spot price is less than the contract price and when the spot price is higher, it is the seller who compensates the buyer for the amount deviated from the contract price. Contract markets are usually traded bilaterally. For instance, the New Zealand electricity market (NZEM) and Australian electricity market have a bilateral contract market for differences. Similarly, England and Wales use CfDs for bilateral (two-party) trades [19].

\textsuperscript{1}Pennsylvania, Jersey and Maryland power pool
Chapter 1 Electricity Markets

Capacity Market

Efficient long-term capacity expansion is significant for accommodating the use of fast-growing electricity-based technologies. One of the ways of ensuring long-run efficiency is through providing effective incentives for investments in the infrastructure. Given the high levels of uncertainty involved in the decision making for long time horizons, planning efficient capacity expansion is one of the most challenging objectives in electricity market design. In theory, efficient long-run investment is induced from the right spot prices. However, in practice, it is not the only factor, where the reliability requirement complicates the efficient investment [20].

1.3 Reserve Markets

In an electricity market, the balance of supply and demand must be checked continuously. The measure used for representing such balance is the system frequency (typically 50 or 60 Hz). The system would go out of balance when there is either a shortage or excess of supply with respect to demand. A supply shortfall causes an Under Frequency Event (UFE). Subsequently, such imbalances can also happen if there are unexpected, downward, demand fluctuations or transmission failures. Such imbalances may lead to generation units being de-synchronized.

There are two main types of reserve, the frequency regulating reserve and the contingency reserve. Frequency regulating reserves respond to continuous but relatively small fluctuations in demand and, also the variations in supply, that are mostly caused by intermittent renewable generators. Regulating reserve is provided by generation units that continuously ramp their production level up and down [21]. On the other hand, contingency reserves are typically needed to manage sudden fluctuations in the system due to the sudden shutdown of a large generation unit or, much less commonly, due to the loss of a large load center. Contingency reserve is procured from either fast-responding generation units, or load curtailment of major consumers who offer ILR [22].

When there is a relatively large disruption in the grid that cause the frequency to move outside the safe range, generators disconnect from the grid to avoid further equipment damage. This in itself causes more disturbance in frequency keeping and triggers a sequence of disruptions that may lead to cascading outages.
Co-optimized Energy and Reserve

Ensuring the grid operates at a proper level of reserve imposes additional costs on procurement of energy. Therefore, developing the right model for incorporating reserve constraints plays an important role in the efficient procurement of energy. To control the frequency, many power systems started by employing a simple reserve constraint in the dispatch algorithm. In [23] the required reserve level is set to be a percentage of the generation units’ maximum capacities. In addition, in [24] an algorithm is developed that determines the least expensive generation schedule required to maintain a specific regulating reserve level. However, these methods do not take into account the dynamic attributes of energy markets, which may lead to inefficiencies.

There has been significant literature on co-optimizing the energy dispatch with reserve procurement so that the system can be dispatched in a least-cost manner and in accordance with frequency control criteria [25]. In some energy markets, such as Singapore and New Zealand, contingency reserves are simultaneously optimized with the dispatch. This thesis is focused on the energy markets with a co-optimized reserve and energy market. In Chapter 1.5 we discuss the properties of the New Zealand co-optimized market in more detail.

1.4 Demand Response

Putting all the risk on the supplier of energy can result in high costs of both electricity and reserve procurement services. In addition, at times of scarcity the available units may be very expensive or polluting to operate, and demand can become greater than the capacity of all the available power plants altogether. Whereby, demand-side participation in the market can bring efficiency and reduce the risk of outages.

The main idea of demand response stems from the ability of consumers to re-act to the realized dispatch prices, through changing their upcoming electricity purchase decisions. For instance, A price spike that becomes several times more expensive than the normal rate is a good signal to cut down on consumption. Due to the hockey stick nature of the supply function (i.e. as the system capacity is approached the prices rise sharply), it is sufficient that only a small percentage of consumers reduce their load. By doing so, they not only avoid paying an abnormally high cost, in the process they reduce the prices that all others pay.

Energy markets are typically designed without considering consumers as genuine participants, being capable of playing an active role in the market, but simply as a load that
needs to be served under all conditions. This is partly because short-term demand for electricity was initially perceived as inelastic. Another reason for the bias of not including consumers in the policy making process is that, unlike the suppliers, the consumers’ primary task is not focused on electricity prices and procurement. Therefore, they do not have the resources and the financial incentive to invest significant time effectively in such a complicated task [26].

Since the early 1980s, a number of authors have discussed demand response in restricted formats (e.g. [27–29]) and have examined its potential and limitations [30]. In addition, the introduction of disruptive technologies such as solar photovoltaics and battery storage has motivated more research on household demand response [31–34]. In the literature on energy markets, implementing larger consumers’ demand response has mostly been viewed through the perspective of the ISO, and not that of the major consumers. For instance, in [35], various forms of demand response bids are discussed from the perspective of an ISO.

Furthermore, Chao, in [36] and [37], discusses efficient pricing of demand response based on an estimate of a customer’s baseline, where the baseline is the counter-factual consumption in the absence of demand response. Recently Ferris et al. [38] have extended this notion and developed a model for the ISO to control demand response efficiently. They designed a bi-level programming model where demand response decisions are almost completely delegated to the ISO, who also estimates the consumers’ baseline. In this rather restricted demand response setting, the ISO decides on optimal (minimal) demand response over a wholesale electricity market subject to a so-called net benefit requirement, set by a demand-weighted average price of electricity. The consumers in this model do not reflect their respective utilities; they are simply compensated for each megawatt hour curtailed with the locational marginal price of electricity, which may not align with their utility.

**Industrial Consumers**

When it comes to demand-side participation, the majority of the literature focuses on the residential sector’s response to high prices of energy. However, major consumers of energy play a major role in alleviating spike prices by load curtailment. In the past few decades, many major consumers have invested in accommodating elastic demand by introducing new technologies to their operations. In addition, some major consumers have installed on-site generation, or pump-storage that increase their elasticity of load procured from the wholesale market.
In each market the definition of a large consumer could differ, depending on the size of the market. Generally a consumer with a load of tens of MW is considered a major consumer. Whereby, large manufacturers form the majority of major customers of electricity. In the United States, 26% of electricity is consumed in the industrial sector [39]. In New Zealand, industries consume 44% of the energy, and a single manufacturer’s (Tiwai Aluminum Smelter) load is 15% of the total national demand.

Although most of the literature on demand response focuses on the residential sector, there are a limited number of studies on industrial consumers’ demand-side participation. In [40], the potential and limitations of industrial demand response are discussed. In addition, in a survey by Shoreh et al., the industrial demand response is discussed with the emphasis on participation of industrial consumers in ancillary services [41].

In the electricity market literature, industrial demand response is generally approached through Time of Use (ToU), or Real Time Pricing (RTP) schemes. Examples of industrial consumption management under ToU pricing have been discussed in [42, 43]. In addition, [44, 45] present mathematical models for industrial demand response with RTP. Moreover, in [46], Yusta and Dominguez lay out a model that compares the ToU and RTP methods. In [47], price responsiveness (PR) in measured, showing that larger consumers have higher index of PR and are more desirable for participation in the market. In [48], a method is developed to measure the response of large industrial customers to dynamic and marginal cost-based electricity prices over a broad range of industries. In [49] Conejo et al. present a model for major consumers of energy that optimizes the quantity of energy purchased from the pool, the amount of energy bought from bilateral contracts, and energy procured from a self-production facility.

In this thesis our focus is on the industrial consumers’ demand response, with the objective of maximizing the strategic consumer’s profit. In Section 1.5 we explore different forms of demand response in New Zealand with the emphasis on the major consumers’ part in providing demand elasticity.

1.5 New Zealand Electricity Market

New Zealand has a deregulated electricity market where energy is co-optimized with spinning reserve. New Zealand (NZ) has two main islands the North Island and South Island. The transmission grid runs through both islands which are connected by a high voltage DC line. NZ’s population is more than 4.5 million, two-thirds of whom live in the North Island. The normal range of annual rainfall for NZ is between 600 and 1600
mm, with the west coast of the South Island being considered the wettest area in NZ [50].

The energy sector in NZ benefits from the abundance of rain, which has led to a hydro-dominated electricity market. More than 60% of energy is procured from hydro-generators and the rest is split evenly across other renewable sources of energy and thermal generation. Total annual consumption of electricity in New Zealand is almost 40,000 GWh, and 32% of demand for electricity is from the residential sector.

Demand for electricity in the North Island is around 63% of the national consumption, but the North Island only generates 56% of energy which leads to a net transfer of power from the South Island to the North Island. Almost all of the South Island’s electricity (98%) is produced by hydro-generators, and less than 2% from wind. On the other hand, in the North Island, 29% and 28% of generation are through geothermal and gas sources respectively.

Industrial consumers make up the biggest share of electricity consumption, with 44% of national demand. A large portion of industrial consumption is by a few major manufacturers. Large consumers are potentially able to curtail and shift their load. This enables the market to effectively incentivize industrial demand response, which is the main focus of this thesis.

In this section we focus on the evolution of the NZEM. We will discuss the NZ wholesale electricity market’s reform, followed by a description of electricity transmission network in NZ. One of the key characteristics of the NZEM is how the spinning reserve is co-optimized with energy, hence we have devoted a section to presenting the properties of the NZ reserve market. Furthermore, we discuss demand response, and the significant role of major consumers in the NZEM, which has been enabled through dispatchable demand. Subsequently, we discuss the influence of the largest consumer in the NZEM, the Tiwai Point Aluminium Smelter.

### 1.5.1 Wholesale Energy Market

Currently, the NZEM operates as a deregulated electricity market, which functions as a uniform price auction with locational marginal pricing. Similar to many countries, New Zealand energy trading policies have gone through many changes over the past few decades [51]. Until the mid-1980s, electricity generation and transmission were owned and operated by a government ministry. In 1987, a policy reform was introduced that led to privatization and competition in retail and generation markets. In the subsequent few years, the ownership of generation and transmission became separated.
In the 1990s major amendments of existing market arrangements were studied to enable trading at marginal prices. A new proposed statue particularly provided for deregulation and information disclosure. Subsequently, a wholesale market was created by development of a light-handed regulatory configuration, and the competitive wholesale electricity market started under a multilateral contract [52].

As at 2018, the NZEM network consists of approximately 285 nodes. In the NZEM, retailers and large consumers make electricity bids at grid exit points (GXP), and generators offer electricity at grid injection points (GIP), through submitting offer stacks. These stacks are supply functions that indicate a willingness to sell blocks of energy at designated prices. In Figure 1.1, an actual energy offer stack is shown. Each step of an offer stack is called a tranche, which is defined by its amount of energy and the corresponding price.

![Energy offer stack](image)

**Figure 1.1: Energy offer stack**

In the NZ wholesale electricity market, prices are based on bids and offers from market participants, where the price is not capped. Nodal prices in the spot market are calculated every half-hour by the market clearing problem and vary depending on supply and demand and the location of the node on the national grid. The market clearing problem in the NZEM is referred to as SPD (Scheduling, Pricing and Dispatch). A vectorized replica of SPD (vSPD) is publicly available online in [53]. Figure 1.2 presents the monthly and yearly demand-weighted average energy prices in New Zealand, over the past eight years. In New Zealand, the demand-weighted average yearly prices are volatile and range from $58 to $87. The higher prices in the NZEM are mainly caused by dry winters, when the lake levels are low coupled with increase in demand due to the low winter temperatures.

The volatility of spot prices, particularly, in years of low lake levels may impose significant risks on market participants. Therefore, in order to hedge against the risk of price spikes, in addition to the spot market, there exists a market for long-term contracts, called the hedge market. Accordingly, in addition to buying electricity directly from the
spot market, retailers and industrial consumers may also enter into financial contracts, that smooth out their exposure to some of the volatility in the spot price.

However, in New Zealand, the development of hedge markets has been slow, due to the vertical integration of some generators and consumers which provides an alternative to hedging by contracts. Vertical integration is a strategy in which generators and retailers merge to deal directly with the consumer. As a result of this, vertical market power can be exercised to raise prices and increase profits for the whole market; this also makes obtaining hedge contracts harder for major consumers or generators and retailers that are not vertically integrated [54].

Transmission

The operations and management of the transmission network of the NZEM has also gone through reform, in order to keep up with the evolution of the wholesale market. In the 1990s, the ownership of distribution and supply became fully separated, and Transpower, the company that owns and maintains the transmission network, was established as a State owned Enterprise (SoE). In 2004, new regulatory framework was announced to develop grid reliability standards and investment in upgrade plans. In 2006, after facing an extended electricity blackout in Auckland, the government’s new policy emphasized investment in transmission infrastructure. This has led to installing and upgrading more transmission lines in the past decade.

Currently, the NZEM grid consists of 11,800 km of high voltage transmission lines and 178 substations and switchyards. Regional substations (referred to as GXPs) transfer electricity through the distribution network to local substations and on to the end users. Generally, transmission lines operate at 220 kV, although they can have capacity of up to 400 kV [55].
One of the most important parts of the NZ transmission grid is the high-voltage direct-current (HVDC) link between nodes Benmore (located in the middle of the South Island) and Haywards (located at the south of the North Island). This link integrates power supply between South and North Islands, and it is critical to balancing energy use between the two islands. Until it was upgraded in 1993, the HVDC inter-island link had normal operating voltages of 250 kV and a maximum power transmission capacity of about 600 MW. Recently, Transpower installed new converter station facilities and the two HVDC poles now have a total capacity of 1200 MW [52].

Normally, the HVDC flow is from the South Island to the North, since most of hydro-generators are based in the South. However, during winter months in dry years the flow is from North to South. The increase in the frequency of such dry years in the past few years has incentivized investing in other forms of renewable energy in the South Island [56].

1.5.2 Reserve Procurement

Ancillary services are designed to ensure a stable and reliable energy market. There are different types of reserve, such as frequency-keeping and contingency reserve. The first is in charge of retaining the frequency within the operating band. In the NZEM, the latter operates as the N-1 contingency reserve, which is calculated for each island (North and South). At each trading period, each island has its own calculated reserve level, where the amount required is the maximum of the largest operating generator on that island and the incoming flow on the HVDC, in order to secure an adequate amount of reserve against the shutdown of the largest generator in each island, or the disconnection of the HVDC link (whichever is larger) [57].

The NZEM employs two types of reserve. Fast instantaneous reserve (FIR) is a type of reserve that is available within six seconds of an unexpected generation or transmission outage. Sustained instantaneous reserve (SIR) is available within 60 seconds and must remain available for 15 minutes. The NZEM operates on co-optimized primary (FIR) and secondary (SIR) contingency reserves, with frequency-keeping being partially co-optimized. In this thesis the term reserve refers to the FIR or SIR contingency reserve and not to the frequency-keeping reserve. Examples of different terminology throughout the world is discussed in [21] and [58].

To illustrate the range and volatility of reserve prices, we plot the daily and weekly average SIR and FIR prices respectively. Figure 1.3 shows the daily average SIR prices in 2017. As demonstrated in this plot, North Island prices are normally higher than those in the South Island, and apart from a few spikes, the average daily SIR prices
are under $10. However, the individual time period reserve prices may get very high. For instance, on July 12th 2017. The North Island SIR price at 8pm was approximately $340/MWh. The FIR prices follow a similar pattern.

![Figure 1.3: SIR prices in 2017](image)

To show a more comprehensive trend of reserve prices, the weekly average FIR prices, from year 2010 to 2017 are depicted in Figure 1.4. While spikes in reserve prices are not frequent, their impact in annual costs to consumers is high, which incentivises demand response and in particular, ILR. The impacts of enabling ILR on reserve prices are discussed in an illustrative example in the next section.

![Figure 1.4: Yearly FIR prices](image)

The generators and major consumers can offer reserve through submitting monotone non-decreasing offer stacks (similar to Figure 1.1). Each admissible reserve offer stack for each type of instantaneous reserve must have a maximum of three price tranches for each trading period [59]. In order to ensure the availability of reserve, the optimal power flow (OPF) problem incorporates an additional set of physical constraints on reserve for generators, this is discussed in detail in Section 3.2.
Interruptible Load Reserve

In addition to the generators’ ancillary services procurement, major consumers can participate in the reserve market by offering ILR. Hence, given the NZEM’s co-optimized energy and reserve, a major consumers’ interruptible load reserve and demand are interrelated. When a consumer is called upon in response to an under-frequency event, it must lower the quantity purchased at that GXP in accordance with a dispatched reserve offer [59]. For FIR offers, the drop in load (in MW) is within one second of the grid system frequency falling to or below 49.2 Hertz and is sustained for a period of at least 60 seconds.

Major consumers can impact on reserve prices by offering ILR. In the following illustrative example, we discuss the impacts of ILR on energy and reserve prices. Consider a single-node market with two generators and one consumer with a load of 350 MW, with the generation and reserve offers given in Table 1.5.2. The market clearing problem for this system minimizes the cost, hence the cheaper offer, from Generator 2, is dispatched first, and the rest (350 − 160 = 190 MW) is procured through Generator 1.

<table>
<thead>
<tr>
<th>Energy Reserve</th>
<th>Quantity (MW) Price ($/MWh)</th>
<th>Quantity (MW) Price ($/MWh)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Generator 1</td>
<td>500</td>
<td>3.00</td>
</tr>
<tr>
<td>Generator 2</td>
<td>160</td>
<td>1.00</td>
</tr>
</tbody>
</table>

Table 1.1: Generators offers

Assume that this system’s reserve is calculated with an N − 1 contingency. Therefore, the required amount of reserve is equal to the largest dispatched generation unit, which is 190 MW at a price of $5.00, from Generator 1.

At this stage, the marginal cost of consumption (cost of consuming one more unit of energy) consists of paying for one more MW of electricity at the price of $3.00, which adds one more unit to the amount of N − 1 reserve (190 + 1 = 191), costing an additional $5.00. Therefore, the marginal cost is $3.00 + $5.00 = $8.00.

Now, assume a similar configuration, but suppose that the consumer is able to offer up to 300 MW of ILR at a price of $0.01. In this case, the price of reserve will drop to $0.01 and the new marginal cost of consumption will be $3.00 + $0.01 = $3.01. Major consumers who are able to curtail their load can potentially save money by simply offering a part of their load as ILR. The example above shows that flexible major consumers can cover the risk of their own load by offering ILR.
Reserve Capacity

The reserve capacity is another issue to consider in the NZ reserve market. In 2003, the prospect of a dry year for the hydroelectric system led to an electricity-saving campaign to help reduce the risk of winter power outages. Later that year, a new government commission was established to secure extended reserve generation throughout dry years without the need for power saving campaigns.

The new policy consisted of several initiatives, including procurement of low fixed cost options for reserve energy, which might have high variable costs; in addition, demand response by major consumers was proposed in this policy [60]. The market would procure extended reserve that would be called upon in the event of an unexpected failure of more than one large power plant or the breakdown of HVDC at times of high level of transfer. In addition, if needed, distribution companies and those large consumers that are directly connected to the grid would automatically curtail some of their load [52].

1.5.3 Demand Response

Most consumers of energy do not wish to have their load dispatched by the system operator and will remain outside the dispatch process. However, consumers with load that can be controlled quite precisely without significant cost, and those with on-site generation, may choose to participate more directly in the spot market. Recently, in the NZEM, demand-side participation has become a new focus for retailers and major consumers. The benefits of elasticity of demand are known; however, applying effective demand response, needs complying infrastructure and an adequate number of consumers who are willing to participate in the market. The NZEM is a successful case of implementing dispatchable demand in an electricity market. In addition, given that the industrial sector has a large share of total demand, the penetration of demand response can become high enough to affect prices significantly.

The hockey stick nature of offer stacks in the market, increases the effectiveness of demand-side participation in lowering market prices. Figure 1.5 shows an actual aggregated supply stack in the South Island for a morning peak in February 2018. For the presented time period, if the total demand gets close to 3,500 MWh, the price will increase dramatically. However, submitting bids in the market that are price-sensitive can prevent such jumps in the price, simply by adding elasticity into the demand bid.

Demand response is most beneficial at times of scarcity and unusual peak prices. Here we present an analysis in which we probe the potential impact of the two largest industrial consumers of energy in NZ in reducing the price spikes that were reported on 13th of
July 2017. The demand-weighted average of that day was $405/MWh. By reducing the load of both Tiwai Point Aluminum Smelter and NZ Steel (located at grid exit points Tiwai and Glenbrook, respectively) by 10% and 20% in each period over the full day, the national average price of energy in NZ drops to $336 and $264, respectively. Figures 1.6 and 1.7 show the impact on the energy prices at grid exit points Tiwai and Glenbrook.

Note that these two major consumers have adapted technologies that allow them to change their consumption rate and shift their load from time periods with high prices to off-peak time periods, which is the best known form of demand response. The NZEM has enabled such large manufacturers to express their elasticity via bid stacks, which is known as dispatchable demand.
Dispatchable Demand

Dispatchable demand is a relatively new feature of the NZEM that allows wholesale electricity purchasers to participate voluntarily in the spot market in a similar way to generators and therefore respond more efficiently to wholesale market conditions. A nominated bid submitted by a consumer can have a maximum of 10 tranches for each trading period [61]. Figure 1.8 demonstrates a bid stack that was submitted to the NZEM in February 2018. The bid stacks are monotone decreasing and show their acceptable price range for given quantities, which is particularly influential in preventing price spikes.

In practice, major consumers have not yet fully exploited the dispatchable demand, and most of bid stacks only try to mitigate costs associated with periods where the
price exceeds $1,000/MWh, while remaining inflexible for lower prices. One key issue is that because final spot prices are published after real time, electricity users that are exposed to spot prices are uncertain of the price they will have to pay for their electricity consumption until after they have made their consumption decisions. Such uncertainty, adds to the risk faced by manufacturers where the electricity price is one of the elements (among many others) that affect the cost of production. In addition, calculating the right stack is not an easy task for a major consumer, given that multiple factors are included in estimating the value of energy and a comprehensive understanding of the detail of energy markets is not the priority of most major consumers.

Aluminium Smelter

New Zealand’s Aluminum Smelter (NZAS) has been operating since 1971, and produces around 340,000 tonnes of aluminum each year. NZAS is the biggest industrial consumer of energy with around 12% of total national demand, where NZAS’s use of electricity is approximately equal to 704,000 households’ demand in New Zealand [62].

Most of the energy for the smelter is supplied from the Manapouri hydroelectric power station in the South Island, via two double circuit 220 kV transmission lines. The smelter has an approximately fixed consumption of 570 MW, which is more than 30% of total South Island demand. The production operation requires the NZAS to consume electricity with a constant current rate to keep the temperature of electrolysis cells at the operating range. However, the fixed load policy of the NZAS has marginalized its potential impact in the market. Hypothetically, by introducing demand elasticity, NZAS can become highly influential in determining energy and reserve prices at times of high prices (Figure 1.6). For instance, changes in NZAS demand could shorten the period during which electricity flows from the North Island to the South Island in dry winters and result in more competitive reserve prices.

NZAS collaborates with the Light Metals Research Centre (LMRC) at the University of Auckland, which is a centre for fundamental research, technology and development for the international smelting and light metals industry. Recently, the LMRC has been involved in implementing a new technology for smelters, known as the Shell Heat Exchanger, which has motivated our research. This technology enables the aluminum smelters to change the rate of their energy consumption, while keeping the temperature of the electrolysis cells in the desired range [63]. As a result of adopting this technology, NZAS would be able to benefit from dispatchable demand. In addition, its consumption flexibility enables it to offer ILR, which in addition to the revenue of selling reserve, can contribute to more competitive reserve and energy prices.
In this thesis our focus is on stochastic co-optimization of load and reserve with case studies for NZAS, which are also applicable to other major consumers. Planning dispatchable demand and ILR requires appropriate understanding of mechanisms and uncertainties within the market. In the next section we lay out the specific OPF problem of the NZEM, with a focus on the attributes of the co-optimization of reserve and energy.
Chapter 2

Mathematical Background

Review

In this chapter, we present the mathematical foundation on which we build our models of energy markets. We address the contributions of existing literature, and review the mathematical background that is used in this thesis.

2.1 Motivation and Structure

At each trading period in deregulated electricity markets, several units of energy are auctioned. Depending on the number, size and behavior of the agents in the market, the auction can be translated into different equilibrium models. Each agent in the market, maximizes its own profit and one of the significant factors that determines the type of optimization problem that is to be solved by a firm is its capability of impacting market prices. Thereby, the firms who are relatively small are considered price-takers, and the larger agents, whose actions can impact the clearing price, are modeled as price-makers.

Markets with price-maker agents are called oligopolistic, in which the strategic agents can anticipate reactions of their rival agents, exercise their market power and raise prices. Study of markets with oligopoly is more complex due to the use of bi-level optimization approaches to capture the impact of each strategic agent’s decisions on price. In order to be able to solve such bi-level programs with standard solvers, reformulation techniques, such as using the equivalent Karush-Kuhn-Tucker (KKT) conditions, are required.

Our main purpose in mathematically modeling an agent is to lay out an optimized policy over a time horizon. Given the volatility of demand and supply throughout a time horizon, properly modeling the impacts of present decisions on future actions is
important for achieving accurate policies. Dynamic programming is common tool to solve multistage decision making problems. Furthermore, by introducing uncertainty to the model, stochastic dynamic programming can be used for solving stochastic multistage optimization problems.

Large optimization programs that model real-world problems, can be utilized if they are solvable in practical time-frames. One of the powerful tools that facilitates solving computationally cumbersome problems is the use of decomposition methods to reformulate optimization problems and reduce their solve time. In different stages of this thesis, we have used reformulation methods to decrease the complexity of our proposed models. Lagrangian methods, for instance, have been used to decompose our stochastic multistage demand-side planning problem. In the following sections we discuss the mathematical foundation of tools and theorems we have used in this thesis.

The chapter is laid out as follows; first, we lay out the basics of auctions and discuss the energy market auctions and the resulting equilibria in more detail. Next, we present the mathematical tools to address equilibrium constraints. Furthermore, we discuss the properties and the extent of bi-level optimization in energy systems. We continue with a review of the dynamic programming approach and its application in long-term planning of supply and demand of energy. Finally, a brief mathematical introduction on Lagrangian methods is presented, with the emphasis on its contribution in decomposing complex integer programs.

2.2 Auctions

Auctions are platforms to buy and sell goods and services through offers and bids. In an auction, the highest value each bidder is willing to pay for the object that is being sold is unknown to the seller. The auctions can be for one or multiple different or identical objects. In electricity markets, agents participate repeatedly in auction rounds for buying and selling multiple units of electricity. Therefore we study multi-unit auctions, in which the several identical items are sold at the same time [64]. In electricity market multi-unit auctions, prices could either be discriminatory, i.e. set for each buyer-seller pair, or uniform across all buyers and sellers, which results from the marginally matched buyer and seller.

The focus of this thesis is on the uniform-price auctions, where multiple homogeneous goods are sold at a market clearing price in a way that the demand and supply are in balance. This concept was first introduced in [65]. Uniform-price auctions are widely used in electricity markets around the world. However, in 2001, England and Wales
adopted the discriminatory auction format, in order to prevent strategic manipulation of price by large agents in uniform-price auctions. Studies in [66] and [67] show that, under certain circumstances, the discriminatory auction has advantages over the uniform pricing.

In each iteration of the auction the electricity is traded between generators and consumers. The generators are the sellers of the electricity who offer their supply function in the market, and retailers and major consumers bid in the market via their demand functions. If the number of sellers and buyers in the market is large enough, so that they are small relative to the size of the market, the firms can be considered price-takers. The rationale behind this statement is that, since the market contains a large number of firms, the change in the quantity of each firm has little effect on the price. In the next section, we discuss the competitive equilibrium and its corresponding mathematical modeling method in more detail.

2.3 Equilibrium Constraints

In this section we use the term equilibrium, to imply that in the market aggregate demand is equal to the aggregate supply, and the resulting equilibrium price clears the market. Equilibrium constraints arise in methods of finding solutions to the market equilibrium models. Here, we present the fundamentals of the two types of equilibrium models that we will use throughout this thesis.

When all agents in a market are price-takers who maximize profit or minimize their individual costs, the market yields a competitive equilibrium. There has been a vast amount of research on existence of such equilibria, both in theory and in practice [68, 69], as well as their extended properties, such as the study of perfect competition under uncertainty [70].

However, having disproportionally large agents, and in particular for electricity markets, the transmission constraints may prevent markets from being fully competitive. Therefore, we also study the markets with price-maker agents who can anticipate the impacts of their actions, and make strategic decisions. We discuss both types of markets and present optimized demand and ILR bidding schemes for both price-taker and price-maker consumers.

In this section we discuss the reformulation of price-taker and price-making agents’ optimization problems through defining equilibrium constraints. A common technique for reformulating these models is through their equivalent Karush-Kuhn-Tucker (KKT)
conditions. Consider the following generic optimization problem:

$$\min_{x \in \mathbb{R}} c(x)$$

s.t. 
$$h_i(x) \leq 0 \quad i = 1, \ldots, m$$
$$g_j(x) = 0 \quad j = 1, \ldots, r.$$ 

Assume $x^*$ is a local optimal solution, and $h_i(x)$ and $g_j(x)$ are continuously differentiable at $x^*$. Then there exist $\nu_i \geq 0, \forall i = 1, \ldots, m$ and $\mu_j \geq 0, \forall j = 1, \ldots, r$, known as KKT multipliers, such that:

- $h_i(x) \leq 0 \quad \forall i = 1, \ldots, m$ [primal feasibility]
- $\nu_i \geq 0 \quad \forall i = 1, \ldots, m$ [dual feasibility]
- $\nu_i h_i(x) = 0 \quad \forall i = 1, \ldots, m$ [complementary slackness]
- $0 \in \partial c(x) + \sum_{i=1}^m \nu_i \partial h_i(x) + \sum_{j=1}^r \mu_j \partial g_j(x)$ [stationarity]

The set of equations above are called KKT conditions [71]. If $c(x)$ and $g_j(x)$ are continuously differentiable convex functions, and $h_i(x)$ functions are affine, the KKT conditions are sufficient for showing the global optimality of $x^*$.

### 2.3.1 Competitive Equilibrium

In competitive equilibrium models each player maximizes their utility function, while the strategies of the other players are considered fixed to them. Therefore, the equilibrium is found by simultaneously solving the optimization problems associated with each firm. The equilibrium can be found by deriving the equivalent set of KKT conditions of each agent’s optimization problem and solving them simultaneously.

Since, the KKTs include sets of equilibrium constraints, the problem that is solved is called a mathematical program with equilibrium constraints (MPEC). Solving such problems can be cumbersome, and in the literature there are various reformulations that attempt to solve MPECs with exact or approximate schemes. Throughout this thesis we work with MPECs and present algorithms to solve them.

In the competitive equilibrium, each firm takes the price at which it can sell its output as given. If these agents submit their cost and value functions as offers and bids, to the market, the price and dispatched quantities of the market clearing problem will be equivalent to the solution of the competitive equilibrium. In Chapter 3, we will lay out a comprehensive example of competitive equilibrium in NZEM.
2.3.2 Bi-level Optimization

To address hierarchical structure, multi-level programming is used, in which a set of nested optimization problems are defined over a feasible region. Each optimization problem has a set of decision variables, which may effect the objective function of some, or all levels [72]. In bi-level programming, the upper level planner controls the effective resource space for the lower level planner. This produces a solution structure with the feasible region viewed by the upper level as a non-convex feasible region.

Bi-level optimization is a well-studied area in the operations research literature. In [73], Shi et. al have extended the KKT approach for linear bi-level programming to solve a wider class of problems that their upper levels have arbitrary linear forms. Furthermore, in [74–76], different solution methods have been proposed that attempt to find the global optimum of such problems, and in [77, 78] the conditions for existence of such solutions are discussed. In Section 4 we discuss the use of bi-level optimization in electricity markets literature.

Here, we present the mathematical formulation of bi-level programming. Consider a general optimization problem in the form given below:

$$\begin{align*}
\text{[UL]} \quad & \max_y \ z(x, \lambda, y) \\
\text{s.t.} \quad & F(y) = 0 \\
\text{subject to} \quad & y \geq 0
\end{align*}$$

Where the vectors \(x\) and \(\lambda\) are obtained from a set of decision variables \((x)\) and their corresponding dual multipliers \((\lambda)\) of the convex optimization problem \([LL]\):

$$\begin{align*}
(x, \lambda) = \arg\min_x \ c(x, y) \\
\text{s.t.} \quad & f(x, y) = 0 \quad [\lambda] \\
& x \geq 0
\end{align*}$$

In order to solve a problem with the above description, we use a bi-level optimization program, with \([UL]\) as the upper level problem and, with \([LL]\) embedded inside of it as the lower level problem. Therefore, we have the bi-level optimization problem laid out as below:

$$\begin{align*}
\text{[BL]} \quad & \max_y \ z(x, \lambda, y)
\end{align*}$$
\[ \text{s.t. } F(y) = 0 \]
\[ y \geq 0 \]
\[ \text{[LL]} \min_x c(x, y) \]
\[ \text{s.t. } f(x, y) = 0 \quad [\lambda] \]
\[ x \geq 0 \]

In electricity markets the bi-level optimization is used for agents who are considered price-makers in a pool-based market. The lower level optimization problem is the OPF problem, and the upper level is the strategic agent’s profit maximizing problem. In Chapter 4 we present the mathematical program for a price-maker consumer in the NZEM, and discuss the different reformulation methods that are used to solve it.

### 2.4 Dynamic Programming

When faced with large and cumbersome problems, dynamic programming (DP) is one of the methods that is widely used to tackle the complexity. Normally, the problem is broken down to smaller sub-problems that are sequentially solved in a recursive manner, in order to find a solution for the main problem.

DP algorithms are mostly used in multistage optimization models. In many cases, determining the optimal sequence of decisions can be reduced to finding the first iteration’s optimal decision. In such problems, a value function is defined for each state of the system, within the decision making step. Firstly, the last stage’s value functions are calculated, and the value functions corresponding to earlier stages can be determined, using a recursive relationship, known as the Bellman equation [79].

The Bellman equation (presented in (2.1)) was first introduced in [80], followed by [81], which further probed the application of dynamic programming.

\[ V(x) = \sup_{y \in \Gamma(x)} \{ F(x, y) + \beta V(y) \} \]  
(2.1)

Here \( x \) is called a state variable, and \( G(x) = \{ y \in \Gamma(x) : V(x) = F(x, y) + \beta V(y) \} \), where \( G(x) \) is called a policy function, if it is single-valued.

### Stochastic dynamic programming

There is a large class of multistage processes that are not deterministic and both the effect of decisions and the realization of future states are uncertain. Most frequently,
stochastic dynamic programming (SDP) is used to obtain an expected optimal policy. In a finite SDP, the optimal policy maximizes the expected value of the sum of the rewards earned over a finite time horizon [82]. Whereby, at each recursive step, equation (2.2) is solved:

$$V_t(s) = \max_{a \in A_t} [R(s, a) + \sum_{s' \in S'} P(s'|s, a)V_{t+1}(s')]$$  \hspace{1cm} (2.2)$$

Here, $V_t(s)$ denotes the maximum expected return for stages $t, t+1, \ldots, T$, given the stage $t$, starts with the state $s$. $R(s, a)$ is the reward gained at state $a$, by action $a$. $A_t$ is the set of all possible actions at stage $n$. $P(s'|s, a)$ is the probability of being in state $s'$ at the beginning of stage $t+1$, given the current state $s$, and taking the action $a$. Although, to model multistage problems, SDP is a widely used approach, in practice, for long time horizons, or large number of states, the so-called “curse of dimensionality” makes the problem computationally very challenging to solve. Therefore, approximate solution techniques are often used to generate “good” policies for large problems. In Chapter 7, we introduce an SDP for the multistage demand-response problem, and discuss the heuristic approaches to tackle our large-scale problem.

2.5 Lagrangian Methods

Lagrangian methods are considered among the most effective approaches towards solving “hard” problems. The main idea is to remove the complex constraints, and price them in the objective function with Lagrangian multipliers. Lagrangian methods are widely used in integer programming, where they facilitate finding a lower bound (for minimization problems) on the optimal objective function value. Thereby, Lagrangian relaxations can be used as a relaxation in a branch and bound method [83].

In the next parts of this section, we discuss the mathematical properties of Lagrange multipliers, which is followed by the Lagrangian sufficiency theorem and its utilization in broad range of problems.

Lagrange multiplier

Suppose we are given a generic optimization problem in the form below:

$$[\mathbf{P}]: \min \ f(x)$$

s.t. \hspace{1cm} $g(x) = b$ \hspace{1cm} $[\lambda]$
We define the Lagrangian as:

\[
L(x, \lambda) = f(x) - \lambda^\top (g(x) - b)
\]  

(2.3)

Here \( \lambda \) is a vector, and each component of \( \lambda \) is called a Lagrange multiplier. Note that, each of these multipliers have one corresponding constraint. By utilizing Lagrangian multipliers some or all of the constraints can be relaxed and instead penalized in the objective function, while being weighted by their corresponding value of \( \lambda \). This method was first called Lagrangian relaxation in [84]. If we use Lagrangian relaxation for all constraints in \([P]\) we will have a new optimization problem \([P2]\), laid out as below:

\[
[P2]: \min \ f(x) - \lambda^\top (g(x) - b)
\]

s.t. \( x \in X \)

In the following part, we introduce a theorem that discusses the relationship between \([P2]\)- and \([P]\)’s optimal solutions.

**Lagrangian sufficiency theorem**

**Theorem 2.1.** If \( x^* \) and \( \lambda^* \) exist\(^1\) such that \( x^* \) is feasible for \([P]\) and \( L(x^*, \lambda^*) \leq L(x, \lambda^*) \) \( \forall x \in X \), then \( x^* \) is optimal for \([P]\).

**Proof.** Define

\[
X_b = \{x : x \in X \text{ and } g(x) = b\}
\]

Hence \( \forall x \in X_b \) we have:

\[
L(x, \lambda) = f(x) - \lambda^\top (g(x) - b) = f(x)
\]

Now \( x \in X_b \subseteq X \), and so by assumption:

\[
f(x^*) = L(x^*, \lambda^*) \leq L(x, \lambda^*) = f(x) \quad \forall x \in X_b.
\]

Which concludes, \( x^* \) is optimal for \([P]\). \( \square \)

Apart from the broad application of Lagrangian methods in linear programming, various solution methods utilize this theorem to approximately solve integer programs, such as

\(^1\)For integer/non-convex problems there is no guarantee that such a solution \((x^*, \lambda)\) exists.
multiplier adjustment method, finding feasible solutions, and guiding the search in enumeration methods [83, 85, 86]. In Chapter 6, we use Lagrange multipliers to decompose a multistage mixed integer program.
Part II

Single-Stage Co-Optimization of Energy and ILR
Chapter 3

Price-taking Agents

3.1 Introduction

Analysis of efficient markets generally will assume that no agent will have market power. Practically, this means that all agents are price takers. These agents observe a distribution of prices that cannot be influenced by their actions. This price-taking assumption is possibly valid in markets with a large number of competing agents or strict regulation on bidding and offering behavior. Therefore, in this chapter we lay out the properties of the competitive equilibrium that is derived from an electricity market that only has price taker agents.

In a competitive equilibrium, every agent has self-interested behavior to maximize its own profit. Given that these agents are price takers, they do not need information on other firms’ cost and value functions, and their knowledge about their own firm (and the distribution of prices) is sufficient to take optimal decisions. The competitive allocation is the aggregate outcome of all the firms’ actions and the physical laws governing the flow of electricity, where the resulting clearing prices achieve efficiency. In [68] proofs of the existence of an equilibrium for a competitive economy are given.

The OPF problem in electricity markets determines the economic dispatch, which is based on bids and offers of consumers and suppliers of energy. For an energy-only electricity market, in the presence of convexity and perfect competition, it is well known that the clearing price derived from the dispatch problem is efficient, in the sense that it maximizes the total welfare of producers, consumers and the transmission operator.

For electricity markets in which reserve and energy are co-optimized, such as New Zealand and Singapore, the agents are involved in repeated multi-good auctions. The generators and major consumers who offer reserve, submit their energy and reserve bids
to the market based on their corresponding co-optimized cost and value functions for the two commodities. In this chapter, we show that for marginal-utility-cost revealing consumption bids and energy and reserve offers, the co-optimized welfare-maximizing economic dispatch is also a price-taking competitive market equilibrium.

In the co-optimized OPF model, generators/major consumers must submit energy offers/consumption bids that are independent from their reserve/ILR offers. This means that firms may be unable to reflect their true utility function for sales of these products and this may result in inefficiencies. In the final part of this chapter, we examine the cost of independent bidding/offering for two co-related commodities, in a multi-item multi-unit auction.

The outline of the chapter is as follows: first we define the nomenclature that is used in the mathematical programs that are discussed in this chapter. We then lay out the NZEM’s co-optimized energy and reserve dispatch problem that is solved for each half-hour trading period as a multi-good and multi-unit uniform-price auction. In Section 3.3, we present a model for the competitive market, as a multiple optimization problem with equilibrium constraints (MOPEC). We then show that the MOPEC that is derived from individual firm’s profit maximization problems is equivalent to the OPF model introduced in Section 3.2. Furthermore, in Section 3.4, we will probe the effect of the inability to reflect one’s true cost or value within an offer or bid of a multi-item auction. Lastly in, Section 3.5, we conclude the chapter.

**Specific Chapter Nomenclature**

**Parameters**

- $f_{ij}$: The flow between node $i$ and $j$.
- $K_{ij}$: The line capacity in arc $ij$.
- $r_e$: The reserve level required in island $e$.
- $V_n$: The minimum amount of difference between ILR and consumption, at node $n$. It denotes the level of consumption that can not be interrupted.
- $B_n$: The fraction of generation allowed to be offered at reserve, at node $n$.
- $W_n$: The maximum total amount of generation and reserve, offered by the generator, at node $n$. 
Sets, Matrices and Indices

$\mathcal{N}$ The set of all nodes in the network. Note that in our model we have one agent per node. This is not a restrictive assumption.

$\mathcal{A}$ The set of all arcs in the network.

$\mathcal{N}_e$ The set of all nodes in island $e$.

$L$ The loop constraint matrix, where $L_{l,ij}$ corresponds to row $l$ (associated with each loop) and the column $ij$ corresponds to arc $ij$.

$\mathcal{T}_{i_e}^n$ The set of interruptible consumption tranches.

$e_n$ indicates the island that node $n$ is located in.

$\{\mathcal{N}, \mathcal{S}\}$ The set of islands, north and south.

$\mathcal{Z}$ The set for types of tranches, i.e. consumption, generation, ILR and reserve.

$\mathcal{Z} = \{c, g, rc, rg\}$

$\mathcal{T}_z$ The set of all offered tranches of type $z$.

$\mathcal{T}_z^n$ The set of all tranches of type $z$ at node $n$.

Variables

$x^z_t$ The variable associated with the dispatch quantity tranche type $z$.

$p^z_t$ The price of tranche $t$ of type $z$.

$q^z_t$ The quantity of tranche $t$ of type $z$.

$r_e$ The reserve level required in island $e$.

3.2 Co-optimized Energy and Reserve Optimal Power Flow Problem

In deregulated electricity markets, including the NZEM, a so-called OPF market clearing problem is solved to determine the optimal dispatch of generation and the corresponding locational marginal prices of electricity [14]. In the NZEM, the optimal dispatch also ensures that the North and South Islands each have sufficient reserves procured against an $N-1$ contingency, where a large source of energy supply may trip (in either island). To model the physical requirements around reserve, the dispatch problem incorporates three additional sets of physical constraints on reserve, for generators. Firstly, ramping constraints on generating turbines prevent a steep rate of increase in production in the event that reserves are called upon. Secondly, there is an upper bound on each reserve offer. Lastly, each generating unit has a maximum capacity which must be divided
between energy and reserve. These three constraints together constitute the so-called inverse bathtub constraints, formulated by the inequalities below.

\[ x^r \leq B x^g, \]
\[ x^r \leq R, \text{ and} \]
\[ x^r + x^g \leq W. \]

Here \( x^r \) is the procured reserve and \( x^g \) denotes the dispatched generation. Our ramp rate parameter is \( B \), whereas \( R \) and \( W \) denote the maximum reserve offer and the unit capacity, respectively. Figure 3.1 depicts the feasible region that is formed by these constraints for each generator.

Large consumers who offer ILR need to comply with the following constraint that ensures that their ILR level is not more than their curtailable load. Here \( V \) is the amount of consumption that cannot be interrupted\(^1\).

\[ x^{ILR} \leq x^d - V \]

In order to lay out the OPF problem we also need to address both the thermal limits of the lines and Kirchhoffs Law for loop flow constraints. In (3.5) and (3.6) we demonstrate the loop flow and transmission capacity constraints respectively. Here \( F \) is the maximum transmission capacity, \( f \) is the transmission flow, and \( L \) is the loop flow matrix.

\[ Lf = 0 \]
\[ |f| \leq F \]

By accounting for the above constraints, we have laid out the full dispatch model as below. We denote this energy and reserve optimal power flow problem as the [EROPF].

\(^1\)In this chapter, we have incorporated (3.4) in the dispatch model, in order to ensure feasibility and compliance. However, in practice, in the NZEM, this constraint has not yet been implemented in the SPD model, and must be internalized by the consumer.
Here, the flow constraints ((3.5) and (3.6)) are applied in (3.9) and (3.10). Inverse bathtub constraints (3.1) and (3.3) are captured in [EROFP] as (3.13) and (3.14). In addition, constraint (3.2) is embedded in (3.11). Note that these sets of additional constraints link the energy and reserve prices (dual variables on constraints (3.7) and (3.8), respectively).

\[
\text{[EROFP]} \quad \max_{x^c, x^g} \sum_{t \in T_c} p^c_t x^c_t - \sum_{t \in T_g} p^g_t x^g_t - \sum_{t \in T_{rg}} p^r_{tg} x^r_{tg} - \sum_{t \in T_{rc}} p^r_{tc} x^r_{tc} \\
\text{s.t.} \quad \sum_{t \in T^n_c} x^c_t + \sum_{i|ni \in A} f_{ni} - \sum_{i|in \in A} f_{in} = \sum_{t \in T^n_g} x^g_t \quad [\pi^d_n] \quad (3.7) \\
- \sum_{n \in \mathcal{N}_e \cap \{rc, rg\}} \sum_{t \in T^n_c} x^c_t = -r_c \quad [\pi^r_c] \quad (3.8) \\
\sum_{ij \in A} L_{liij} f_{ij} = 0 \quad [\lambda_l] \quad (3.9) \\
-K_{ij} \leq f_{ij} \leq K_{ij} \quad [\eta^+_{ij}, \eta^-_{ij}] \quad (3.10) \\
0 \leq x^c_t \leq q^c_t \quad [\nu^+_c, \nu^-_c] \quad (3.11) \\
\sum_{t \in T^n_n} x^r_{tc} - \sum_{t \in T^n_{rc}} x^c_t \leq 0 \quad [\theta_n] \quad (3.12) \\
\sum_{t \in T^n_g} x^r_{tg} \leq B_n \sum_{t \in T^n_g} x^g_t \quad [\phi_n] \quad (3.13) \\
\sum_{t \in T^n_g} x^r_{tg} + \sum_{t \in T^n_g} x^g_t \leq W_n \quad [\phi'_n]. \quad (3.14)
\]

Here (3.7), and (3.12) to (3.14) hold $\forall n \in \mathcal{N}$. (3.8) holds $\forall e \in \{N, S\}$. (3.9) holds for each loop, indicating that sum of impedance adjusted flows across the loop must be zero. The inequalities in (3.10) hold $\forall ij \in \mathcal{A}$, whereas those in (3.11) hold $\forall t \in \mathcal{T}_z, \forall z \in \mathcal{Z}$. Besides the analysis on the competitive equilibrium in the NZEM that is discussed in this chapter, we will use the NZEM’s OPF problem ([EROFP]) as the base of our bi-level optimization problems in the following chapters.

The EROPF problem above is convex, therefore it is equivalent to its set of KKT conditions. We first present the dual feasibility constraints which corresponds to the energy and reserve prices for the [EROFP] is laid out as below:

\[
\begin{align*}
\hat{p}_t^c &= \pi^d_n + \nu^+_c - \nu^-_c - \theta_n & \forall t \in \mathcal{T}^n_c, \forall n \in \mathcal{N} \\
\hat{p}_t^g &= \pi^d_n - \nu^+_g + \nu^-_g + B_n \phi_n - \phi'_n & \forall t \in \mathcal{T}^n_g, \forall n \in \mathcal{N} \\
\hat{p}_t^{rc} &= \pi^r_{en} - \nu^+_t^{rc} + \nu^-_t^{rc} - \theta_n & \forall t \in \mathcal{T}^n_{rc}, \forall n \in \mathcal{N} \\
\hat{p}_t^{rg} &= \pi^r_{en} - \nu^+_t^{rg} + \nu^-_t^{rg} - (\phi_n + \phi'_n) & \forall t \in \mathcal{T}^n_{rg}, \forall n \in \mathcal{N}
\end{align*}
\]
\[\pi_i^d - \pi_j^d + \eta_{ij}^+ - \eta_{ij}^- + \lambda_l L_{l,ij} = 0 \quad \forall l \in \mathcal{L}, \forall ij \in \mathcal{A} \] (3.19)

The last set of KKT conditions pertains to the orthogonality conditions that are derived from the complementarity slackness. Below we present the complementarity constraints for the [EROPF] model.

\[0 \leq x_i^t \perp \nu_i^t^- \geq 0 \quad \forall t \in \mathcal{T}_z, \forall z \in \mathcal{Z} \] (3.20)

\[0 \leq q_i^t - x_i^t \perp \nu_i^t^+ \geq 0 \quad \forall t \in \mathcal{T}_z, \forall z \in \mathcal{Z} \] (3.21)

\[0 \leq \sum_{t \in \mathcal{T}_n^g} x_i^t - \sum_{t \in \mathcal{T}_r} x_i^t \perp \theta_n \geq 0 \quad \forall n \in \mathcal{N} \] (3.22)

\[0 \leq B_n \sum_{t \in \mathcal{T}_n^g} x_i^t - \sum_{t \in \mathcal{T}_r} x_i^t \perp \phi_n \geq 0 \quad \forall n \in \mathcal{N} \] (3.23)

\[0 \leq W_n - \sum_{t \in \mathcal{T}_r^g} x_i^t \perp \phi_n' \geq 0 \quad \forall n \in \mathcal{N} \] (3.24)

\[0 \leq f_{ij} + K_{ij} \perp \eta_{ij}^- \geq 0 \quad \forall ij \in \mathcal{A} \] (3.25)

\[0 \leq K_{ij} - f_{ij} \perp \eta_{ij}^+ \geq 0 \quad \forall ij \in \mathcal{A} \] (3.26)

In the next section we lay out the MOPEC corresponding to the equilibrium arising from simultaneously solving each agent’s welfare maximization optimization problem, and show the equivalence of the MOPEC to the energy and reserve co-optimized power flow problem.

### 3.3 Competitive Equilibrium

A competitive equilibrium is the combination of optimizing behavior for buying and selling firms, as well as a set of market clearing conditions. Deregulated electricity markets promote competition to ensure merit-order dispatch, whereas in the literature, many researchers have assumed a competitive equilibrium as the base of their model. For instance, in [87] an electricity market with competitive equilibrium is introduced and the increase in efficiency by participation of consumers in real-time pricing is studied.

For convex OPF problems, the competitive equilibrium exists, and the efficient market clearing price is derived from the Lagrange multiplier of the supply-demand balance constraint. In addition competitive equilibria may exist in electricity markets with non-convex network constraints and nonlinear cost and value functions, even when the duality gap is nonzero for the OPF problem [88].

Centralization of power, however, can interfere with achieving a competitive equilibrium. To address this issue, a deregulated market mechanism is developed in [89] by utilizing
the Lagrangian duality theory, which ensures that self-interested market participants, would reach a competitive equilibrium.

In this section we demonstrate the efficiency of energy and reserve prices under the price-taking agents assumption. We introduce an equilibrium model where each agent optimizes their cost/utility function, where the energy and reserve prices are Walrasian. We then show that the solution to the equilibrium pertaining to the NZEM model is equivalent to the optimality conditions of [EROPF]. This result naturally leads to a simple optimized offer strategy, for a perfectly competitive large consumer: namely to bid in their marginal utility of consumption, and offer all their available ILR capacity at the cost of interruption.

In what follows, we have assumed that each generator offers, and each consumer operates at a single node of the network. This assumption is not restrictive and is merely for ease of presentation. Problems [GP_n] and [CP_n] refer to a generator’s problem and a consumer’s problem (located at node n), respectively.

\[
[\text{GP}_n] \quad \max_{x^g, x^r} \sum_{t \in T^g_n} (\pi^d_{it} x^g_t - p^q_{it} x^g_t) + \sum_{t \in T^r_n} (\pi^r_{it} x^r_t - p^q_{it} x^r_t) \\
\text{s.t.} \quad 0 \leq x^g_t \leq q^g_t \quad [\nu^+_n, \nu^-_n] \\
0 \leq x^r_t \leq q^r_t \quad [\nu^+_n, \nu^-_n] \\
\sum_{t \in T^g_n} x^g_t + \sum_{n \in T^r_n} x^r_n \leq W_n \quad [\phi'_n] \\
\sum_{n \in T^r_n} x^r_t \leq B_n \sum_{t \in T^g_n} x^g_t \quad [\phi_n]
\]

\[
[\text{CP}_n] \quad \max_{x^c, x^r_c} \sum_{t \in T^c_n} (p^c_{it} x^c_t - \pi^d_{it} x^c_t) + \sum_{t \in T^r_c_n} (\pi^r_{it} x^r_t - p^q_{it} x^r_t) \\
\text{s.t.} \quad 0 \leq x^c_t \leq q^c_t \quad [\nu^+_n, \nu^-_n] \\
0 \leq x^r_c \leq q^r_c \quad [\nu^+_n, \nu^-_n] \\
\sum_{t \in T^c_n} x^c_t - \sum_{n \in T^r_c} x^r_t \geq V \quad [\theta_n]
\]

The transmission operator maximizes the congestion rents:

\[
[\text{TO}] \quad \max_f \sum_{i,j \in A} (n_{ij}^d - \pi^d_{ij}) f_{ij} \\
\text{s.t.} \quad \sum_{i,j \in A} L_{t,i,j} f_{ij} = 0 \quad [\lambda_t] \quad \forall t \in L
\]

---

\(^2\)In this section we assume that each agent’s cost/utility functions are separable. Later in this chapter we discuss the implementation of joint cost functions.
$$-K_{ij} \leq f_{ij} \leq K_{ij} \quad [\eta_{ij}^+, \eta_{ij}^-] \quad \forall ij \in A$$

[WP] denotes the supply-demand balance and market clearing conditions that result in Walrasian prices:

$$\begin{align*}
\text{[WP]} & : 0 = \sum_{t \in T^g} x_t^g - \sum_{t \in T^c} x_t^c - \sum_{i|m \in A} f_{mi} + \sum_{i|m \in A} f_{in} \perp \pi^d_n & \quad (3.27) \\
0 & = \sum_{n \in N} \sum_{z \in \{rc, rg\}} \sum_{t \in T^z} x_t^z - r_e \perp \pi^r_e & \quad (3.28)
\end{align*}$$

Note that in (3.27) and (3.28) the orthogonality relationships could be written as inequality complementarity relationships if the modeler assumed free disposal of power (this would ensure non-negative prices). The MOPEC model solves all these optimization problems simultaneously, by replacing each optimization problem with its optimality conditions. We use $\perp$ as the orthogonality operator in the below formulation:

$$\begin{align*}
\text{[WP]} & \\
\text{[TO] feasibility constraints} & \quad (3.29) \\
\text{[GP] and [CP] feasibility constraints} & \quad \forall n \in N \\
p_t^c & = \pi^d_n + \nu_t^c+ - \nu_t^c- - \theta_n & \quad \forall t \in T^n_c, \forall n \in N & \quad (3.30) \\
p_t^g & = \pi^d_n - \nu_t^g+ + \nu_t^g- + B_n \phi_n - \phi'_n & \quad \forall t \in T^n_g, \forall n \in N & \quad (3.31) \\
p_t^{rc} & = \pi^r_n - \nu_t^{rc+} + \nu_t^{rc-} - \theta_n & \quad \forall t \in T^n_{rc}, \forall n \in N & \quad (3.32) \\
p_t^{rg} & = \pi^r_n - \nu_t^{rg+} + \nu_t^{rg-} - (\phi_n + \phi'_n) & \quad \forall t \in T^n_{rg}, \forall n \in N & \quad (3.33) \\
0 & \leq x_t^z \perp \nu_t^z \geq 0 & \quad \forall t \in T_z, \forall z \in Z & \quad (3.34) \\
0 & \leq q_t^z - x_t^z \perp \nu_t^z+ \geq 0 & \quad \forall t \in T_z, \forall z \in Z & \quad (3.35) \\
0 & \leq \sum_{t \in T^n_c} x_t^c - \sum_{t \in T^n_{rc}} x_t^{rc} - V \perp \theta_n \geq 0 & \quad \forall n \in N & \quad (3.36) \\
0 & \leq B_n \sum_{t \in T^n_g} x_t^g - \sum_{t \in T^n_{rg}} x_t^{rg} \perp \phi_n \geq 0 & \quad \forall n \in N & \quad (3.37) \\
0 & \leq W_n - \sum_{t \in T^n_{rg}} x_t^{rg} - \sum_{t \in T^n_g} x_t^g \perp \phi'_n \geq 0 & \quad \forall n \in N & \quad (3.38) \\
\pi^d_i - \pi^g_i + \eta^+_ij - \eta^-ij & = 0 & \quad \forall ij \in A & \quad (3.39) \\
0 & = \sum_{ij \in A} L_{ij} f_{ij} \perp \lambda_l & \quad \forall l \in L & \quad (3.40) \\
0 & \leq f_{ij} + K_{ij} \perp \eta^-ij \geq 0 & \quad \forall ij \in A & \quad (3.41) \\
0 & \leq K_{ij} - f_{ij} \perp \eta^+ij \geq 0 & \quad \forall ij \in A & \quad (3.42)
\end{align*}$$

Here (3.30) – (3.42) are the combined optimality conditions of [TO], [GP]_n, and [CP]_n.
which altogether form the MOPEC’s KKT. Recall now the set of KKTs that we presented in previous section for the [EROPF], which are identical to the MOPEC KKT laid out above. Observe that the prices \(\pi^d\) and \(\pi^r\) decouple the [EROPF] into this MOPEC.

In this welfare maximizing mechanism, each agent is expected to submit its true cost and utility functions. In this model we have implicitly assumed that the the utility functions of all agents are separable. However, the separate bidding and offering for energy and reserve does not allow agents to represent their true joint utility/cost functions. We explore the issue of multi-item auctions further in the next section, where we discuss how the limitation of this market design can lead to inefficient dispatch.

3.4 Multi-item Auctions

In electricity markets with co-optimized energy and spinning reserve, the agents submit separate bids for the two commodities, and therefore the two items have separate clearing prices. Such markets operate as multi-item auctions, and in the case of co-optimized electricity and reserve markets, there exist several units of energy and reserve, where reserve and energy are treated as heterogeneous commodities. However, the two items are co-related, given the cost of procurement of reserve, might incur opportunity costs for generators or major consumers. This is similar to the idea of complements and substitutes in differentiated products models which was introduced by Vives in [90]. Multi-item auctions are studied in the economic literature as a generalization on the second-price auction. The focus of the literature in this area, is on the mechanism in which bidders are interested in buying only one item, and it is required for each bidder to submit a bid listing his or her valuation of all the items. The existence of equilibrium for such auction is proved in [91]. Furthermore, Demange and Gale, in [92], introduce the strategy structures for the two-sided matching markets under generalized conditions, in which a unique vector of equilibrium prices is optimal. In [93], the existence of equilibrium in two different dynamic auction mechanisms is discussed. These types of auctions are used in job or university allocation, where each position is occupied by one person. In addition, in [94], the structure of the winning bids in multi-item progressive online auctions are studied.

Our focus, however, is on auctions in which the sellers offer multiple type of items, which is different from the conventional multi-item auctions layout in which each seller has only one item for sale. Our work is based on multi-unit multi-item auctions, or in other words, auctions for multiple groups of divisible goods. In [95], this type of multi-item auction is provided with results on clock auctions. Also, in [35], a model is
presented that extends the bid format for demand response for electricity, which enables load-shifting over time, where the energy consumed at different times can be considered as different products, that are partial substitutes.

In co-optimized energy and reserve markets, electricity and reserve supply stacks are submitted separately, however, the value and cost functions are dependent on other commodity’s dispatch price and quantities. In [96] a model with ”externalities” is presented in the context of multi-item auctions, in which the preference over two objects for a buyer, may depend on the price of a third object. In [97], Read introduces “fan curves”, that capture the impact of offering reserve in the generation offer stacks for generators in co-optimized markets.

Major consumers of electricity in a co-optimized energy and reserve market are simultaneously buyers of energy and sellers of ILR, in a multi-item auction. The main goal of this section is to examine the effect of offering/bidding for the two co-related commodities, and its efficiency implications for such a major consumer of energy.

3.4.1 A major consumer’s electricity and ILR joint value-cost function

In the NZEM, for dispatch-capable load stations, who also provide ancillary services, the amount of offered ILR, must not exceed the dispatched load (unless they have some additional embedded generation). The demand bids submitted by the consumer must be such that they ensure that enough demand is consumed so that any ILR bid is feasible. This compliance constraint may lead to inefficiency if the dispatch point cannot be accurately predicted. Assume that we have the utility function \( U(x) = a(Q^2 - (x - Q)^2) \), where \( a \) is some positive constant, \( x \) is the amount of consumption (MW) and \( Q \) is the maximum amount of consumption (MW) for the major consumer. We denote the event that the ILR is triggered by \( \xi = 1 \) and set \( \xi = 0 \), otherwise; furthermore, we set the probability of the ILR being triggered to be \( P(\xi = 1) = \rho \). Lastly, we define the marginal cost of interrupting load as \( \mu \). The profit of the consumer \( Z(\xi) \) is given below:

\[
Z(\xi) = \begin{cases} 
U(y^d) - \pi^d y^d + \pi^r y^r & \text{if } \xi = 0 \\
U(y^d - y^r) - \pi^d y^d + \pi^d y^r + \pi^r y^r - \mu y^r & \text{otherwise}.
\end{cases}
\]

Note that in \( Z(1) \) we have captured the cost of load interruption (\( \mu y^r \)). Now, given that the probability of the reserve being triggered is \( \rho \), we can compute the following expected profit:

\[
\mathbb{E}[Z] = (1 - \rho)U(y^d) + \rho(\pi^d y^r - \mu y^r + U(y^d - y^r)) - \pi^d y^d + \pi^r y^r. \quad (3.43)
\]
We can derive the marginal utility of consumption and the marginal cost of offering ILR, by taking the derivatives of the objective function. In the following equations, for sake of simplicity and without loss of generality we set $a = 1$.

\[
\frac{\partial Z}{\partial y^d} = (1 - \rho)U'(y^d) + \rho U'(y^d - y^r) - \pi^d = (1 - \rho)(-2y^d + 2Q) + 2\rho(y^d - y^r - Q) - \pi^d \tag{3.44}
\]

\[
\frac{\partial Z}{\partial y^r} = -\rho U'(y^d - y^r) + \rho(\pi^d - \mu) + \pi^r = -2\rho(y^d - y^r - Q) + \rho\pi^d - \rho\mu + \pi^r \tag{3.45}
\]

The marginal utility of the consumer that is derived in (3.44) is affected by the amount of cleared ILR multiplied by the probability of interruption. Similarly, the marginal cost of offering ILR depends on the level that the consumer is dispatched, and the probability $\rho$. In this section we lay out an example that probes the impact of implementing joint cost functions in the ILR offer of a major consumer.

Assume a single-node market that solves the co-optimized OPF problem that was described in Section 3.2. We start this example with the assumption that the major consumer’s bidding structure is able to reflect its joint value of energy and cost of ILR. Our aim is to compare the system welfare, when the consumer is able to coordinate its energy bids and reserve offers, versus the standard separate bidding and offering in [EROPF].

In this example, there are two generators, one who sells energy, and another who offers reserve. Furthermore, there exists one major consumer who buys electricity from the spot market, and also offers ILR. In addition, we assume an exogenous fixed demand ($d$) and reserve requirement ($r = d/5$) for this example. The opportunity cost for a major consumer is

\[
\text{Figure 3.2: Generation offer stack}
\]
consumer being called upon to interrupt its load is related to the probability of reserve being triggered, $\rho$.

The generators in our example submit the generation and reserve offer stacks that are depicted in Figures 3.2 and 3.3, where we have $T_g = \{1, 2, 3\}$, $T_{rg} = \{1, 2\}$. In addition, for the sake of simplicity and without loss of generality, we set the amount of uninterruptible load to be zero ($V = 0$).

For the sake of simplicity, and compatibility with the EROPF model, we assume a step-wise utility function for the major consumer, that yields the bid stack (outlined in Figure 3.4) with three tranches ($T_c = \{1, 2, 3\}$). Here, the consumer has a maximum demand of 50MW, and is able to offer ILR, but the costs depend on which tranche of consumption is being interrupted. Moreover, we assume that there is a disruption cost of $10 / \text{MWh}$ in addition to the loss in utility, if their supply is interrupted.

Let $x^c_t$ be the cleared consumption in tranche $t$. This cleared amount would represent the maximum curtailment at a given price. For example, if $x^c_1 = 10$ and the probability of reserve being called upon is $\rho = 0.1$, then the consumer could offer at most 10MW at price $(500 + 10) \rho = 51$. We implement this in model [J-EROPF] below.

\[\text{[J - EROPF] min } \sum_{t \in T_g} p^g_t x^g_t + \sum_{t \in T_{rg}} p^{rg}_t x^{rg}_t - \sum_{t \in T_c} p^c_t x^c_t + 51 x^c_1 + 16 x^c_2 + 7 x^c_3 \]
\[\text{s.t. } \sum_{t \in T_g} x^g_t = d + \sum_{t \in T_c} x^c_t \]

Figure 3.3: Generator’s reserve offer stack
Let us compute the dispatch for inelastic demand levels $d \in \{50, 60, 70, 80, 90, 100\}$. In Figure 3.5 we see how the optimal dispatched load, and ILR from the consumer adjusts as the rest-of-system demand changes. When $d$ is less than or equal to 70MW, the demand tranches with lower utility are dispatched, and the consumer’s load is high. This permits the dispatch of cheap ILR tranches from the consumer. Interestingly, the consumer’s load increases slightly as the rest-of-system demand increases from 50MW to 60MW; this is due to the system operator’s reserve requirement increasing from 10MW to 12MW. For $d$ greater than or equal to 80 we see that the consumer’s demand has dropped and the reserve is instead purchased from another participant in the market.

In order to numerically show the impacts of incorporating a joint cost function, we lay out the standard separated bid and offer system, in the [S-EROPF] model. In the [S-EROPF], we assume that the consumer must offer separate consumption bids and ILR offers to the market. In this case, in order to guarantee that the ILR offer made can be complied with, we must ensure that the minimum total consumption exceeds the total ILR offer made by the consumer (practically this can be achieved by bidding for this quantity at a high price). In order to construct a separated ILR offer that takes into account the information on the joint cost function, we need to assume a realization of demand, and compute the corresponding offer. For this example we will take the solution
from model [J-EROPF] with \( d = 75 \) (the middle point). For this realization of \( d \), by yielding the optimal dispatch values of [J-EROPF], we can determine the endogenous ILR availability of the strategic consumer; this consists of two reserve tranches. One is the cheaper tranche with 15MW of reserve at a price of $16; and the more expensive tranche with 10MW of reserve at price of $51. According to these values, we set our ILR offer parameters for [S-EROPF] model as \((q^1_{rc}, p^1_{rc}) = (10, 51)\) and \((q^2_{rc}, p^2_{rc}) = (15, 16)\).

In order to compare the joint and separate cost function models, we report on the optimal strategic consumption, ILR and the corresponding spot prices in Figure 3.1.
Table 3.1: Separate versus joint cost function results

<table>
<thead>
<tr>
<th>$d$</th>
<th>Load</th>
<th>ILR</th>
<th>$\pi^d$</th>
<th>$\pi^r$</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td>25</td>
<td>35</td>
<td>0</td>
<td>10</td>
</tr>
<tr>
<td>60</td>
<td>25</td>
<td>37</td>
<td>0</td>
<td>12</td>
</tr>
<tr>
<td>70</td>
<td>25</td>
<td>30</td>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td>80</td>
<td>20</td>
<td>20</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>90</td>
<td>10</td>
<td>16</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>100</td>
<td>10</td>
<td>18</td>
<td>8</td>
<td>8</td>
</tr>
</tbody>
</table>

Similar to the joint model, we compute the dispatch for a range of inelastic demand levels $d \in \{50, 60, 70, 80, 90, 100\}$ for [S-EROPF]. In Figure 3.6 we see how the optimal dispatched load, and ILR from the consumer adjusts as the rest-of-system demand changes.

**Figure 3.6:** Optimal cleared energy quantities for consumer’s offer and bid tranches in [S-EROPF]

Given that our separate ILR cost function is based on the middle $d = 75$MW point, for $d$ values that are higher or lower, we can observe inefficiency in the dispatch. For instance, for $d$ values that are lower than 70MW, the strategic consumer has no dispatched ILR value, because its ILR offer tranches are more expensive than other participant’s offer. On the other hand, when $d$ value is higher than 80MW, although the consumer is only dispatched for its first energy tranche, the optimal solution suggest the second tranche of the strategic consumer’s ILR offer must be dispatched. This reveals another type of inefficiency in the dispatch model, where the cost of interrupting the major consumer is underestimated. In Figure 3.7, we demonstrate the impact of the above observations on the system welfare. The system welfare in each model consists of the
major consumer’s payment\(^3\) minus the generation and reserve costs. We have also taken into account the payments that are incurred due to the strategic consumer inaccurately estimating its ILR costs.

\[ \text{Figure 3.7: Welfare under various demand realizations.} \]

In Figure 3.7, as expected, we observe that the welfare for model [J-EROPF] is never lower than that of model [S-EROPF]. In particular, for demand levels lower than 75MW, we see that the welfare in the [S-EROPF] is lower, due to higher reserve costs. Moreover, for demands higher than planned for \((d = 75)\), we also see that welfare is lower; much of this inefficiency is due to the consumer being dispatched ILR at a price below the cost of curtailment. Of course, the consumer could change their offers to avoid this outcome, but this would lead to inefficient market clearing outcomes at other demand levels. For example, an alternative strategy may be to offer no ILR, but this could result in less efficient outcomes.

### 3.5 Conclusions

In this chapter, we laid out a co-optimized OPF that represents the NZEM’s dispatch problem. In this model we incorporated both fixed and dispatchable demand, and the availability of ILR offering by large purchasers of electricity. We also assumed exogenous reserve requirements for each island of New Zealand. The EROPF model is the base in which we built our competitive equilibrium analysis, the multi-item auctions in this chapter.

We showed that for an energy and reserved co-optimized market, under certain conditions on truthful behavior of agents, the EROPF model is equivalent to the competitive

\[ ^{3}\text{Note that for sake of simplicity, we do not account for the welfare associated with the inelastic demand's payments. This is why in our example negative welfare is possible.} \]
equilibrium that is derived from individual self-interested agents’ profit maximization problems.

In Section 3.4, in a simplified model, we considered how multi-item auctions can lead to inefficient allocation if agents cannot adequately represent the joint-value function for the different products. This particularly causes issues when the products are strategic substitutes or complements, or are linked by constraints.

We showed that for large consumers purchasing energy and offering ILR, these products are strategic complements – one cannot offer ILR without a corresponding energy consumption. This, again, requires the agent to anticipate their cleared consumption, in order to appropriately offer reserve. Through our examples we can see that, failing to estimate the true dispatch point can lead to a lower total welfare, compared to the market structure allowing for contingent bids and offers.

This chapter was dedicated to probing the price-taking behavior and the resulting equilibrium. In addition, We laid out qualitative examples to gain intuition about such markets. However, in reality, it is nearly impossible to find a perfectly competitive market. In the competitive equilibrium we assume that each agent submits its true costs and utilities, but, in practice, these exact costs and value functions may not be able to be calculated. For instance, the hydro-generators may evaluate the water in their reservoirs differently from one another, and determining the the true value of water cannot be determined. On the other hand, the major consumers are often not willing to reveal the contribution of electricity consumption to their net profit from production. While the consumer will know its own approximate value of consuming electricity, regulatory authorities cannot discover this easily. Furthermore, the comprehensive joint utility/cost functions must be represented with multi-dimensional surfaces, and the resulting non-convexities may not be supported by the market.

In addition to these substantial limitations, the price-taking model does not allow consumers to represent their joint utility/cost functions for reserve and electricity in their bids and offers. Furthermore, for major consumers the price-taking assumption may not be reasonable since we would ignore the potential impacts of the change in load on the electricity prices. Also, markets like the NZEM (e.g. Singapore and many parts of Europe) are deregulated to the extent that the assumption of price-making major consumers more effectively models reality. Later, in Chapter 5, we review an existing stochastic price-making model BOOMER-consumer, which treats demand as fixed, and finds the optimal reserve stack; this avoids the issues with joint utility/cost functions, but does not capture the benefits of co-optimization that we have observed in this chapter. However, before we examine this, in the next chapter we will lay out our deterministic
price-making major consumer model, in which we use the [EROPF] model as the base of our bi-level optimization program.
Chapter 4

Price-making Major Consumer

Modeling demand-side bidding behavior can be considered from the perspective of a price taker or a price maker. In Chapter 3, we studied a model with price-taking agents resulting in a competitive equilibrium. The competitive model requires the assumption that each agent does not anticipate the effect that its actions have on the market prices. This contrasts with a model of a price-making major consumer, who recognizes how prices may change as a result of its actions. Explicitly modeling this impact may lead to higher profits, and also will make the model more realistic, especially in markets where bids and offers are not highly regulated. In this chapter we develop a single-period model for a price-making consumer and explore the interactions between energy consumption and offering ILR on prices.

4.1 Introduction

In deregulated electricity markets without strict monitoring of bids and offers, relatively large agents are able to increase their profit through the exercise of market power. Most of these markets have a few large generators who produce most of the total generation in that market. Such dominating generators are perceived as price-making agents who can manipulate prices. While the assumption of a strategic, price-making agent may not be a suitable model for strictly monitored markets, it is a reasonable model for imperfectly competitive markets (e.g. the NZEM, Singapore, and some European jurisdictions). With the introduction of demand-side participation, consumers can also impact the prices through their consumption decisions. For instance, a large manufacturer who typically purchases a significant amount of energy from the spot market can employ strategic behavior to alter energy and reserve prices in its favor by anticipating the impacts of its actions on the market.
Studying the strategic behavior of large generators in the electricity markets, has been the focus of many studies in the past few decades; in particular, in [98–105] among others, and recent literature surveys [106, 107]. However, most of the literature on demand response has been from either a market design perspective, or from the point of view of industrial consumers who face exogenous prices. There exist few models who study strategic consumers. In [108], Rassenti et al. study the economics of strategic bidding of energy consumers in a stylized model. In [109], Kazempour et al. model stochastic strategic bidding for a large consumer in an energy-only setting, over a 24-node network. Daraepour et al., in [110], extended the strategic bidding model that is presented in [109] to a network with high wind penetration, where the strategic consumer’s energy bid is co-optimized with a price-taking reserve offer, for a day-ahead market, which is solved for a small-scale network. Our work, however, models a novel comprehensive optimization program for a profit maximizing major consumer, who is a price-making agent in both energy and reserve markets. Our model is designed to be solved over a large-scale network, and the parameters of the model are derived from real-world historical data.

Large consumers have sizable enough loads so that reducing their consumption could have a significant impact on the market prices. Due to the hockey stick nature of electricity stacks (as we approach system capacity, the prices rise sharply), responding to price in high-price periods not only reduces the amount of expensive electricity purchased but also leads to a decrease in spot prices. As an aside, this demand response can potentially reduce generation from expensive and polluting peaking plants. Moreover, as illustrated in Section 1.5.2, major consumers who offer interruptible load can also alleviate any pressure on reserve that may in turn affect the price of electricity. Cleland et al. discuss this issue in detail in [111, 112].

In the NZEM, and several other jurisdictions (e.g. Singapore, Mid-continent ISO), energy and reserve are co-optimized, however the exact form of the co-optimized OPF problems differ. The concept of offering reserve at the same time as purchasing energy replicates the idea of purchasing electricity with a particular reliability standard. In energy and reserve co-optimized market the consumer pays for its electricity and receives some compensation for the reserve offered. The dispatchable load can also be considered a very unreliable source of energy, where the consumption varies so that the price paid never exceeds a target value. Consumers may be able to forecast the price of energy, this is quite different to the ILR, since these are physical events that occur randomly.

In the NZEM, the optimal dispatch ensures that both of the zones (the North and South Islands) have sufficient reserves procured against an \(N - 1\) contingency, wherein a large source of energy supply may trip (in either zone). Therefore, in this chapter,
we consider a large consumer not only capable of strategically reducing consumption, but also of offering ILR, in a co-optimized energy and reserve market. Cleland et al. [113] approach this problem through simulation and optimization over a discrete set of possible offer stacks. Our contribution is to model the problem comprehensively and consider all eligible demand and reserve supply levels.

A standard approach to capture the properties of price-making agents is through bi-level optimization. In the electricity market literature, there has been broad research on modeling the behavior of the price-making generators through bi-level optimization [6, 114, 115]. Our proposed model, however, captures the strategic behavior of a major consumer. We present a leader-follower type model captured as a bi-level optimization that maximizes the profit of the strategic consumer, while having the co-optimized OPF embedded inside as the lower level problem.

As discussed in Chapter 2, a common approach for solving a bi-level model is to reformulate it as an MPEC, using the equivalent KKT conditions. However, most standard MPEC solvers will not guarantee global optimality, due to non-convexity coming from the optimality conditions of the lower-level problem. Therefore the MPEC is reformulated into a MIP in order to find globally optimal solutions. This method has been broadly applied in the context of electricity market models. For instance, in [116], a bi-level optimization problem is laid out that maximizes the profit for a strategic producer. The bi-level model is first reformulated to an MPEC, which is subsequently transformed into a MIP. In [117], the same approach is used to reformulate the strategic generator’s bi-level program to a MIP, for a small network, using price-quota curves. In [118] the bi-level model is reformulated into an MPEC, where the complementarity conditions are modeled via disjunctive parameters that discretize the decision variables. In [115], Hobbs lays out an MPEC model for multiple strategic generators, which is solved deterministically over a 30-node network. Furthermore, in [119], a bi-level to MIP solution is utilized that presents optimal offering strategies by electricity producers in day-ahead energy markets with step-wise energy offer stacks, over a single period. Finally, in [38] a bi-level programming model is introduced where demand-response decisions are almost completely delegated to the Independent System Operator (ISO), who also estimates consumers’ baseline consumption levels.

The electricity market models that capture strategic behavior of generators, above, are either focused on an energy-only auction, or are developed for a small-scale network. Our model however, accounts for a co-optimized energy and reserve market. Given that in our model the strategic consumer is able to offer ILR, we can probe the impacts of this co-optimization on the consumer’s optimal actions, and the nodal prices. In addition, we implement our model for the full-scale NZEM model.
Chapter Structure

In Section 4.2, we lay out the profit-maximizing strategic consumer’s bi-level program, which embeds the NZEM’s co-optimized OPF, which was introduced in Chapter 3, as the lower-level problem. In Section 4.3, we use the KKT conditions of the dispatch problem, and binary variables with Big-M right-hand sides to enforce the complementary constraints. The resulting model is a MIP which we use to conduct an experiment to illustrate the impacts of strategic behavior and co-optimization of actions in Section 4.4. The example is set over a single-node network, but is designed to capture the interactions of reserve and energy in the market. Moreover, in Section 4.5, we extend our example to study the effects of introducing CfDs to the strategic consumer’s co-optimization model. Section 4.6 summarizes the chapter.

Specific Chapter Nomenclature

Parameters

\(f_{ij}\) \(\) The flow along arc \(ij\) (from node \(i\) to node \(j\)).

\(K_{ij}\) \(\) The line capacity in arc \(ij\).

\(r_e\) \(\) The reserve level required in island \(e\).

\(V_n\) \(\) The minimum amount of difference between ILR and consumption, at node \(n\); this is the level of consumption that can not be interrupted.

\(B_n\) \(\) The fraction of generation allowed to be offered as reserve, at node \(n\).

\(W_n\) \(\) The maximum total amount of generation and reserve, offered by the generator, at node \(n\).

Variables

\(x^z_i\) \(\) The variable associated with the dispatch quantity tranche type \(z\).

\(p^z_t\) \(\) The price of tranche \(t\) of type \(z\).

\(q^z_t\) \(\) The quantity of tranche \(t\) of type \(z\).

\(r_e\) \(\) The reserve level required in island \(e\).
Sets, Matrices and Indices

\( \mathcal{N} \) The set of all nodes in the network. Note that in our model we have one agent per node; this is not a restrictive assumption.

\( \mathcal{A} \) The set of all arcs in the network.

\( \mathcal{N}_e \) The set of all nodes in island \( e \).

\( L \) The loop constraint matrix, where \( L_{l,ij} \) corresponds to row \( l \) (associated with each loop) and the column \( ij \) corresponds to arc \( ij \).

\( T^n_e \) The set of interruptible consumption tranches.

\( e_n \) indicates the zone that node \( n \) is located in.

\( \mathcal{E} \) The set of zones, in New Zealand \( \mathcal{E} = \{N, S\} \), representing North and South Island, respectively.

\( Z \) The set for types of tranches, i.e. consumption, generation, ILR and reserve.

\( Z = \{c, g, rc, rg\} \)

\( T_z \) The set of all offered tranches of type \( z \).

\( T^n_z \) The set of all tranches of type \( z \) at node \( n \).

Duals

\( \left[ \right] \) Indicates the dual of its corresponding constraint.

## 4.2 Bi-level Optimization Formulation

We start by laying out the bi-level model for the major consumer in a co-optimized energy and reserve market, where there are zonal requirements for the reserve level. The upper level problem maximizes the major consumer’s profit, which includes the consumer’s utility of consumption plus the revenue of offering ILR minus the cost of purchasing energy. The lower-level problem is similar to [EROPF] that was discussed in the previous chapter, but is altered to incorporate the upper-level problem’s variables (the large consumer’s consumption and ILR offer). To present a general model, we do not restrict the major consumer to be located at a single node. We define the strategic nodes by allowing the strategic consumption bid \( y^d_n \) and \( y^r_n \) to be positive. Hence, we set the node all other non-strategic nodes’ \( C^d_n \) (maximum consumption) and \( C^r_n \) (maximum ILR) to be zero. In addition, we define the utility function for consumption of electricity \( U(x) = ux \) where \( u \) is a constant (i.e. we have a constant marginal utility). However our methods are applicable for any concave/convex utility/cost function. The objective is to maximize the profit, taking into account revenues obtained through ILR. The cost
of offering ILR only affects the major consumers if they are actually called upon to interrupt their load. We define the probability of being called upon to be $\rho$. Hence, the utility part of our objective function is: $u \sum_{n \in \mathcal{N}} (y^d_n - \rho y^r_n)$, where in practice $\rho \simeq 0$.

For the sake of simplicity (and generally without any effect on the optimal solution), we omit $u \rho y^r_n$ from the objective function. We lay out the bi-level optimization problem, that we call $[B-L]$, below:

\[
[B-L] \quad \max_{y^d, y^r} \sum_{n \in \mathcal{N}} \left( u y^d_n - \pi^d_n y^d_n + \pi^r_n y^r_n \right) \tag{4.1}
\]

s.t. $0 \leq y^d_n \leq C^d_n \quad \forall n \in \mathcal{N}$ \tag{4.2}

$0 \leq y^r_n \leq C^r_n \quad \forall n \in \mathcal{N}$ \tag{4.3}

$y^d_n - y^r_n \geq V_n \quad \forall n \in \mathcal{N}$ \tag{4.4}

$[\text{EROPF2}] \quad \max_{\{x^c \mid z \in \mathbb{Z}\}} \sum_{t \in \mathcal{T}^c} p^c_t x^c_t - \sum_{t \in \mathcal{T}^g} p^g_t x^g_t - \sum_{t \in \mathcal{T}^r} p^r_t x^r_t \tag{4.5}
\]

s.t. \hspace{1cm} (3.9) - (3.14) ([\text{EROPF}] \text{ constraints}) \hspace{1cm} \sum_{t \in \mathcal{T}^a} x^a_t + \sum_{i \mid n \in \mathcal{A}} f^a_{ni} - \sum_{i \mid n \in \mathcal{A}} f^a_{in} = \sum_{t \in \mathcal{T}^a} x^a_t - y^d_n - [\pi^d_n] \quad \forall n \in \mathcal{N} \tag{4.6}

\hspace{1cm} - \sum_{n \in \mathcal{N}_e \mid z \in \{rc, rg\}} \sum_{t \in \mathcal{T}^e} x^e_t = \sum_{n \in \mathcal{N}_e} y^r_n - r_e - [\pi^r_e] \quad \forall e \in \mathcal{E}. \tag{4.7}

Note that here the consumer also has a limit on its ILR offer, determined by their level of consumption and a load margin $V_n$ that can not be interrupted. We call the market clearing problem in this model $[\text{EROPF2}]$. The difference between this problem and the previous version ($[\text{EROPF}]$) is adding the upper-level variables $y^d$ and $y^r$ in (4.6) and (4.7) as inelastic demand and ILR for the lower-level problem; this is equivalent to bid energy and offer reserve at infinite and zero price, respectively, in order to ensure full dispatch. These two sets of variables are viewed as constants in the dispatch model $[\text{EROPF2}]$. In the next section we lay out the standard approach that is used to solve a bi-level problem, by replacing the lower-level problem with its optimality conditions, and subsequently employ a mixed-integer reformulation to find globally optimal solutions.

### 4.3 Standard MIP Reformulation

In this section we use the standard reformulation method to reformulate $[B-L]$ to a MIP. First we start by reformulating the lower-level problem, which is the dispatch problem in our model. Given that the dispatch model is convex, we are able to replace it by its set of KKT conditions and we can present the new formulation as an MPEC.
The consumer’s profit maximization problem includes the primal and dual feasibility constraints of [EROPF2] as well as complementarity conditions. We assume natural bounds on the primal and dual variables of the [EROPF2]. The prices are bounded by the value of lost load. We lay out the resulting MPEC problem below:

\[
[MPEC] \quad \max_{y^d, y^f} \sum_{n \in N} \left( uy_n^d - \pi_n^d y_n^d + \pi_n^e y_n^e \right) \\
\text{s.t.} \quad (3.30) - (3.42), (3.9) - (3.14), (4.2) - (4.7) \quad \text{from [EROPF]} \\
\text{KKTs} \quad \text{from [B-L]}
\]

In the next step we reformulate [MPEC] to a MIP. Following [120] we use Big-M upper bounds with binary variables \((s)\) to linearize the complementary constraints\(^1\). The Big-M parameters are given as \(U\) and \(F\) that are appropriately indexed based on each constraint, and are sufficiently large as upper bounds for the corresponding constraints. This formulation has a bi-linear objective function and linear constraints with binary variables, we call this model \([MIP]\):

\[
[MIP] \quad \max_{y^d, y^f} \sum_{n \in N} \left( uy_n^d - \pi_n^d y_n^d + \pi_n^e y_n^e \right) \\
\text{s.t.} \quad (3.9) - (3.14), (3.30) - (3.33), (3.39), (4.2) - (4.7), \\
\nu_l^+ \leq F_l^{(\nu^+)} s_l^{(\nu^+)} \quad \forall t \in T, \forall z \in Z \quad (4.8) \\
x_l^+ - q_l^+ \geq -U_l^{(\nu^+)} (1 - s_l^{(\nu^+)}) \quad \forall t \in T, \forall z \in Z \quad (4.9) \\
x_l^- \leq U_l^{(\nu^-)} (1 - s_l^{(\nu^-)}) \quad \forall t \in T, \forall z \in Z \quad (4.10) \\
\phi_n \leq F^{(\phi_n)} s^{(\phi_n)} \quad \forall n \in N \quad (4.12) \\
\sum_{t \in T_n^g} x_t^g - B_n \sum_{t \in T_n^g} x_t^g \geq -U^{(\phi_n)} (1 - s^{(\phi_n)}) \quad \forall n \in N \quad (4.13) \\
\phi_n^+ \leq F^{(\phi_n^+)} s^{(\phi_n^+)} \quad \forall n \in N \quad (4.14) \\
\sum_{t \in T_n^g} x_t^g + \sum_{t \in T_n^c} x_t^c - W_t \geq -U^{(\phi_n)} (1 - s^{(\phi_n)}) \quad \forall n \in N \quad (4.15) \\
\theta_n \leq F^{(\theta_n)} s^{(\theta_n)} \quad \forall n \in N \quad (4.16) \\
\sum_{t \in T_n^c} x_t^c - \sum_{t \in T_n^g} x_t^g + V_n \geq -U^{(\theta_n)} (1 - s^{(\theta_n)}) \quad \forall n \in N \quad (4.17) \\
\eta_{ij}^+ \leq F_{ij}^{(\eta^+)} s_{ij}^{(\eta^+)} \quad \forall i,j \in A \quad (4.18) \\
f_{ij} - K_{ij} \geq -U_{ij}^{(\eta^+)} (1 - s_{ij}^{(\eta^+)}) \quad \forall i,j \in A \quad (4.19)
\]

\(^1\)Depending on the properties of each problem, alternate methods may be used to reformulate an MPEC to MIP, for instance, in [38], SOS2 variables are used for reformulation instead of Big-M parameters. We have tested the use of this alternate method for our model, which resulted in an increase in the solution time. In Section 5.4.4 we will discuss this method in more detail.
η_{ij}^+ - η_{ij}^- \leq F(η^-) - s^-(η^-) \quad \forall ij \in \mathcal{A} (4.20)

η_{ij}^+ + K_{ij} \leq U(η^-)(1 - s^+(η^-)) \quad \forall ij \in \mathcal{A} (4.21)

η_{ij}^+, η_{ij}^- \geq 0 \quad \forall ij \in \mathcal{A} (4.22)

s^+(η^+), s^-(η^-) \in \{0, 1\} \quad \forall ij \in \mathcal{A} (4.23)

ν^+_t, ν^-_t \geq 0 \quad \forall t \in \mathcal{T}, \forall z \in \mathcal{Z} (4.24)

s^+(ν^+), s^-(ν^-) \in \{0, 1\} \quad \forall t \in \mathcal{T}, \forall z \in \mathcal{Z} (4.25)

s(θ_n), s(φ_n), s(φ'_n) \in \{0, 1\} \quad \forall n \in \mathcal{N} (4.26)

In order to solve [MIP] via a standard MIP solver, we must linearize our objective function which contains non-linear terms (i.e. \(ydπd\) and \(yrπr\)). In the next section we present a theorem that enables us to reformulate the objective function of [MIP] to a linear function.

### 4.3.1 Linearizing the Objective Function

In making decisions such as the consumption level for a major consumer, or the supply offer as a major generator, bi-level problems are formulated where the product of price and quantity is optimized over the set of equilibrium constraints stemming from the dispatch problem (OPF). For these to reflect the consumer’s (or producer’s) problem, some of the right-hand sides \(d_j, j \in \mathcal{J}\) will be considered as decision variables. We also define \(Q\) as a diagonal matrix with \(Q_{jj} = 1\) for \(j \in \mathcal{J}\), and 0 otherwise. Below we prove a simple theorem that the objective function of such bi-level problems can be reformulated as a linear function. The bi-level program of interest takes the form below:

\[
\text{[Bi-level]} \quad \min_{\pi(\mathbf{x}), d, d_j \in \mathcal{J}} \quad d^T Q \pi
\]

s.t. \quad \text{[OPF]} \quad \min_{\mathbf{x}} \quad c^T \mathbf{x}

s.t. \quad M \mathbf{x} = d \quad [\pi]

\quad A \mathbf{x} \leq b \quad [\nu]

\quad \mathbf{x} \geq 0.

The lower-level problem (OPF) is a linear program and can be solved using its KKT conditions. At optimality, the primal and dual linear program objectives coincide. That is \(c^T \mathbf{x} = d^T \pi + b^T \nu\). The MPEC of interest formed by the set of equilibrium constraints stemming from OPF is then of the form:

\[
\text{[MPEC]} \quad \min_{\mathbf{x}, \pi, \mathbf{d}, d_j \in \mathcal{J}} \quad d^T Q \pi
\]
s.t. \[ Mx = d, Ax \leq b, x \geq 0 \]
\[ MT\pi + AT\nu \geq c, \nu \geq 0 \]
\[ (b - Ax)^T \nu = 0 \] \( (4.27) \)
\[ (MT\pi + AT\nu - c)^T x = 0 \] \( (4.28) \)

**Theorem 4.1.** The objective of the problem [MPEC] above can equivalently be written as a linear function.

**Proof.** Note that \( d^T Q\pi = d^T \pi - \sum_{j \in J} d_j \pi_j \) and that \( \sum_{j \in J} d_j \pi_j \) is clearly linear since \( d_j \) are fixed parameters and not decision variables. Now from \( Mx = d \) and (4.28), we obtain that \( \pi^T d = (c - A^T \nu)^T x \). Using (4.27), we obtain that \( d^T \pi = c^T x - b^T \nu \) and the result is established. \( \square \)

For our particular problem [MIP] to linearize the objective function, we substitute \( y^d_n, y^r_n, \pi^d_n \) and \( \pi^r_n \) respectively with their equivalents in (4.6), (4.7) and (3.30) – (3.33). We also utilize the fact that \( \nu^+_{t_s}, \nu^-_{t_s}, \theta_n, \phi_n \) and \( \phi'_n \) are the dual variables for (3.10), (3.12), (3.13) and (3.14) respectively. Our [MIP] model (which is derived from the consumer’s MPEC model) satisfies the requirements of Theorem 4.1, hence we obtain a linear objective function:

\[
\sum_{n \in N} (u - \pi^d_n) y^d_n + y^r_n \pi^r_n = u \sum_{n \in N} y^d_n + \sum_{e \in \{S,N\}} c^e \pi^r_n + \sum_{t \in T_e} p^e_t x^c_t - \sum_{t \in T_q} p^q_t x^g_t
\]
\[
- \sum_{t \in T_g} p^g_t x^g_t - \sum_{t \in T_c} p^c_t x^c_t - \phi'_n W_n + \theta_n V_n
\]
\[
- \sum_{z \in Z} \sum_{t \in T_z} q^z_t \nu^+_{t_s} - \sum_{ij \in A} [\eta^+_{ij} + \eta^-_{ij}] K_{ij}.
\]

The linearized objective function enables us to solve [MIP] with standard MIP solvers, such as Gurobi[121] and CPLEX [122]. By implementing this model for a major consumer we can derive the optimal consumption and ILR level for the major consumer. In the next section, we utilize the MIP model to explore a simple example that illustrates the effects of strategic consumption and its co-optimization with the consumer’s ILR level.

### 4.4 Strategic Co-optimization Effects

In order to examine the effects of demand response and offering ILR in the electricity and reserve markets, we lay out a comprehensive example over a single-node network.
We use the [MIP] model where we only have a single node. Although, a single-node market does not model the network effects, namely price differences between nodes, it still captures the fundamental effects that a major consumer has on the prices of energy and reserve. At this node we have 3 agents: a generator who is able to submit an energy offer stack and a reserve offer stack; inelastic demand \((d)\); and a major consumer capable of demand response and offering ILR, as depicted in Figure 4.1.

For the sake of simplicity, we only apply one of the inverse bathtub constraints for the generator: (3.14), which enforces an upper-bound on the sum of energy generation and reserve. We assume that 10 MW of the inelastic demand \((d)\) can be offered as ILR at price of $300/MWh. Table 4.1 demonstrates the parameter values that we use in this example, while Table 4.2, gives the generation and reserve offer stacks\(^2\).

![Figure 4.1: The single-node network.](image)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>(u)</th>
<th>(W)</th>
<th>(V)</th>
<th>(r)</th>
<th>(C^d)</th>
<th>(C^r)</th>
<th>(T_g)</th>
<th>(T_{rg})</th>
<th>(d)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Value</td>
<td>190</td>
<td>255</td>
<td>0</td>
<td>97</td>
<td>100</td>
<td>100</td>
<td>{1,2,...,8}</td>
<td>{1,2,...,8}</td>
<td>65–130</td>
</tr>
</tbody>
</table>

Table 4.1: Parameter values of the single node example

In order to illustrate the impacts of strategic co-optimization of electricity and reserve we construct two versions of this example. First we assume that the major consumer is only allowed to submit its quantity of consumption \((y^d)\), and not allowed to offer in ILR; we call this version V1. In the second version the major consumer is able to offer a variable quantity of ILR \((y^r)\), as well as determining its consumption quantity; we call this version, V2. To present a comprehensive illustration, we apply changes to the amount of inelastic demand. We solve V1 and V2 for several levels of inelastic demand (from 65 to 130 MW with steps of 1 MW), and we plot the energy and reserve levels along with profits and prices.

\(^2\)In this example, each iteration of problem is assumed to be solved over a one hour period.
Figure 4.2 demonstrates the optimal profit (utility minus costs) of the large consumer from V1 and V2 experiments. It shows that the bi-level problem’s optimal objective value decreases as the inelastic demand increases. Also the figure depicts the difference in optimal objective values obtained from experiments V1 and V2 that, as anticipated, shows more profit in co-optimized version, V2.

Next we report on the optimal values of $y^d$ and $y^r$ for the two experiments. Note that, as shown in Figure 4.3, in V2, the ILR offer curve is always below the electricity consumption (dotted curve), indicating the consumer can not offer any more ILR ($y^r$) than the quantity consumed ($y^d$). The flat regions of the ILR offer curve show that, initially, the consumer withholds from offering reserve at full capacity to utilize the higher prices of reserve. We see the non-increasing strategic electricity consumption and ILR offer pattern until $d = 122$, where we observe a sudden increase in both these values. Note that, although more consumption results in higher electricity prices, this cost is offset by being able to offer more ILR. On the other hand, in the absence of the option to supply ILR, in V1, the curve indicating ($y^d$) decreases constantly to keep the total demand (major consumer plus the inelastic demand) constant.

### Table 4.2: Energy and reserve offer stack data.

<table>
<thead>
<tr>
<th>Tranche</th>
<th>Energy Stack</th>
<th>Reserve Stack</th>
<th>Interruptible Load</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$p_t^g$</td>
<td>$q_t^g$</td>
<td>$p_t^r$</td>
</tr>
<tr>
<td>1</td>
<td>17</td>
<td>25</td>
<td>13</td>
</tr>
<tr>
<td>2</td>
<td>29</td>
<td>15</td>
<td>39</td>
</tr>
<tr>
<td>3</td>
<td>51</td>
<td>18</td>
<td>57</td>
</tr>
<tr>
<td>4</td>
<td>76</td>
<td>16</td>
<td>67</td>
</tr>
<tr>
<td>5</td>
<td>90</td>
<td>22</td>
<td>98</td>
</tr>
<tr>
<td>6</td>
<td>119</td>
<td>20</td>
<td>110</td>
</tr>
<tr>
<td>7</td>
<td>126</td>
<td>25</td>
<td>129</td>
</tr>
<tr>
<td>8</td>
<td>200</td>
<td>90</td>
<td>200</td>
</tr>
</tbody>
</table>

**Figure 4.2:** Profit maximization objective function

**Figure 4.3:** Strategic consumer decision variables
The last set of results pertains to the clearing prices for energy $\pi^d$ for the experiments (illustrated in Figure 4.4) and reserve prices $\pi^r$, (illustrated in Figure 4.5). We observe a jump in electricity price in V2, which is a result of sudden increase in electricity consumption. For the major consumer, this rise in electricity cost is compensated for by being paid more for ILR. In V1, the price is constant. The major consumer is able to keep the price constant by decreasing its own demand to cancel out the increase in the inelastic demand.

Reserve price is also affected by ILR offers of the strategic agent. In V2, the reserve prices are at the most equal to the V1 reserve prices. Note that in V2, reserve prices are more competitive because of the participation of the major consumer in the reserve market. These results show that a major consumer who co-optimizes their energy bids and reserve offers strategically is able to improve their return. In the next section we extend this example to a co-optimized model with a major consumer that has a contract for difference, and probe how the strategic consumer’s behavior affects the market.

### 4.5 Strategic Consumer with Contracts for Differences

Due to the nature of electricity spot prices, they can either rise unpredictably or drop to much lower levels than may have been anticipated for significant periods of time. For large manufacturers that purchase energy on the spot market, the potentially large price fluctuations may drastically increase their energy consumption costs or may lead to high opportunity costs.

In addition to utilizing price-responsive demand policies in order to avoid potential high energy costs, the major consumers can also manage the risk of price volatility by signing hedge contracts. In this section we probe the effects of major consumers of electricity holding hedge contracts when they also engage in demand response.
Medium to large consumers of electricity, might seek to arrange an electricity supply contract linked to the wholesale electricity market prices. Such contracts may lead to more savings compared with the alternative of a fixed price contract. One of the most common forms of such hedge contracts is the CfD which is also known as swap. When a major price-making consumer with dispatchable demand purchases CfDs, its demand-responsive behavior may change in order to achieve a higher combined profit, from the CfD and the spot market.

In CfDs a contract seller provides a fixed volume of energy at a negotiated price for the length of the contract. Monetary transactions between the two parties ensure a fixed price throughout the contract period. If the spot price is higher than the negotiated price, then this consumer is paid the difference, whereas if the contract price is higher than the spot price, then the consumer has to pay the difference to the seller.

There has been a wide range of research on the effects of CfDs on the market, which shows their role in creating private information and the promotion of anti-competitive behavior in the market [102, 123–125]. In addition, there have been studies which viewed CfDs from the agents’ perspective. For instance, Allez and Vila discussed the efficiency of forward markets through laying out a two-generator Cournot model in [126]. Ruddell et al. consider an extension to this model, accounting for a price premium (above the spot price) in the forward market, due to purchasers anticipating the strategic behavior of producers [127]. Moreover, Philpott and Pettersen also presented a Cournot model in which a purchaser of electricity strategically bids in the day-ahead market [128].

In this section we use a CfD version of our [MIP] model to determine the optimal actions of the strategic major consumer, where the objective function is modified to incorporate the CfD payment scheme in the utility function of the consumer. The payments in CfDs are based on the fixed volume, stated in the contract, not the actual amount of consumption. For instance, a consumer with flexible demand can curtail its load in a high spot-price trading period to reduce its spot purchase costs, but still receive the CfD payment for the contract volume. In other words, the difference between the spot price and the contract price is profit for the amount of reduced load (below $y_c$). This profit is equal to $(\pi^d - \pi^c)y_c$, where $\pi^c$ and $y_c$ are the contract price and quantity, respectively, and $\pi^d$ is the cleared price at the strategic agent’s node. Therefore, when a CfD is in place for the strategic consumer, $z'$ is the new objective function for [MIP]. We call this model [C-MIP], and will use it show the impacts of CfDs in consumer’s profit and the nodal prices.

$$z' = U(y^d) - \pi^d y^d + \pi^c y^c + (\pi^d - \pi^c)y_c$$  (4.29)
At the first glance, load reduction at time periods with high spot price seems like a highly profitable strategy. However, a price-maker consumer may alleviate the price spike by lowering its consumption; this then stops them from receiving the anticipated high CfD payment. Hence, the optimal policy of a strategic consumer might sometimes suggest an increase in load, instead of load reduction for a high spot price. Therefore, we attempt to study the effects of implementing strategic behavior, when lowering the cost of spot purchases can lead to a decrease in profit that is gained through the CfD.

In the following example we utilize the co-optimized single-node setting that was laid out in the previous section (V2) but with the difference that we change the objective function of our MIP model to (4.29). We start with setting the contract price $\pi_c = 190$, and the contract quantity of $y^c = 40$. Figure 4.6 demonstrates the corresponding profit for the major consumer versus the level of inelastic demand $d$. In addition, in Figure 4.7 we depict the optimal decision variables ($y^d, y^r$) for the different levels of parameter $d$ in this example. As shown in Figure 4.7, the structure of CfD payments incentivizes higher load levels when compared with a no-contract setting (Figure 4.3). In addition, from Figure 4.6 we observe that when the total demand is higher, the consumer’s profit increases, which is an opposite effect to that of the no-contract model (Figure 4.2).

To further illustrate the impact of implementing CfDs, in Figures 4.8 and 4.9 we plot the corresponding price of energy and reserve for each level of $d$. As we observe in these figures, price of energy and reserve are much higher than the no-CfD case (depicted in Figures 4.4 and 4.5), as a consequence of higher consumption levels, when $d$ is high. This example shows that a CfD may decrease the incentive to lower the energy price by the strategic consumer.
4.6 Conclusions

In this chapter we directly addressed how major consumers whose demand for energy and ability to provide ILR can affect the market. Our model, given the offers and demand of other market participants can determine the optimal energy bids and ILR offers for a major consumer. Such models facilitate the short-run efficiency of the underlying electricity models as well as benefiting the consumer. One of the attributes of this model is to capture the link between electricity consumption and ILR offers, yielding insights for instance that a major consumer may choose to consume more electricity in order to benefit from further participation in reserve market. Also, by offering ILR, the consumers themselves cover the risks imposed on the system.

We laid out an illustrative example, in which a strategic consumer responds to a range of inelastic demands. This example presented insight on how a major consumer adjusts its actions according the changes in rest-of-market demand. We also addressed incorporating CfDs within a strategic consumer’s co-optimization problem and showed the impacts of CfDs on the market prices.

The model that we solved ([MIP]) in this chapter is a deterministic wait-and-see model and does not account for real time uncertainty associated within the energy markets. In order to address this issue, in Chapter 5, we lay out a stochastic model. Stochastic co-optimization of energy bids and reserve offers provides the strategic consumer with here-and-now decision making tools. However, when dealing with large-scale optimization problems, simultaneously optimizing both demand bid and ILR offer curves can lead to a large and intractable mathematical program. Therefore, in the next chapter we
present tailored reformulation method to enhance the performance of our mixed-integer stochastic optimization problem.
Chapter 5

Stochastic Price-making Major Consumer

In Chapter 4, we assumed we had full information about the electricity market, including information on bids and offers of other market participants. While a deterministic model is a useful starting point, in reality it is required to facilitate here-and-now decision making since we are exposed to uncertainty due to both fluctuations in demand and lack of information regarding other market agents’ utility/cost functions.

Most deregulated markets also allow major consumers, who have real-time flexibility in their load to utilize dispatchable demand. Dispatchable demand enables these consumers to submit bid curves, whereby their consumption will adapt to different realizations of the market clearing price. In addition, in the NZEM, large consumers are able to submit ILR offers, which allows them to cover their own risk on the market. Examining the efficient utilization of these two features of the market by the major consumers is one of the main goals of this thesis.

In this chapter, we model a major consumer who maximizes profit in a full-scale co-optimized energy and reserve market over a set of future price realizations. We start by making these decisions scenario dependent. Should these scenario-dependent quantities form a monotone bid (and also supply offer for reserves) then the resulting ex-post optimum stack is admissible to the market and, in fact, optimal in every scenario (and not just in expectation). Since this is not always the case, we use monotonicity constraints, to ensure that the optimal solution to this optimization problem renders admissible energy bid and ILR offer stacks. In practice, in order to retrieve optimal bids and offers that accurately address the future uncertainty, several scenarios need to be incorporated in the optimization problem. Each scenario comprises parameters for the full-scale network, yielding a large optimization program. Solving such a problem could become even more
cumbersome if the scenarios are coupled via linking constraints. We report on how to tackle the problem with enforced monotonicity constraints and the reformulations we used to enhance the solution method.

5.1 Introduction

Volatility of real-time demand realizations and incomplete information on other market participants’ cost and utility functions necessitate here-and-now decision making that takes into account the uncertainty within the agent’s dispatched quantity and price. For instance, generators account for uncertainty in real-time dispatch in their offer curves. These curves show their declared costs under different realizations of the residual demand curves. Given the common issue of incomplete available information at time of decision making, various models of generator offer-stack optimization exist. For instance, in [129], a stochastic model is presented that optimizes offering strategies where there is imperfect information among competing generators. In [130], a stochastic risk-constrained offering strategy is developed for a generation company in a price-based unit commitment problem with uncertain market prices. In [131] the strategic offering in electricity markets with step-wise functions is discussed with applications in China’s electricity market.

Recently, a few models have also been developed for the consumer’s strategic stochastic bidding problem. In [109] a complementarity bi-level model is used for modeling the stochastic strategic behavior of a large consumer. They used a heuristic method for decomposing the model to each scenario and decrease the solution time. In [110], a strategic demand response model with integrated wind-power is presented that provides day-ahead strategic co-optimization; their method is applied to a small network.

Solving the co-optimized major consumer model over a large-scale network has also been addressed in recent studies. In [113], Cleland et al. introduce a stochastic model in which they lay out a co-optimized electricity bid and reserve offer using simulation-optimization. This model is extended in [112], where the reduced version of the full-scale co-optimization problem is solved. In this model a limited set of consumption levels is considered and, for each consumption level, they produce both an optimal reserve offer stack and return an associated price distribution by using market simulation. This approach does not produce a demand bid curve and is restricted to limited quantities of consumption. Furthermore, the market simulation step is computationally expensive. (We further discuss this method and compare it with our proposed model in subsection 5.3.3.)
Our contribution to the stochastic co-optimized strategic bidding problem is that we derive an optimization model for a price-maker major consumer over a large-scale network where the consumer’s strategic energy bid curves and reserve offer curves are simultaneously co-optimized. We also introduce exact reformulation methods for enhancing the performance of our model over a full-scale model.

A market with dispatchable demand, or in other words, having the ability to submit bid curves to the market instead of a single-quantity load level, allows the consumers to show their willingness of being dispatched based on different levels of realized prices. (These result from the different types of realized trading periods). Given the convexity of the market clearing problem, such bid curves must be monotone decreasing. When enforcing the monotonicity constraints in the strategic consumer’s optimization problem, the scenarios will be linked; this could immensely increase the complexity of the problem. However, using the standard MPEC to MIP reformulation method may be useful for solving the model for small networks, but many studies have used different reformulation and heuristic schemes to enhance the performance of their solution methods. In Section 5.4, we introduce and implement several of the existing bi-level reformulation methods that are presented in [38, 132, 133]. However, we will show that our novel bi-parametric sensitivity analysis decomposition method performs best for the stochastic strategic co-optimization.

The ultimate goal of our model is to generate optimal stacks that are derived from historical data, and use them for the future trading periods, through submitting them to the co-optimized OPF, which renders the dispatched quantities and prices for the upcoming realized scenario. Hence, in the case-study that is presented in this chapter, we simulate the performance of the optimal stacks for out-of-sample scenarios, and report the profit attained. We also compare our simulated optimal policy with an inelastic consumption policy.

Chapter Structure

This chapter consists of four sections: the stochastic strategic consumer’s model standard formulation; illustrative examples showing the effects of stochastic co-optimization, MIP reformulation methods; and the NZEM case study.

Firstly, we lay out the stochastic bi-level program for a price-making consumer, in Section 5.2. Moreover, in Section 5.2.1, we introduce the monotonicity constraints that ensure that our model generates admissible bids and offers.
In Section 5.3, we probe the impacts of strategic co-optimization. In subsection 5.3.2, we lay out an illustrative example that compares the optimal demand bid curves considering both the presence and absence of incorporating reserve offers in the optimal decision. In addition, in subsection 5.3.3, we compare the performance of optimal policies that are derived from our proposed model with the BOOMER-Consumer method; this uses simulation optimization to generate optimal strategic reserve offers for major consumers. In addition, in subsection 5.3.4 we analyze the effects of incorporating stochasticity and monotonicity constraints on the solution time of our model.

We dedicate Section 5.4 to reformulating the stochastic MIP model, in order to be able to solve it over several number of scenarios and a real-world large-scale electricity market, in a reasonable time frame. We present reformulations that are based on existing and novel methodologies and compare their performance on our model.

In Section 5.5, we lay out our case-study on NZAS Tiwai point in the NZEM. We simulate our proposed stochastic policy over out-of-sample scenarios that are derived from historical data. We also compare the results of our policy with clairvoyant and fixed consumption methods. Finally, Section 5.6 summarizes the findings of this chapter.

### 5.2 Stochastic Co-optimization Formulation

Consider the single-scenario case, where in the absence of reserve and over a single-node market, the optimal consumption level is effectively singled out by the quantity that determines the dispatch on the aggregate market residual supply function. The seminal paper of Klemperer and Meyer [134] lays out the premise for using supply functions because these kinds of offers adapt better to uncertain environments that are faced with multiple scenarios.

A price-making agent with stochastic strategic behavior takes into account the uncertainty and the impacts of its decision under each realization of future. Therefore, the strategic action is to submit optimal admissible bids and/or offers that maximize the consumer’s expected profit. These bids (offers) must be monotone decreasing (increasing), each of which is fitted to the ex-ante optimal quantity-price pairs corresponding to the set of possible future realization, which we call the scenario set. Thereafter we extend our bi-level model [B-L] from Chapter 4 to a stochastic optimization problem where the upper-level problem maximizes the expected profit (over the scenario set) for a strategic consumer, subject to a lower-level problem which maximizes social welfare for each scenario.
In this stochastic optimization, the consumer faces a set of scenarios $\omega \in \Omega$, where the probability of each scenario $\omega$ occurring is $\rho_\omega$. These scenarios can, for instance, capture different levels of system demand, or different generation offers.

In the formulation of our stochastic MIP, we use similar notation to deterministic model in Chapter 4, with the additional super-script $\omega$ that denotes the parameters or variables belonging to the corresponding scenario $\omega$. Similar to the deterministic case, we allow strategic demand consumption by a firm spread over multiple nodes. Therefore, we define the strategic nodes by allowing $y_n^d\omega$ and $y_n^s\omega$ to be positive. Hence, we set the parameters $C_n^d\omega$ and $C_n^s\omega$ of all non-strategic nodes to be zero. Below we lay out the first part of the stochastic MIP, which includes the aggregation of individual scenarios’ bi-level problems. In section 5.2.1, we introduce the second part of the stochastic MIP that includes the monotonicity constraints which ensures admissibility of the optimal bid and offers. We denote the optimization problem consisting of these two parts [S-MIP]

\[
\text{Max. } \sum_{\omega \in \Omega} \rho_\omega \sum_{n \in \mathcal{N}} \left( r^e_n \pi^e_n + u^\omega_n y_n^d + \sum_{t \in \mathcal{T}_n^d \omega} p_i^e x_i^e + \sum_{t \in \mathcal{T}_n^s \omega} p_i^q x_i^q - \sum_{t \in \mathcal{T}_n^g \omega} p_i^g x_i^g \right)
- \sum_{t \in \mathcal{T}_n^r \omega} \sum_{i \in \mathcal{A}} \sum_{t \in \mathcal{T}_n^u \omega} q_{ij}^{\omega} \nu_{ij}^{\omega} - \sum_{i \in \mathcal{A}} \left[ \eta_{ij}^{\omega} + \eta_{ij}^{-\omega} \right] K_{ij}^{\omega} \right)
\]

s.t. \begin{align*}
0 \leq y_n^d \omega & \leq C_n^d \omega & \forall n \in \mathcal{N}, \forall \omega \in \Omega \\
0 \leq y_n^s \omega & \leq C_n^s \omega & \forall n \in \mathcal{N}, \forall \omega \in \Omega \\
y_n^d \omega - y_n^s \omega & \geq V_n^\omega & \forall n \in \mathcal{N}, \forall \omega \in \Omega \\
\sum_{t \in \mathcal{T}_n^r \omega} x_i^e + \sum_{i \in \mathcal{A}} f_n^e - \sum_{i \in \mathcal{A}} f_n^s & = \sum_{t \in \mathcal{T}_n^s \omega} x_t^g - y_n^d \omega & \forall n \in \mathcal{N}, \forall \omega \in \Omega \\
\sum_{i \in \mathcal{A}} t_{i,j}^e f_i^{\omega} & = 0 & \forall \omega \in \mathcal{A}, \forall \omega \in \Omega \\
- K_{ij}^{\omega} & \leq f_{ij}^{\omega} \leq K_{ij}^{\omega} & \forall i,j \in \mathcal{A}, \forall \omega \in \Omega \\
0 \leq x_t^r & \leq \underline{x}_t & \forall t \in \mathcal{T}_n^r, \forall \omega \in \Omega \\
\sum_{t \in \mathcal{T}_n^r \omega} x_t^r - \sum_{t \in \mathcal{T}_n^s \omega} x_t^s & \leq 0 & \forall \omega \in \mathcal{N}, \forall \omega \in \Omega \\
\sum_{t \in \mathcal{T}_n^g \omega} x_t^g & \leq p_n^g \sum_{t \in \mathcal{T}_n^s \omega} x_t^q & \forall \omega \in \mathcal{N}, \forall \omega \in \Omega
\end{align*}
The optimal solution of this problem consists of two stacks (the consumer’s demand-side bid and ILR offer). The admissible stacks are monotone step functions. In particular,
the bid stack must be decreasing and the ILR stack increasing. In the next subsection we introduce the monotonicity constraints to our model.

5.2.1 Admissible Bids and Offers

It is tempting to determine the optimal quantity of consumption for each scenario in isolation. The problem with this approach is that the sequence of these consumption decisions may not support a monotone curve. Fig. 5.1 demonstrates the 4 optimal points obtained by solving our model. Note that there is no single monotone decreasing demand curve that can pass through these 4 points, meaning that there does not exist an ex post optimal monotone curve.

To construct admissible bids and offers, we first add monotonicity constraints on the resulting bid and offer curves. We define binary variables \( \zeta_{ij}^d \) to ensure that the energy price in scenario \( i \) is higher than that of \( j \), provided the quantity consumed in scenario \( i \) is less than scenario \( j \). Similarly for the ILR offer, we define variables \( \zeta_{ij}^r \) that ensure monotonicity of the ILR offer stack. Once the problem is solved with the monotonicity constraints included, we obtain a monotone solution as depicted in Fig. 5.2.

\[
\begin{align*}
  y_{di} &\leq y_{dj}^j + M^d \zeta_{ij}^d & \forall i, j &\in \Omega, i \neq j \\
  y_{ri}^i &\leq y_{rj}^j + M^r \zeta_{ij}^r & \forall i, j &\in \Omega, i \neq j \\
  \pi_{di}^i &\geq \pi_{dj}^j - M^\pi \zeta_{ij}^d & \forall i, j &\in \Omega, i \neq j \\
  \pi_{ri}^i &\leq \pi_{rj}^j + M^\pi \zeta_{ij}^r & \forall i, j &\in \Omega, i \neq j \\
  \zeta_{ij}^d + \zeta_{ji}^d & = 1 & \forall i, j &\in \Omega, i \neq j \\
  \zeta_{ij}^r + \zeta_{ji}^r & = 1 & \forall i, j &\in \Omega, i \neq j \\
  \zeta_{ij}^d, \zeta_{ij}^r &\in \{0, 1\} & \forall i, j &\in \Omega, i \neq j
\end{align*}
\]
If we solve our model with these monotonicity constraints enforced, the optimal energy and reserve stacks can have up to $|\Omega|$ tranches. However, many markets allow for a maximum number of tranches in each energy or reserve stack; this number may be smaller than the number of scenarios that are included in our stochastic model. In order to address this issue, we can reformulate the model to force each consumer to submit an energy bid curve with $m^d$ tranches, and a reserve offer stack with $m^r$ tranches; we also define $k = |\Omega|$. In our current model $k = m^d = m^r$; however in what follows, we present monotonicity constraints for market in which there exists a limit on the number of tranches on admissible stacks.

![Figure 5.3: Monotone demand bid with $m = 3$ tranches](image)

To construct the bid stack we define $v^d_i (i \in \{1, 2, ..., m^d+1\})$ and $v^r_i (i \in \{1, 2, ..., m^r+1\})$ as the maximum quantity of energy/reserve in each tranche in the optimal energy/reserve stacks respectively. Similarly, we denote $h^d_i$ and $h^r_i$ as the price of each tranche in the optimal energy bid and reserve offer curve respectively. Figure 5.3 demonstrates an optimal bid stack with three tranches. In order to assign each scenario $\omega$’s corresponding quantity-price pair $(y^{d\omega}, \pi^{d\omega})$ to the optimal bid stack we use the following mapping method:

\[
\begin{align*}
  v^d_1 &\leq y^{d\omega} < v^d_2 \rightarrow \pi^{d\omega} = h^d_1 \\
  y^{d\omega} &= v^d_2 \rightarrow h^d_2 < \pi^{d\omega} < h^d_1 \\
  v^d_2 &< y^{d\omega} < v^d_3 \rightarrow \pi^{d\omega} = h^d_2 \\
  y^{d\omega} &= v^d_3 \rightarrow h^d_3 < \pi^{d\omega} < h^d_2 \\
  \vdots \\
  v^d_m &< y^{d\omega} < v^d_{m+1} \rightarrow \pi^{d\omega} = h^d_m \\
  y^{d\omega} &= v^d_{m+1} \rightarrow \pi^{d\omega} = h^d_m \\
  y^{d\omega} &= v^d_{m+1} \rightarrow \pi^{d\omega} \leq h^d_m < h^d_{m+1}.
\end{align*}
\]
Similarly, for the optimal reserve stacks we follow the above mapping method, with the difference that the reserve offers stack is set to be monotone increasing, instead of decreasing. In order to formulate this mapping scheme to a set of constraints we use binary variables \( z_{i,j}^{\omega}, z_{i,j}^{\omega'} \in \{0, 1\} \) where \( \omega \in \Omega \) and \( j \in \{1, 2\} \) (the index \( j \) indicates whether the point is on the horizontal or vertical part of the stack). Utilizing big-M parameters \( M \), we lay out the following monotonocity constraints:

\[
\begin{align*}
    v_i^d & \leq v_{i+1}^d & \forall i \in \{1, 2, \ldots, m^d\} & (5.44) \\
v_i^d & \leq h_{i+1}^d & \forall i \in \{1, 2, \ldots, m^d\} & (5.45) \\
v_i^d & = 0 & \forall i \in \{1, 2, \ldots, m^d\} & (5.46) \\
h_{m+1}^d & = 0 & & (5.47) \\
v_i^{\omega} & \leq v_{i+1}^{\omega} & \forall \omega \in \Omega & (5.48) \\
h_{m+1}^{\omega} & \leq \pi^{\omega} & \forall \omega \in \Omega & (5.49) \\
y_i^{\omega} & \geq v_i^d - M(1 - z_{i,j}^{\omega}) & \forall \omega \in \Omega, \forall i \in \{1, \ldots, m^d\} & (5.50) \\
y_i^{\omega} & \leq v_i^d + M(1 - z_{i,j}^{\omega}) & \forall \omega \in \Omega, \forall i \in \{1, \ldots, m^d\} & (5.51) \\
\pi^{\omega} & \leq h_i^d + M(1 - z_{i,j}^{\omega}) & \forall \omega \in \Omega, \forall i \in \{1, \ldots, m^d\} & (5.52) \\
\pi^{\omega} & \leq h_i^d - M(1 - z_{i,j}^{\omega}) & \forall \omega \in \Omega, \forall i \in \{1, \ldots, m^d\} & (5.53) \\
y_i^{\omega} & \geq v_i^d + M(1 - z_{i,j}^{\omega}) & \forall \omega \in \Omega, \forall i \in \{1, \ldots, m^d\} & (5.54) \\
y_i^{\omega} & \leq v_i^d - M(1 - z_{i,j}^{\omega}) & \forall \omega \in \Omega, \forall i \in \{1, \ldots, m^d\} & (5.55) \\
\pi^{\omega} & \leq h_i^d + M(1 - z_{i,j}^{\omega}) & \forall \omega \in \Omega, \forall i \in \{1, \ldots, m^d\} & (5.56) \\
\pi^{\omega} & \geq h_i^d - M(1 - z_{i,j}^{\omega}) & \forall \omega \in \Omega, \forall i \in \{1, \ldots, m^d\} & (5.57) \\
\sum_{i,j} z_{i,j}^{\omega} & = 1 & \forall \omega \in \Omega & (5.58) \\
v_i^r & \leq v_{i+1}^r & \forall i \in \{1, 2, \ldots, m^r\} & (5.59) \\
h_i^r & \leq h_{i+1}^r & \forall i \in \{1, 2, \ldots, m^r\} & (5.60) \\
v_i^r & = 0 & \forall i \in \{1, 2, \ldots, m^r\} & (5.61) \\
h_i^r & = 0 & \forall i \in \{1, 2, \ldots, m^r\} & (5.62) \\
v_i^{\omega} & \leq v_{i+1}^{\omega} & \forall \omega \in \Omega & (5.63) \\
h_i^{\omega} & \leq \pi^{\omega} & \forall \omega \in \Omega & (5.64) \\
y_i^{\omega} & \geq v_i^r - M(1 - z_{i,j}^{\omega}) & \forall \omega \in \Omega, \forall i \in \{1, \ldots, m^r\} & (5.65) \\
y_i^{\omega} & \leq v_i^r + M(1 - z_{i,j}^{\omega}) & \forall \omega \in \Omega, \forall i \in \{1, \ldots, m^r\} & (5.66) \\
\pi^{\omega} & \leq h_i^r + M(1 - z_{i,j}^{\omega}) & \forall \omega \in \Omega, \forall i \in \{1, \ldots, m^r\} & (5.67) \\
\pi^{\omega} & \geq h_i^r - M(1 - z_{i,j}^{\omega}) & \forall \omega \in \Omega, \forall i \in \{1, \ldots, m^r\} & (5.68) \\
y_i^{\omega} & \leq v_i^r + M(1 - z_{i,j}^{\omega}) & \forall \omega \in \Omega, \forall i \in \{1, \ldots, m^r\} & (5.69) \\
y_i^{\omega} & \geq v_i^r - M(1 - z_{i,j}^{\omega}) & \forall \omega \in \Omega, \forall i \in \{1, \ldots, m^r\} & (5.70)
\end{align*}
\]
\[
\pi^ω_i \geq h_i - M(1 - z^ω_{i,2}) \quad \forall \omega \in \Omega, \forall i \in \{1, \ldots, m^r\} \quad (5.71)
\]
\[
\pi^ω_i \leq h_{i+1} + M(1 - z^ω_{i,2}) \quad \forall \omega \in \Omega, \forall i \in \{1, \ldots, m^r\} \quad (5.72)
\]
\[
\sum_{i,j} z^ω_{i,j} = 1 \quad \forall \omega \in \Omega \quad (5.73)
\]

In order to incorporate the monotonicity constraints to our stochastic model, we can add constraints (5.37) – (5.43) to [S-MIP]. However, when solving our model for the NZEM, given the limit on maximum number of tranches of energy and reserve stacks, we add monotonicity constraints (5.44) – (5.73) to our [S-MIP] model. When solving the problem with multiple scenarios, the number of constraints and variables increase proportionally with the number of scenarios. In addition, the monotonicity constraints link the scenarios together meaning that decisions in one scenario may affect the other scenarios’ optimal solutions. This yields a large MIP, which can become computationally intractable with a large number of scenarios. Thus we consider reformulating the model to improve the solve times in Section 5.4.

Note that, although the simultaneous optimization accounts for all scenarios in the sample space, the structure of the optimal energy and ILR stacks do not allow for joint cost/utility functions (see Section 3.4). This is a shortcoming of the auction design which arises from the requirement of having two step-wise functions as offer and bid curves. Therefore, in order to be compatible with the market design, in this chapter we construct optimal bid and offer stacks in the form of step-wise functions.

### 5.3 Stochastic Co-optimization Effects

In this section we probe the effects of stochastic co-optimization by a strategic consumer. Recall the single-node example in Section 4.4, where we compared the wait-and-see decision making, with and without the ability of offering ILR. In this section we lay out a similar experiment, however, we explore the here-and-now co-optimization of dispatchable demand and ILR. In subsection 5.3.2, we present a stylized example over a 3-node network model. In this example we compare the optimal bid-curve of a major consumer both when the consumer chooses to offer ILR, and when the consumer is only participating in the energy market. In subsection 5.3.3, we lay out a small illustrative example to compare our bi-level co-optimization method with the BOOMER-consumer method, which is the most recent work on demand response in the NZEM [112, 113]. The BOOMER-consumer method has captured the effect of offering ILR by large consumers. However, this method does not allow for continuous consumption levels, and
it only optimizes the reserve stacks for each discrete demand level. By comparison, our method simultaneously optimizes demand and reserve stacks.

### 5.3.1 3-Node Network

In order to illustrate the impacts of co-optimized stochastic actions, we start by implementing [S-MIP] over a simple network. This network has one loop (impedance equals 1.0 for all arcs) with buses A, B and C. In this example we assume that there are two generators, one at bus B, and one at bus C. We also assume there is one inelastic consumer at bus C, and one strategic consumer at bus A. The line capacities are shown in Figure 6.6.

![Network Diagram](image)

**Figure 5.4:** The 4-node network.

The maximum amount of aggregated reserve and energy for generators B and C is 50 and 75 MW, respectively. Table 6.1 presents the generators’ offer stack data.

<table>
<thead>
<tr>
<th>Gen B Energy</th>
<th>Gen C Energy</th>
<th>Gen B Reserve</th>
<th>Gen C Reserve</th>
</tr>
</thead>
<tbody>
<tr>
<td>quantity</td>
<td>price</td>
<td>quantity</td>
<td>price</td>
</tr>
<tr>
<td>15</td>
<td>9</td>
<td>15</td>
<td>10</td>
</tr>
<tr>
<td>7</td>
<td>13</td>
<td>15</td>
<td>15</td>
</tr>
<tr>
<td>7</td>
<td>18</td>
<td>10</td>
<td>15</td>
</tr>
<tr>
<td>8</td>
<td>20</td>
<td>9</td>
<td>10</td>
</tr>
<tr>
<td>7</td>
<td>23</td>
<td>9</td>
<td>10</td>
</tr>
</tbody>
</table>

**Table 5.1:** Energy and reserve offer stack data.

In this example, we lay out a stochastic model which is solved over five scenarios (a–e). Here, we assume that uncertainty is only in the inelastic demand and reserve requirement, while the energy and reserve offer stacks are the same in all scenarios. Table 6.2 presents the values of demand and reserve parameters associated with each of the scenarios.
In order to maximize the expected profit for the major consumer that is located at bus A, we assume a constant marginal utility of $18 for this consumer and solve the [S-MIP] model. Given that this example has a small network, and only 5 scenarios, the solution time for the corresponding MIP is under one second, when solved with Gurobi 7.5 on an Intel Core i7-4770 CPU @ 3.40 GHz. However, the [S-MIP]'s solution time is much higher for full-scale and complicated systems; we address this issue in Section 5.4. This section, however, is dedicated to illustrating the impacts of stochastic strategic decision making. In subsections 5.3.2 and 5.3.3, we lay out two more examples based on this 3-node network.

5.3.2 Co-optimized Bid and ILR Curve Example

Here, we use the [S-MIP] model to construct two examples that probe the effects of stochastic co-optimization on optimal actions, for the major consumer over the illustrated 3-node network. First, we solve [S-MIP] when we allow the strategic consumer at bus A to submit ILR offer stacks as well as energy bid stacks. Second, we solve [S-MIP], when the strategic consumer only bids in energy market. Lastly, we simulate the performance of optimal bids and offers that are derived from these two examples, over out-of-sample scenarios, and compare the results.

Fig. 5.5 depicts optimal monotone consumption bids with and without co-optimization, as well as the optimal ILR offer stack, for the 5-scenario instance of our problem. The red dots correspond to consumption-price pairs of the monotone co-optimized model, and blue dots correspond to those of the model in absence of ILR. Finally, the black dots represent the reserve price-quantity pairs in the corresponding scenario. Note that these three sets of optimal price-quantity pairs are derived from the [S-MIP], with the additional monotonicity constraints, hence they form the corresponding optimal monotone offer- and bid-curves.

In this example, the ability to offer ILR enables the consumer to make additional revenue, leading to an expected profit of $221.78, compared to $149.66 when no ILR is offered.

### Table 5.2: Inelastic demand and reserve levels

<table>
<thead>
<tr>
<th>Scenarios</th>
<th>a</th>
<th>b</th>
<th>c</th>
<th>d</th>
<th>e</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bus C Inelastic Demand (MW)</td>
<td>0</td>
<td>13</td>
<td>31</td>
<td>49</td>
<td>67</td>
</tr>
<tr>
<td>Reserve Requirement (MW)</td>
<td>15</td>
<td>30</td>
<td>45</td>
<td>60</td>
<td>75</td>
</tr>
</tbody>
</table>
In addition, as shown in Figure 5.5, the strategic bid-stack in the two experiments are different, since for a given scenario the co-optimized bid has a higher load level in order to allow for more ILR procurement. Note that for the first tranche of the co-optimized bid the price is more than the marginal utility ($u = \$18$) because the incurred energy costs could be covered by the higher revenue of ILR.

In order to further assess the difference in profit when offering ILR, we simulate the optimal stacks of these two experiments for out-of-sample scenarios, where the inelastic demand level and reserve requirement are sampled from random variables with uniform distributions $U[0, 67]$ and $U[15, 75]$, respectively. We simulated the optimal stacks over 200 out-of-sample scenarios and report on the average profit values in Table 5.3.

As shown in Table 5.3, the average profit in the strategic bidding with ability to offer ILR results is over twice as much as the energy-only strategic behavior. Furthermore, the average demand in the co-optimized experiment, is 55% higher, in order to allow for more
ILR procurement. Even though this leads to slightly higher average electricity price, this is offset by the additional revenue from offering ILR. Finally, in the co-optimized experiment, the average reserve price is 19% lower than that of the energy-only bidding method; this is due to the participation of the strategic consumer in the reserve market in the co-optimized experiment.

The comparisons of the stochastic strategic bidding with and without ILR, are similar to the deterministic example in the previous chapter, showing that participation in the reserve market not only benefits the strategic consumer, but also lowers the reserve price which affects all the agents in that zone. In addition, in Figure 5.5, we observe that the consumer’s co-optimized energy bid allows for higher energy prices, compared to the energy-only bid. This is consistent with the results of the deterministic example in Section 4.4, where we showed that, when the strategic consumer is able to offer ILR, it is willing to consume more and consequently pay more for electricity.

In the next section, we will extend our study on stochastic co-optimization effects, by comparing the optimal actions of our method with a previous work on strategic consumers who operate in a co-optimized energy and reserve market.

5.3.3 BOOMER-Consumer Comparison

In this experiment we compare the most recent and relevant method in the existing literature on strategic consumers’ demand response with our bi-level co-optimization method. In [112, 113], Cleland et al. introduced a stochastic model called the BOOMER-consumer, in which they lay out a co-optimized electricity bid and reserve offer using simulation-optimization. They solve the problem of what fixed optimal quantity to consume and what optimal reserve stack (coupled with this consumption quantity) to offer. In contrast, we address the real problem of offering the optimal demand bid curve and reserve offer to the market.

We lay out our experiment, utilizing the same 3-node model that is described in subsection 5.3.1. Therefore, the optimal energy and ILR stacks for the [S-MIP], are the same as shown in Figure 5.5. On the other hand, in order to make the optimal reserve stack, the BOOMER method fixes the consumption level and calculates the optimal reserve stack over the given scenarios for the given consumption level. In this example, we solve the BOOMER model for 9 consumption levels (20–36). Table 5.4 shows the details of optimal solution values of reserve stacks for each consumption level as described in [112, 113], and Figure 5.6 shows the optimal reserve stacks that correspond to each designated fixed consumption level.
At this step, we compare the optimal actions of BOOMER method with our bi-level model's actions. Note that the reserve stacks in the BOOMER method are better informed as they are tailored for a particular consumption level. However, the reserve offer shown in 5.5 is optimized in expectation for various levels of consumption; this means it is a here-and-now decision rather than a wait-and-see approach. In order to complete this comparison we report on the difference of profit, given the two methods in Table 5.5.
The comparison on the expected profit shows that the bi-level method performs better on average and compared with each of the designated consumption levels. Note that the aim in both these methods is to produce optimal (in expectation) reserve stacks while taking into account the co-optimization of reserve and energy. However, strategic and simultaneous energy bidding and ILR offering results in higher profit for the strategic consumer than the reserve-only strategic behavior.

In this section, so far, we have probed the impacts of stochastic co-optimization on the optimal actions, and optimal expected profit. However, in the next section, we focus on the impacts of incorporating uncertainty on the solution time.

5.3.4 Stochastic Co-optimization Solution Time

Accounting for uncertainty in decision making results in higher expected profit on average, but on the other hand, the global optimal solution for a large stochastic model may not be found in a reasonable time frame. In order to study the effects of adding stochasticity on the solution time of our MIP model, we report on the performance of the 3-Node model over different levels of stochasticity. We construct our stochastic models by introducing a new set of scenarios in which, similar to Section 5.3.1, the uncertainty is assumed to be on the reserve requirement and inelastic demand level. We define these scenarios by randomly sampling uniformly distributed values of inelastic demand \((U[0, 70] \text{ MWh})\), and reserve requirement \((U[15, 75] \text{ MWh})\).

If we solved the [S-MIP] model without the monotonicity constraints embedded in the optimization problem, the model could be decomposed to each single scenario, with trivial solve time. However, the monotonicity constraints link the scenarios together, which has a significant effect on the solution time. To demonstrate this impact, we first solve the model without enforcing monotonicity constraints. Secondly, we solve the same model, with monotonicity constraints enforced. In Table 5.6, we report on results of the experiment for three different sample sizes 10, 20 and 30.

From Table 5.6 we can deduce that the non-monotone model is decomposed by the solver, given the number of variables is noticeably reduced via the pre-solve process. On the other hand, for the monotone model, we see that the solution time increases exponentially with the number of scenarios, to the extent that for the model with 30 scenarios, the problem cannot be solved to optimality in a 24-hour time frame.
Table 5.6: Non-monotone and monotone solution-time comparison

In this section we studied the benefits of strategic stochastic co-optimization, over a small network. However, our main goal is to study a price-making major consumer who purchases electricity from the New Zealand electricity spot market. Although results of here-and-now strategic decision making in the full-scale NZEM model may follow the same patterns as our 3-node example, the solution method may not scale for the full NZEM model. Therefore, in the next section, we present different solution methods that we used to decrease the solve-time and incorporate more scenarios in our sample set.

## 5.4 Reformulation Methods

Implementing decomposition and reformulation methods for large and complicated optimization programs can significantly reduce their solve times. However, there is no one method that would fit all types of large problems, thereby the specific features of each problem need to be taken into account. In this chapter, we use several methods to reformulate the [MIP] model. Our aim is to find the reformulation method that best enhances the solution time.

We first lay out the methodology of each reformulation scheme and then compare their performance. We implement our reformulation methods for the 3-node network that was described in subsection 5.3.1. Moreover, in Section 5.5.2, we will compare the performance of the best two methodologies when solved for the Tiwai point NZAS over the full NZEM.

For piece-wise, enhanced MIP, and disjunctive reformulations, for sake of simplicity, we use the deterministic form, given these formulations can easily be transformed to the stochastic version, similar to the [MIP] to [S-MIP] model. For the bi-parametric sensitivity analysis reformulation, however, we illustrate the implementation of the method in full detail.
5.4.1 Piece-wise Linear Reformulation

Here we present a reformulation method that utilizes the special nature of the supply offer and demand bid functions in electricity markets. In OPFs each offer or bid stack, consists of a number of price-quantity pairs that define the blocks of the stack and is therefore a piece-wise step function. We define these offer and bid stacks by two parametric functions called \( q(t) \) and \( p(t) \), using Special Ordered Set (SOS) type 2 variables [135]. Here the parameter \( t \) is the distance travelled from the origin along the offer stack in question. Due to the step-wise nature of the offer stacks, \( p(t) \) is constant when \( q(t) \) is changing and vice versa. Therefore, the function \( \Pi(t) = p(t) \times q(t) \), when \( q(t) \) and \( p(t) \) are obtained from an offer stack, is a piece-wise linear function. By using \( q(t) \), \( p(t) \), instead of using separate variables for determining the dispatched tranches, we can reduce the number of binary variables used in the KKTs of the dispatch model. This is a consequence of the piece-wise linearity of our objective function and the fact that by using SOS-2 variables a significant number of binary variables are eliminated; this approach was previously used in [132] for a similar type of bi-level program (a strategic generator model).

Below we outline the MIP reformulation using the parametrized version of demand, bid and reserve supply functions. Note that constraints addressing the tranche capacities along with their associated duals are also eliminated by introducing the parametrized supply and demand functions. For the sake of simplicity, we outline a deterministic model (i.e. a single-scenario model). Accordingly this can be extended to the stochastic version, by introducing an expected profit objective function over multiple scenarios, and monotonicity constraints as illustrated in subsection 5.2.1.

\[
\begin{align*}
\text{Max.} & \quad u \sum_{n \in \mathcal{N}} y^d_n - \sum_{n \in \mathcal{N}} \pi^d_n y^d_n + \sum_{e \in \{N,S\}} \sum_{n \in \mathcal{N}_e} \pi^r_e y^r_n \\
\text{s.t.} & \quad (4.2) - (4.4), (4.18) - (4.21) \\
& \quad q^e_n(t^e_n) + \sum_{i | ni \in \mathcal{A}} f_{ni} - \sum_{i | in \in \mathcal{A}} f_{in} + y^d_n = q^g_n(t^g_n) \quad \forall n \in \mathcal{N} \quad (5.74) \\
& \quad \sum_{n \in \mathcal{N}_e} \left( q^{rg}_n(t^{rg}_n) + q^{rc}_n(t^{rc}_n) + y^r_n \right) = r_e \quad \forall e \in \mathcal{E} \quad (5.75) \\
& \quad q^{rc}_n(t^{rc}_n) \leq q^e_n(t^e_n) - V_n \quad \forall n \in \mathcal{N} \quad (5.76) \\
& \quad q^{rg}_n(t^{rg}_n) \leq B_n q^g_n(t^g_n) \quad \forall n \in \mathcal{N} \quad (5.77) \\
& \quad q^{rg}_n(t^{rg}_n) + q^d_n(t^g_n) \leq W_n \quad \forall n \in \mathcal{N} \quad (5.78) \\
& \quad p^d_n(t^e_n) = \pi^d_n - \theta_n \quad \forall n \in \mathcal{N} \quad (5.79)
\end{align*}
\]
function which we denote (for later use) as $A$

Without loss of generality, we define $\hat{\pi}$ as a designated node in zone $e$, since the nodal reserve prices are equal in each zone. We replace $\pi_n^d$ with its equivalents in (5.79) and (5.80) and $\pi_n^e$ with its equivalents in (5.81) and (5.82). $\theta_n$, $\phi_n$ and $\phi'_n$ are the dual variables for (5.76), (5.77) and (5.78) respectively. When each of these dual variables is zero, the associated nonlinear part of the objective function will be zero. Furthermore, the dual variables are positive when their associated equation is binding, hence we obtain:

$$
\phi_n \left( q_n^g(t_n^g) - B_n \phi_n^g(t_n^g) \right) = 0,
$$

$$
\phi'_n \left( q_n^g(t_n^g) + q_n^g(t_n^g) \right) = W_n \phi'_n,
$$

$$
\theta_n \left( q_n^c(t_n^c) - q_n^c(t_n^c) \right) = -V_n \theta_n.
$$

With $p(t)$ and $q(t)$ defined as above, we can reformulate the bi-linear part of the objective function which we denote (for later use) as $A$:

$$
A = \sum_{n \in N} \pi_n^d y_n^d - \sum_{e \in \{N,S\}} \sum_{n \in N_e} \pi_n^e y_n^e
$$

$$
= \sum_{n \in N} \left( q_n^g(t_n^g) - q_n^c(t_n^c) - \sum_{i | j \in A} f_{ni} + \sum_{i | j \in A} f_{in} \right) (p_n^c(t_n^c) + \theta_n)
$$

$$
- \sum_{e \in \{S,N\}} r_e - \sum_{n \in N_e} q_n^g(t_n^g) - q_n^c(t_n^c) (p_n^c(t_n^c) + \theta_n).
$$

Without loss of generality, we define $\hat{n}_e$ as a designated node in zone $e$, since the nodal reserve prices are equal in each zone. We replace $\pi_n^d$ with its equivalents in (5.79) and (5.80) and $\pi_n^e$ with its equivalents in (5.81) and (5.82). $\theta_n$, $\phi_n$ and $\phi'_n$ are the dual variables for (5.76), (5.77) and (5.78) respectively. When each of these dual variables is zero, the associated nonlinear part of the objective function will be zero. Furthermore, the dual variables are positive when their associated equation is binding, hence we obtain:
\[ \Pi_g^n(t_g^n), \Pi_c^c(t_c^n), \Pi_r^g(t_r^n) \text{ and } \Pi_r^c(t_r^n) \] are linear functions. \( r_e, B_n, W_n \) and \( V_n \) are constant, thus \( A \) is linear:

\[
A = \sum_{e \in \{S, N\}} \sum_{n \in L_e} \left( \Pi_g(t_g^n) - \Pi_c(t_c^n) - p_r^g(t_r^n)r_e + \Pi_r^g(t_r^n) - p_r^c(t_r^n)r_e \right.
\]

\[
+ W_n \phi_n - V_n \theta_n \right) + \sum_{ij \in A} (\eta_{ij}^+ + \eta_{ij}^-) K_{ij}.
\]

Hence the complete objective function is equivalent to \( u \sum_{n \in N} y_n^d - A \). Using this linear objective function, we can solve the piece-wise linear reformulated model with SOS-2 variables. Furthermore, similar to \([S-MIP]\) method, we introduce uncertainty, and monotonicity constraints to this reformulation to enable here-and-now decision making.

### 5.4.2 Improved MIP Method

In this part we extend the same concept as for the piece-wise linear formulation, and utilize the step-wise nature of offer stacks to reformulate the model. Here we impose constraints that could tighten the linear program (LP) relaxation, thereby reducing the branch and bound time. The idea is to ensure that the duals on the tranche constraints comply with the increasing order of the tranches for each node. For instance, in a given stack, we could never see the highest priced tranche at its upper bound when lower priced tranches are not fully dispatched. To incorporate this intuition in the model, we impose additional constraints on the binary variables associated with complementary slackness of tranche capacities \( s_{\nu z}^{(\nu z^+)} + s_{\nu z}^{(\nu z^-)} \). The new improved MIP (we will refer to it as \([I-MIP]\)) will have the additional constraints below. An analogous pair of constraints are applied to the reserve stack.

\[
s_{\nu z}^{(\nu z^+)} \geq s_{\nu z}^{(\nu z^+)} + s_{\nu z}^{(\nu z^-)} \quad \forall t \in T_z^w, \forall n \in N, \forall z \in Z, \forall \omega \in \Omega,
\]

\[
s_{\nu z}^{(\nu z^-)} \leq s_{\nu z}^{(\nu z^-)} + s_{\nu z}^{(\nu z^+)} \quad \forall t \in T_z^w, \forall n \in N, \forall z \in Z, \forall \omega \in \Omega.
\]

### 5.4.3 Disjunctive Reformulation

One of the methods to reformulate the resulting MPEC from our bi-level problem, is to use the strong duality theorem, to replace the complementary constraints. However, using this method results in bi-linear terms in the objective function. In this part, following [133], we use a disjunctive approach to reformulate the bi-linear terms.

Here we lay out the model that utilizes the disjunctive method which enables us to reformulate the problem into a MIP with no complementarity constraints. This model
consists of the primal and dual feasibility and the strong duality conditions of the lower level problem, and the only nonlinear terms are $y_n^d \pi_n^d - y_n^r \pi_n^r$ which will be linearized using the disjunctive method below:

Max. $u \sum_{n \in \mathcal{N}} y_n^d - h_n^d + h_n^r$

s.t. $(4.2) - (4.4), (4.6), (4.7), (3.9) - (3.14), (3.15) - (3.19)$

\[
\sum_{n \in \mathcal{N}} h_n^d - h_n^r = \sum_{e \in \{N,S\}} r_e \pi^r_e + \sum_{i,j \in \mathcal{A}} (\eta_{ij}^+ + \eta_{ij}^-) K_{ij}
\]

\[
+ \sum_{z \in \mathcal{Z}} \sum_{n \in \mathcal{N}} \sum_{t \in \mathcal{T}_n} \nu_z^p p_z^r - \sum_{n \in \mathcal{N}} \sum_{t \in \mathcal{T}_n} x_t^e p_t^r
\]

\[
+ \sum_{n \in \mathcal{N}} \sum_{t \in \mathcal{T}_n} x_{t \in \mathcal{T}_n}^g p_t^g + \sum_{n \in \mathcal{N}} \sum_{t \in \mathcal{T}_n} x_{t \in \mathcal{T}_n}^r p_t^r
\]

\[
+ \sum_{n \in \mathcal{N}} \sum_{t \in \mathcal{T}_n} x_{t \in \mathcal{T}_n}^{rc} p^{rc}_t + \sum_{n \in \mathcal{N}} \phi_n^W W_n - \sum_{n \in \mathcal{N}} \theta_n V_n \quad \forall n \in \mathcal{N}
\]

\[
h_n^d = \sum_{a=p} P \sum_{b=0} 9 10^2 b \bar{\pi}_{n,a,b} \quad \forall n \in \mathcal{N}
\]

\[
y_n^d = \sum_{a=p} P \sum_{b=0} 9 10^2 b z_{n,a,b}^d \quad \forall n \in \mathcal{N}
\]

\[
h_n^r = \sum_{a=p} P \sum_{b=0} 9 10^2 b \bar{\pi}_{n,a,b} \quad \forall n \in \mathcal{N}
\]

\[
y_n^r = \sum_{a=p} P \sum_{b=0} 9 10^2 b z_{n,a,b}^r \quad \forall n \in \mathcal{N}
\]

\[
\sum_{b=0} 9 \pi_{n,a,b}^d = \pi_n^d \quad \forall a \in \mathcal{P}, \forall n \in \mathcal{N}
\]

\[
\sum_{b=0} 9 z_{n,a,b}^d = 1 \quad \forall a \in \mathcal{P}, \forall n \in \mathcal{N}
\]

\[
\sum_{b=0} 9 \bar{\pi}_{n,a,b}^r = \pi_n^r \quad \forall a \in \mathcal{P}, \forall n \in \mathcal{N}
\]

\[
\sum_{b=0} 9 z_{n,a,b}^r = 1 \quad \forall a \in \mathcal{P}, \forall n \in \mathcal{N}
\]

\[
z_{n,a,b}, \bar{z}_{n,a,b} \in \{0, 1\} \quad \forall a \in \mathcal{P}, \forall b \in \{0, 1, ..., 9\}, \forall n \in \mathcal{N}
\]

\[
\bar{\pi}_{n,a,b} \leq \pi_{n,a,b}^d \quad \forall a \in \mathcal{P}, \forall b \in \{0, 1, ..., 9\}, \forall n \in \mathcal{N}
\]

\[
\bar{\pi}_{n,a,b} \geq \pi_{n,a,b}^d \quad \forall a \in \mathcal{P}, \forall b \in \{0, 1, ..., 9\}, \forall n \in \mathcal{N}
\]

\[
\bar{\pi}_{n,a,b} \leq \pi_{n,a,b}^r \quad \forall a \in \mathcal{P}, \forall b \in \{0, 1, ..., 9\}, \forall n \in \mathcal{N}
\]

\[
\bar{\pi}_{n,a,b} \geq \pi_{n,a,b}^r \quad \forall a \in \mathcal{P}, \forall b \in \{0, 1, ..., 9\}, \forall n \in \mathcal{N}
\]

\[
\mathcal{P} = \{p, p + 1, ..., P\}
\]
5.4.4 Alternative Reformulations

We used a few other alternative reformulations for our bi-level problem that have been utilized in the literature. For instance, following [38], we defined KKTs by using SOS-2 variables, instead of Big-M parameters. Here we present this reformulation method, using a general LP format for the lower level problem as below:

\[
\begin{align*}
\min & \quad c^T x \\
\text{s.t.} & \quad Ax = b \\
& \quad x \leq x \leq \bar{x}
\end{align*}
\]

Therefore, the KKT conditions can be reformulated via this method as follows:

\[
\begin{align*}
& c^T - A^T y = v_1 - v_2 \\
& (v_1, v_2) \geq 0 \\
& \bar{x} \leq x \leq \bar{x} \\
& Ax = b \\
& x - \bar{x} \leq z_1(\bar{x} - \bar{x}) \\
& x - \bar{x} \geq z_2(\bar{x} - \bar{x}) \\
& \text{SOS2}(v_1, z_2, z_1, v_2) \\
& (z_1, z_2) \in \{0, 1\}.
\end{align*}
\]

Furthermore, in order to strengthen the LP relaxation, we used triangle inequality constraints, where redundant constraints are added on binary variables associated with monotonicity constraints. Here, we ensure that given three scenarios A, B and C, if A is before B and B is before C, then A must be located before C on the optimal bid curve. In another reformulation, we tried indicator variables, instead of using Big-M parameters to reformulate the complementarities. Lastly we attempted to directly solve the bi-linear program, given there are non-linear solvers that can handle bi-linear terms well (e.g. BARON [136]).

However, because the computational results demonstrated that these methodologies are not effective as our model becomes larger, we dispense with any further explanation and only report on the performance comparison of best three reformulations.
5.4.5 Bi-parametric Sensitivity Analysis Reformulation

In this section, we lay out the structure and the implementation of the last reformulation method that we used, which is also the most effective. The idea that we use in this method is to find reserve and energy prices as functions of the major consumer’s actions. Note that the lower level problem that we solve for each scenario might be large. For instance, the NZEM market consists of around 300 nodes, however, the only nodal prices that affect our upper-level decisions are those corresponding to the strategic consumer’s node. If we can find the energy and reserve prices as a function of the consumer’s actions \((\pi_d, \pi_r) = f(y^d, y^r)\), then each scenario can be reduced to a function instead of modelling all the details of the dispatch problem, thereby avoiding the formulation of a complex MIP.

5.4.5.1 General Format Algorithm

Firstly, we present the method to find the price functions using the fundamentals of the simplex method. We start with a generic form, and then apply the methodology for our specific problem. Suppose that we have a linear program \([LP]\) in standard computational form:

\[
\begin{align*}
\text{[LP]} \quad & \min \quad c^T x \\
\text{s.t.} \quad & Ax = b \quad [\pi] \\
& x \geq 0,
\end{align*}
\]

where we denote \(b_n\) as the \(n^{th}\) component of the right-hand side vector \(b\), and \(\pi\) as the optimal dual vector. We aim to find \((\pi_1, \pi_2)\) as a function of the right-hand sides \((b_1, b_2)\) \(^1\). In order to define this function, we use the following algorithm:

**Step 1: Set Initial Values** Set feasible initial values of \(b_1\) and \(b_2\) in \([LP]\). In this example we set \(b_1 = b_2 = 0\). Define set \(\mathcal{R} = \{}\) as the set of all regions. Define \(\mathcal{R}_i = \{}\) a subset of \(\mathcal{R}\), as the set of all extreme points in region \(i\). Set \(i = 1\).

**Step 2: Set Initial Optimal Basis** Retrieve the optimal basis for \([LP]\) given the values of \(b_1\) and \(b_2\). We store the basis data for the vector of basic variables \(x_B\). Note

\(^1\)This algorithm can be extended to additional parameters but depending on the problem the application may become much more complex.
that:

\[
\text{Given } Bx = b \implies x = B^{-1}b \quad \text{Since } x_B \geq 0 \implies B^{-1}b \geq 0.
\]

**Step 3: Define [S-LP]**

\[
\begin{align*}
[S-LP] \quad & \max_{b_1, b_2} c_1 b_1 + c_2 b_2 \\
\text{ s.t. } & B_{m,1}^{-1} b_1 + B_{m,2}^{-1} b_2 \geq 0 \quad \forall m \in \mathcal{M}
\end{align*}
\]

Here \(B_{m,1}^{-1}\) is the first column of the \(m^{\text{th}}\) row of \(B\) matrix, and \(\mathcal{M}\) is the set of all rows in \(B\) matrix.

**Step 4: Set initial c values** At the first iteration we set \(c_1 = 1\) and \(c_2 = -\infty\).

**Step 5: Solve [S-LP]** Solve [S-LP] and add the pair of optimal solution values \((b_1^*, b_2^*)\) to the set \(\mathcal{R}_i\).

**Step 6: Get objective coefficient sensitivity information** Using sensitivity analysis, find largest objective coefficient value \((c_2)\) at which the current optimal basis would remain optimal. Store it as \(c_2'\).

**Step 7: Change \(c_2\) coefficient**

- If \(c_2' \leq \infty\), set \(c_2 = c_2' + \epsilon\) in [S-LP] and go to step 5.
- If \(c_2' = \infty\) and \(c_1 = 1\), go to step 8.
- If \(c_2' = \infty\) and \(c_1 = -1\), go to step 9.

**Step 8: Change \(c_1\) coefficient**

- Set \(c_1 = -1\), and go to step 5.

**Step 9: Make seed points**

- Define seed set \(\mathcal{S} = \{\}\).
- Find the convex hall formed by the pairs in \(\mathcal{R}_i\).
- Find an exterior point for each edge, and add the pairs to $S$ as $(b^1_s, b^2_s)$.
- Set $i = i + 1$

**Step 10: Use seed points**

- If $S \neq \emptyset$, choose a pair $(b^1_s, b^2_s)$ from $S$. Set $b_1 = b^1_s$ and $b_2 = b^2_s$. Remove $(b^1_s, b^2_s)$ from $S$, and go to step 2.
- if $S = \emptyset$, go to step 11.

**Step 11: Make vertical regions** In order to allow for the optimal solution to lie on the vertical tranches of the residual stack, we add vertical regions to the set of regions $\mathcal{R}$ at this step.

1. Set $i = 1$, and $\mathcal{V} = \{\}$.

2. For each edge in $\mathcal{R}$, take the corresponding pairs of $(y^d, \pi^d_i)$ and $(y^r, \pi^r_i)$ for the two extreme points on the edge. Store them as point 1 and 2, (e.g. $y^d_1, y^d_2$).

3. Find the adjacent region $j$ to the chosen edge, and store $\pi^d_j$ and $\pi^r_j$.

4. Set $Q = \{(y^d_1, \pi^d_1), (y^r_1, \pi^r_1), (y^d_2, \pi^d_2), (y^r_2, \pi^r_2), (y^d_1, \pi^d_j), (y^r_1, \pi^r_j), (y^d_2, \pi^d_j), (y^r_2, \pi^r_j)\}$.

5. If $Q \notin \mathcal{V}$, add $Q$ to $\mathcal{V}$.

6. Set $i = i + 1$

7. If $i \leq |\mathcal{R}|$, go back to line 2.

8. If $i = |\mathcal{R}|$, then $\mathcal{R} \cup \mathcal{V}$ represents the surface that include all the regions.

### 5.4.5.2 Bi-parametric Reformulation Stochastic Application

In this section we discuss the implementation of the bi-parametric sensitivity analysis method for our bi-level problem. Here we use the presented decomposition method to compute the strategic node’s energy and reserve prices as a function of the consumer’s actions $(\pi^d, \pi^r) = f(y^d, y^r)$. We use the dispatch problem [EROPF2] (laid out in Section 4.2), and add 2 extra constraints (5.109) and (5.110) that denote the demand at the
Chapter 5 Stochastic Price-making Major Consumer

strategic node as $b_1$ and the residual reserve level as $b_2$. Hence, by utilizing the algorithm introduced above, we can calculate the price as a function of strategic consumer’s actions.

\[
y_d^n = b_1 \quad n \in \mathcal{N}^* \\
y_r^n = b_2 \quad n \in \mathcal{N}^* 
\]  

We implement this reformulation method on the market clearing problem of the 3-node network that is described in Section 5.3.1. Figures 5.7 and 5.8 demonstrate the decomposed regions, defined by the vertices obtained through our bi-parametric sensitivity analysis. Here regions are divided by black lines, with a top down view, where each demonstrated region has a unique corresponding reserve and energy price. However, when we have a degenerate solution at the edge of one region, the price of energy or reserve could be a convex combination of prices in the two adjacent regions. In order to address this, we have assigned “vertical regions” to the edges of regions. In addition, for our computational purposes, the algorithm that is laid out in subsection 5.4.5.1 is implemented in Gurobi/Python which is demonstrated in Appendix A.

![Figure 5.7: Energy price regions](image)

We utilize these generated regions to solve our stochastic model, we lay out a mixed integer program that reads in the regions corresponding to each scenario $\omega \ (\forall \omega \in \Omega)$ and outputs the optimal expected reserve and energy stacks over $\Omega$, that maximize expected profit for the major consumer. Assume that $R^\omega$ is the set of all regions retrieved from the algorithm described in Section 5.4.5.1, for scenario $\omega$, and $R^\omega_i$ as the set of all the extreme points that form region $i$ in $R^\omega$. In this model, each extreme point $j$, within region $i$, in set $\mathcal{R}^\omega$, is defined by the following four parameters: the corresponding
consumption level, reserve level, energy and reserve prices; these are denoted by $\hat{y}^d_{i,j}$, $\hat{y}^r_{i,j}$, $\hat{\pi}^d_{i,j}$ and $\hat{\pi}^r_{i,j}$ respectively. We call this model $[\alpha\text{-MIP}]$, as below:

$$\begin{align*}
[\alpha\text{-MIP}] & \quad \max_{y^d, y^r} \sum_{\omega \in \Omega} \left( u \ast y^d_{i,j} - \psi^d_{i,j} + \psi^r_{i,j} \right) \\
\text{s.t.} & \quad \psi^d_{i,j} = \sum_{i \in R^d} \sum_{j \in R^d} \alpha^d_{i,j} \hat{y}^d_{i,j} \hat{\pi}^d_{i,j} \quad \forall \omega \in \Omega \\
& \quad \psi^r_{i,j} = \sum_{i \in R^d} \sum_{j \in R^d} \alpha^r_{i,j} \hat{y}^r_{i,j} \hat{\pi}^r_{i,j} \quad \forall \omega \in \Omega \\
& \quad y^d_{i,j} = \sum_{i \in R^d} \sum_{j \in R^d} \alpha^d_{i,j} \hat{y}^d_{i,j} \quad \forall \omega \in \Omega \\
& \quad y^r_{i,j} = \sum_{i \in R^d} \sum_{j \in R^d} \alpha^r_{i,j} \hat{y}^r_{i,j} \quad \forall \omega \in \Omega \\
& \quad \pi^d_{i,j} = \sum_{i \in R^d} \sum_{j \in R^d} \alpha^d_{i,j} \hat{\pi}^d_{i,j} \quad \forall \omega \in \Omega \\
& \quad \pi^r_{i,j} = \sum_{i \in R^d} \sum_{j \in R^d} \alpha^r_{i,j} \hat{\pi}^r_{i,j} \quad \forall \omega \in \Omega \\
& \quad \sum_{j \in R^d} \alpha^d_{i,j} = z^d_{i,j} \quad \forall i \in R^d, \forall \omega \in \Omega \\
& \quad \sum_{i \in R^d} z^d_{i,j} = 1 \quad \forall \omega \in \Omega \\
& \quad 0 \leq \alpha^d_{i,j} \leq 1 \quad \forall j \in R^d, \forall i \in R^d, \forall \omega \in \Omega \\
& \quad z^d_{i,j} = \{0, 1\} \quad \forall i \in R^d, \forall \omega \in \Omega
\end{align*}$$
In $[\alpha\text{-MIP}]$, $y^d_\omega$, $y^r_\omega$, $\pi^d_\omega$ and $\pi^r_\omega$ are the variables corresponding to strategic consumption, ILR, energy price and reserve price, respectively, for each scenario $\omega$. Also, we define $\psi^d_\omega$ and $\psi^r_\omega$ as the variables corresponding to cost of consumption $(y^d_\omega \pi^d_\omega)$ and revenue of ILR $(y^r_\omega \pi^r_\omega)$ in scenario $\omega$ respectively. In this formulation we use binary variable $z^\omega_i$ to determine which region to choose in scenario $\omega$. We called this mathematical program $[\alpha\text{-MIP}]$ due to the use of $\alpha$ variables that define the convex hull representing each region. By adding the monotonicity constraints (5.37) – (5.43), we ensure that the output of this model consists of two monotone optimal stacks (the consumer’s demand-side bid and ILR offer), that are admissible and optimal in expectation over all scenarios. See Appendix A for the implementation of $[\alpha\text{-MIP}]$ for our case study in Gurobi/Python.

5.4.6 Reformulation Methods Comparison

In this part we compare the performance reformulation methods, when solving the model for the strategic major consumer in the 3-node network (described in subsection 5.3.1). For the sake of simplicity we only compare the standard stochastic MIP formulation with the other two best performing reformulation methods.

As shown in Table 5.7, the bi-parametric policy is the most effective reformulation method. In addition, we observe that although the improved MIP model is effective in reducing solve-time, it does not scale well with the size of the problem (as observed for the model with 30 scenarios). Moreover, in order to verify that the bi-parametric sensitivity analysis performs better than other reformulations when solved for larger and more complex methods, in Section 5.5.2, we will compare the performance of the improved MIP and bi-parametric method, implemented for the full-scale NZEM model.

5.5 Case Study

In this section we focus on implementing our model with the most effective reformulation method for the NZAS, a major consumer of electricity. Furthermore, to address uncertainty, we introduce multiple scenarios and optimize the expected profit over a sample set. In order to construct the sample set of scenarios, we use historical data which
<table>
<thead>
<tr>
<th></th>
<th>Continuous variables</th>
<th>Integer variables</th>
<th>Solve time(s)</th>
<th>Iterations</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Model</td>
<td>Pre-solved</td>
<td>Model</td>
<td>Pre-solved</td>
</tr>
<tr>
<td>(15 Scen)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[S-MIP]</td>
<td>1860</td>
<td>1260</td>
<td>1830</td>
<td>1155</td>
</tr>
<tr>
<td>Improved [S-MIP]</td>
<td>1860</td>
<td>838</td>
<td>1830</td>
<td>778</td>
</tr>
<tr>
<td>Bi-parametric</td>
<td>7137</td>
<td>7047</td>
<td>2183</td>
<td>1958</td>
</tr>
<tr>
<td>(20 Scen)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[S-MIP]</td>
<td>2480</td>
<td>1680</td>
<td>2640</td>
<td>1640</td>
</tr>
<tr>
<td>Improved [S-MIP]</td>
<td>2480</td>
<td>1132</td>
<td>2640</td>
<td>1152</td>
</tr>
<tr>
<td>Bi-parametric</td>
<td>9824</td>
<td>9729</td>
<td>3187</td>
<td>2787</td>
</tr>
<tr>
<td>(25 Scen)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[S-MIP]</td>
<td>3100</td>
<td>2100</td>
<td>3550</td>
<td>2175</td>
</tr>
<tr>
<td>Improved [S-MIP]</td>
<td>3100</td>
<td>1436</td>
<td>3550</td>
<td>1586</td>
</tr>
<tr>
<td>Bi-parametric</td>
<td>12289</td>
<td>12189</td>
<td>4235</td>
<td>3610</td>
</tr>
<tr>
<td>(27 Scen)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[S-MIP]</td>
<td>3348</td>
<td>2268</td>
<td>3942</td>
<td>2403</td>
</tr>
<tr>
<td>Improved [S-MIP]</td>
<td>3348</td>
<td>1833</td>
<td>3942</td>
<td>2049</td>
</tr>
<tr>
<td>Bi-parametric</td>
<td>13239</td>
<td>13136</td>
<td>4674</td>
<td>3945</td>
</tr>
<tr>
<td>(30 Scen)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[S-MIP]</td>
<td>3720</td>
<td>2520</td>
<td>4560</td>
<td>2760</td>
</tr>
<tr>
<td>Improved [S-MIP]</td>
<td>3720</td>
<td>2043</td>
<td>4560</td>
<td>2373</td>
</tr>
<tr>
<td>Bi-parametric</td>
<td>14518</td>
<td>14412</td>
<td>5326</td>
<td>4426</td>
</tr>
</tbody>
</table>

Table 5.7: Reformulation comparisons

is publicly available online at the New Zealand Electricity Authority’s EMI repository [137].

In order to compute the optimal policies (stacks), we need to choose a sample of scenarios (Ω) from the scenario space. Bid and offer stacks are built to result in here-and-now actions for different types of time-periods and, by finding the right combination of scenarios significantly improves the performance of the policies. If we build our scenario set from many similar time-periods that are more likely to happen in the future, we miss the outlier scenarios which could be incorporated in the bid stack but without altering the optimal response to the more frequent scenarios. Hence, we avoid picking very similar scenarios in our scenario set (e.g. two consecutive time-periods in one day). This, not only enhances the solve time, by reducing the symmetry in the MIP (and the resulting fractionality), but also enables the major consumer to optimize for various types of time periods, which will form optimal stacks with more tranches. In subsection 5.5.1, we provide two examples that develop intuition for the choice of diversity in sample selection.
5.5.1 Scenario Selection Examples

In order to illustrate our choice of a scenario set regarding the similarity of scenarios, we start this section by laying out an example in which we use the full-scale network’s historical data. Suppose $\Omega_1$ and $\Omega_2$ are two scenario sets, where $|\Omega_1| = |\Omega_2| = 6$. For $\Omega_1$ we sample similar scenarios, all picked from a morning peak on a single day, in winter 2016. On the other hand, for $\Omega_2$, we choose semi-similar scenarios from morning peaks of two days in winter 2016 and 2017. We solve $[\alpha\text{-MIP}]$ for both sets and obtain the optimal expected actions for each scenario set. Figure 5.9 shows the optimal stacks for the two sample sets, here $S_1$ and $S_2$ correspond to optimal bid stacks from optimizing against $\Omega_1$ and $\Omega_2$ respectively.

![Figure 5.9: Sample set comparison](image)

As shown in figure 5.9, $S_2$ consists of three steps, which enables it to respond to different types of time periods. The three mutual scenarios in the two scenario sets, have the same optimal consumption - price pairs in the two stacks (the overlapped points), which results in equal in-sample profit for the mutual points. In addition, the diversity of scenarios in $\Omega_2$, enhances the performance of $S_2$ for out-of-sample scenarios. Table 5.8 shows each stack’s profit when simulated over 100 out-of-sample scenarios. Note that the profit is computed over the length of one trading period is half-hour.

<table>
<thead>
<tr>
<th>Stack</th>
<th>Average</th>
<th>Min.</th>
<th>Max.</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1</td>
<td>7853</td>
<td>5054</td>
<td>10785</td>
</tr>
<tr>
<td>S2</td>
<td>8920</td>
<td>6119</td>
<td>10818</td>
</tr>
</tbody>
</table>

![Figure 5.10: Similar scenarios](image)

Table 5.8: Out-of-sample performance comparison

The example illustrated above, motivated us to further explore the impact of the scenario set on the quality of the optimal solution. Here, we use a generic example to
parametrically calculate the impact of scenario selection. Suppose we have scenario $\omega$, with the residual supply stack shown with black line in Figure 5.10 and with the optimal consumption-price pair $(q, \pi)$. We introduce new scenarios (a–d) where their residual stack deviates from that of $\omega$ with distance $\delta$ in each dimension. For the sake of simplicity we only show 2 out of 4 stacks (a and c) in Fig. 5.10. Here scenario a’s residual stack is shown in red, and its optimal consumption-price pair is $(q - \delta, \pi - \delta)$ and that of scenario c is $(q + \delta, \pi + \delta)$. Similarly, scenario b and d have the optimal consumption-price pairs $(q - \delta, \pi + \delta)$ and $(q + \delta, \pi - \delta)$, respectively. If we solve [\(\alpha\)-MIP] for $\Omega = \{\omega\}$, the optimal stack will have one step with the point $(q, \pi)$. However, in order to cover the optimal consumption for the scenarios whose prices are higher with $\delta$ divergence from $\pi$, we draw the optimal stack based on the quantity-price pair $(q, \pi + \delta)$ which is presented with the dashed gray line in Figure 5.10. We call this stack $\omega$-stack.

In Table 5.9 we compare using the clairvoyant policy versus the optimal stack generated with $\Omega$, for all scenarios $\omega$ and a through d. Here the difference between the clairvoyant policy and $\omega$-stack, depends on $\delta$. Note that when we set $\delta << q$, the difference in profit becomes very small, and it would be beneficial to pick a representative scenario in $\Omega$, instead of including all similar scenarios.

<table>
<thead>
<tr>
<th>Scenario</th>
<th>Clairvoyant</th>
<th>$\omega$-stack</th>
<th>Difference</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\omega$</td>
<td>$(u - \pi)q$</td>
<td>$(u - \pi - \delta)q$</td>
<td>$\delta q$</td>
</tr>
<tr>
<td>a</td>
<td>$(u - \pi + \delta)(q - \delta)$</td>
<td>$(u - \pi - \delta)(q - \delta)$</td>
<td>$2\delta(q - \delta)$</td>
</tr>
<tr>
<td>b</td>
<td>$(u - \pi - \delta)(q - \delta)$</td>
<td>$(u - \pi - \delta)(q - \delta)$</td>
<td>0</td>
</tr>
<tr>
<td>c</td>
<td>$(u - \pi - \delta)(q + \delta)$</td>
<td>$(u - \pi - \delta)q$</td>
<td>$\delta(u - \pi - \delta)$</td>
</tr>
<tr>
<td>d</td>
<td>$(u - \pi + \delta)(q + \delta)$</td>
<td>$(u - \pi + \delta)q$</td>
<td>$\delta(u - \pi + \delta)$</td>
</tr>
</tbody>
</table>

Table 5.9: Profit vs policy

The example above shows that the more scenarios are similar to one another, the less the objective function will improve by including them in the optimal bid stack. Hence, we construct our random sets $\Omega$ while ensuring two similar scenarios are not picked. However, the information on the number of occurrences of similar scenarios will be incorporated into the probability of their representative scenario.

### 5.5.2 Full-Scale Model Solution Time

In this part, we briefly compare the performance of the best two proposed reformulation methods ([I-MIP] and [\(\alpha\)-MIP]) when implemented over the NZAS data. Table 5.10 presents the average optimality gap for [I-MIP] and [\(\alpha\)-MIP], after one hour of solve
time. Similarly, Table 5.11 compares the number of integer variables in each method, given the sample size. As shown in these tables, the $[\alpha$-MIP] method has computational advantage and allows us to incorporate larger number of scenarios in our model, hence we use this model for our full-scale NZAS case study.

<table>
<thead>
<tr>
<th>Method</th>
<th>4</th>
<th>6</th>
<th>8</th>
<th>10</th>
<th>12</th>
<th>14</th>
<th>16</th>
<th>18</th>
<th>20</th>
<th>22</th>
</tr>
</thead>
<tbody>
<tr>
<td>I-MIP</td>
<td>0</td>
<td>12.01</td>
<td>33.61</td>
<td>86.32</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>$\alpha$-MIP</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.8</td>
<td>4.9</td>
<td></td>
</tr>
</tbody>
</table>

Table 5.10: Average bound gaps (%) versus number of scenarios.

<table>
<thead>
<tr>
<th>Method</th>
<th>4</th>
<th>6</th>
<th>8</th>
<th>10</th>
<th>12</th>
<th>14</th>
<th>16</th>
<th>18</th>
<th>20</th>
<th>22</th>
</tr>
</thead>
<tbody>
<tr>
<td>I-MIP</td>
<td>10504</td>
<td>16961</td>
<td>21078</td>
<td>26388</td>
<td>31710</td>
<td>37048</td>
<td>42350</td>
<td>47648</td>
<td>52988</td>
<td>58360</td>
</tr>
<tr>
<td>$\alpha$-MIP</td>
<td>1101</td>
<td>1838</td>
<td>2392</td>
<td>3342</td>
<td>3744</td>
<td>3991</td>
<td>4677</td>
<td>5237</td>
<td>5924</td>
<td>6332</td>
</tr>
</tbody>
</table>

Table 5.11: Number of integers versus number of scenarios.

### 5.5.3 Stochastic Strategic Policy Simulation

Subsequent to choosing the sample set, and the reformulation method, we move on to test and analyze the performance of our proposed policy. We start with simulating our stochastic strategic co-optimization method for the winter peak time-periods. At each iteration of our simulation, in order to define the set $\Omega$, we choose 20 scenarios from a scenario space consisting of 200 trading-periods from morning peaks in July and August 2016 and 2017. We construct random sets of $\Omega$ that represent the different types of time periods. Note that 2016 and 2017 are very different years in terms of energy prices and represent a wide range of scenarios. Hence, we have assigned scenarios to 4 different clusters which are based on the average island energy prices in the morning peak trading periods (6 AM - 10 AM). The sampling scheme is to pick scenarios from all the clusters (i.e. high and low priced weeks), in order to capture the diversity of scenarios.

After sampling a scenario set, we compute the optimal consumption and ILR stack by solving $[\alpha$-MIP]. Figure 5.11 shows the optimal bid stack for the NZAS for one of the sampled scenario sets, which is calculated for different marginal values of electricity consumption ($u$).
Subsequently, these optimal stacks need to be submitted to the co-optimized OPF, which renders the dispatched quantities and prices for the upcoming realized scenario. Therefore, in the following experiment we test the performance of the optimal stacks that are derived from historical data through simulating them (i.e. submitting the stacks to the co-optimized OPF) for 100 out-of-sample scenarios, and report on the average expected gained profit. Figure 7.8 illustrates the stages of the algorithm.

5.5.4 Policy Performance Comparison

Finally, we compare our proposed policy with the clairvoyant optimization (absolute upper bound on profit) and the fixed load quantity that is consumed by NZAS in practice. Table 5.12 outlines the average results of 200 simulations. The experiment is done over
different levels of electricity consumption’s marginal value \((u)\) to compare the performance of our proposed policy over a broader range of parameters\(^2\). The first column of this table shows the different marginal value levels. The second column demonstrates the average profit over out-of-sample scenarios with our strategic policy’s optimal stack. The third column presents the average profit results for the fixed policy in which the consumer targets a given quantity of consumption. We have adjusted this experiment for the different levels of \(u\), thereby the fixed consumption level is set to be higher in the experiments with greater \(u\) values and lower for smaller \(u\) values. The fourth column shows the average profit of the clairvoyant policy when solved for each out-of-sample scenario. The fifth column shows the increase in the profit when using our proposed policy, versus the fixed policy (the policy that is used in practice by NZAS). The last column presents the percentage of clairvoyant policy’s profit that is recovered by our proposed stochastic out-of-sample policy. Moreover, Figure 5.13 lays out the standard box-and-whisker plots corresponding to winter peak profit values for different levels of marginal utility \((u)\). Note that for the following simulations, we compute the profit for the length of one trading period (half-hour).

<table>
<thead>
<tr>
<th>(u)</th>
<th>Optimal Stack</th>
<th>Fixed Quantity</th>
<th>Clairvoyant</th>
<th>Profit increase</th>
<th>% of clairvoyant</th>
</tr>
</thead>
<tbody>
<tr>
<td>30</td>
<td>1702</td>
<td>1622</td>
<td>2156</td>
<td>4.9%</td>
<td>78.9%</td>
</tr>
<tr>
<td>50</td>
<td>5104</td>
<td>3641</td>
<td>5502</td>
<td>40.1%</td>
<td>92.7%</td>
</tr>
<tr>
<td>70</td>
<td>9237</td>
<td>6719</td>
<td>9891</td>
<td>37.4%</td>
<td>93.3%</td>
</tr>
<tr>
<td>90</td>
<td>13619</td>
<td>10003</td>
<td>14827</td>
<td>36.1%</td>
<td>91.8%</td>
</tr>
<tr>
<td>110</td>
<td>17996</td>
<td>13764</td>
<td>20447</td>
<td>30.7%</td>
<td>88.0%</td>
</tr>
<tr>
<td>Average</td>
<td>9532</td>
<td>7150</td>
<td>10565</td>
<td>29.8%</td>
<td>88.9%</td>
</tr>
</tbody>
</table>

**Table 5.12: Policy performance- winter peak**

We also simulated policies for off-peak trading periods in winters 2016 and 2017. Table 5.13 outlines the average results of 300 simulations that implemented the optimal policies for peak and off-peak prices when \(u = \$90/\text{MWh}\).

<table>
<thead>
<tr>
<th>TP Type</th>
<th>Optimal Stack</th>
<th>Fixed</th>
<th>Clairvoyant</th>
<th>Profit increase</th>
<th>% of clairvoyant</th>
</tr>
</thead>
<tbody>
<tr>
<td>Winter peak</td>
<td>13619</td>
<td>10003</td>
<td>14827</td>
<td>36.1%</td>
<td>91.8%</td>
</tr>
<tr>
<td>Winter off-peak</td>
<td>16197</td>
<td>12671</td>
<td>17650</td>
<td>27.8%</td>
<td>91.7%</td>
</tr>
</tbody>
</table>

**Table 5.13: Winter policy performance**
In order to extend the simulation to a broader range of trading periods we have run experiments based on 2018/2019 summer which had a series of high prices due to lake levels running low towards the end of summer. We have sampled scenarios from weekdays in December 2017 and January 2018, separately for summer afternoon peak (1 pm - 6 pm) and early morning off-peak (3 am - 6 am) trading periods. Table 5.14 outlines the simulation results for summer data.

Note that the sample space of the trading periods have higher price than the same time of year in the previous years. The reason we picked this particular set of trading periods is that there had been an anticipation of high prices given the dry summer and low reservoir levels. Hence this demonstrates that such a signal can be used by major consumers to change their usage plans in order to prevent price spikes. In this experiment we have used a mid February week trading periods as the out-of sample scenarios to show the effect of incorporating our proposed policy while taking into account the anticipation of low lake levels, versus the policy in practice in NZAS which disregarded the signal.

\[\text{Fixed} \quad \text{Optimal} \quad \text{Clairvoyant}\]

**Figure 5.13: Winter peak profit vs value of electricity $u$**

---

2The marginal value of electricity for NZAS may change over time as it is affected by exogenous parameters such as the global price of aluminum, government policies, etc.
### 5.6 Conclusions

In this chapter we have addressed the uncertainty in energy and reserve markets. Our stochastic model optimizes over a set of scenarios, delivering admissible optimal ILR offer and consumption bid curves for a large consumer of electricity. We presented illustrative examples to study the impacts of strategically submitting bid and offer curves. The examples in subsections 5.3.2 and 5.3.3 compared the co-optimized strategic curves with the strategic energy-only and strategic reserve-only bidding and offering respectively. In Section 5.3.2, we analyzed the effects of co-optimizing an ILR offer curve with the energy bid curve. We demonstrated that for our 3-node example, simultaneously optimizing for both energy and ILR stacks can result in twice as much expected profit as energy-only optimization. Subsequently, in subsection 5.3.3, we laid out an example that compared the co-optimized bidding and offering, with the BOOMER-consumer method, where the ILR stacks are optimized for discretized levels of consumption. Similarly, this experiment highlighted the advantages of utilizing co-optimized here-and-now decision making for a major consumer.

Solving our model to global optimality requires reformulating the stochastic bi-level model. Using the standard approach to reformulate our model results in a large and complex MIP due to stochasticity and the necessary monotonicity constraints. In order to reduce the complexity and solve time, we introduced different decomposition methods to enhance our formulation. In subsection 5.4.5, we introduced a novel reformulation method in which we used bi-parametric sensitivity analysis to reduce the size of our model. We showed that, compared to the standard reformulation method, the bi-parametric reformulation reduces the number of integer variables up to one order of magnitude. The computational advantage of this decomposition method enabled us to

### Table 5.14: Summer policy performance (with $u = 90$/MWh)

<table>
<thead>
<tr>
<th>TP Type</th>
<th>Optimal Stack</th>
<th>Fixed</th>
<th>Clairvoyant</th>
<th>Profit increase</th>
<th>% of clairvoyant</th>
</tr>
</thead>
<tbody>
<tr>
<td>Summer peak</td>
<td>7867</td>
<td>2403</td>
<td>9742</td>
<td>227.3%</td>
<td>80.7%</td>
</tr>
<tr>
<td>Summer off-peak</td>
<td>14055</td>
<td>9073</td>
<td>18235</td>
<td>54.9%</td>
<td>77.0%</td>
</tr>
</tbody>
</table>

As shown in Table 5.14 the fixed consumption policy performs poorly in the summer peak, and in fact it is possible to achieve a 227.3% profit increase by submitting an optimized bid stack. However, in the summer off-peak periods the improvement over the fixed consumption policy is lower (54.9% higher). Moreover, for both of these types of periods the average profits are only about 20% below the corresponding profit from the clairvoyant policy.
apply our stochastic strategic model for a major consumer in the NZEM and solve it over a set of scenarios, consisting of 20 sample time periods (presented in Section 5.5.3).

In the final part of this chapter we simulated the performance of our policy over out-of-sample scenarios. We then compared the results of our model with the fixed and clairvoyant policy, in subsection 5.5.4. We demonstrated that our stochastic co-optimization resulted in around 30% increase in expected profit, compared to a fixed demand policy.

Our model has provided significant intuition for a major consumer’s utility-maximizing consumption and a reserve offer strategy for one trading period. Our strategic model adjusts the consumption and ILR level with the marginal utility of consuming electricity. However, in practice, many of the manufacturers are committed to their contracts with their clients and, therefore, responding to the changes in marginal value of energy may not be possible. In order to address this, we apply the $\alpha$-MIP model to a multistage setting, in which we solve a cost-minimizing strategic co-optimization problem over a time horizon in the next chapter.
Part III

Multi-stage Co-Optimization of Energy and ILR
Chapter 6

Strategic Consumption over a Finite Time Horizon

In part II, we presented deterministic and stochastic models that construct optimal bid and offer stacks for a strategic major consumer. In these models we maximized profit, which depended on the marginal utility of consumption, price of energy and reserve, and CfDs that the consumer holds. The utility of consuming electricity is affected by various parameters such as the global price of a manufacturer’s product, the labor costs, etc. This utility also may change slightly or drastically throughout time for industrial consumers due to changes in the product price or of government policies. For this reason, manufacturers typically enter into long-term contracts to deliver specific quantities of their product by a given date; so the question becomes “how should a manufacturer choose to schedule consumption in order to minimize its total cost?”.

Therefore, in this chapter we change our point of view from a profit-maximizing perspective to one where we have a total consumption target and we schedule the consumer’s load, over a given time horizon, at least-cost; this is commonly referred to as load-shifting.

6.1 Introduction

Since the 1950s, there has been a wide range of research on solution methods for large-scale multistage linear stochastic program models [138–140]. The classical Benders decomposition technique [141] is a base for multiple decomposition algorithms that tackle large-scale multistage models [142, 143]. In addition, augmented Lagrangian methods have been used to solve such large-scale optimization programs [144, 145]. Moreover, regularized decomposition approach has been used for multistage models in [146, 147].
Progressive hedging algorithm that was introduced in [148], is another method that is known for solving the multistage stochastic linear problems to optimality.

Stochastic Dual Dynamic Programming (SDDP) is a sampling-based nested decomposition method that was first introduced in [149], and further studied in [150–152]. There are also effective general decomposition algorithms that are related to SDDP, such as [153, 154]. In addition, constructive [155], approximate [156] and generalized [157] dual dynamic programming methods utilize similar concepts to tackle multistage models as of the SDDP approach.

On the other hand, while there exists a wide range of work on two-stage stochastic integer programming models, the developed literature on solution techniques for non-linear multistage stochastic programming models is more recent and limited [158]. Many of the decomposition methods used for mixed-integer multistage programs are based on the decomposition of the model into finite number of deterministic optimization problems, where the so-called Non-Anticipativity Constraints (NAC) are satisfied in a block-angular form [159].

Stochastic (augmented) Lagrangian relaxation of coupling constraints is widely used in decomposing the multistage models. The early utilization of this method for integer multistage models can be found in [160, 161]. In addition, more recent work on Lagrangian relaxation decomposition such as [162, 163] apply stochastic Lagrange multipliers on a subset of constraints (clusters), while obtaining tight bounds as the standard approach.

Progressive Hedging (PH) is another decomposition technique that was originally utilized for convex models only. In [164], Løkketangen and Woodruff, apply heuristics on the PH algorithm, where solutions to the induced quadratic mixed-integer sub-problems are obtained using a tabu search algorithm. Furthermore, in [165], a heuristic PH method is introduced to solve scenario-based resource allocation problems with integer variables, where the model is empirically assessed on a two-stage stochastic integer problem. Moreover, in [166], an extension to the integer PH model is presented through utilizing a scenario cluster approach. However, all these integer PH methods, can only be used if the objective function is convex, which may not be the case for many integer multistage models, including our strategic bidding model.

Given the scale of real-world integer dynamic models, a lot of authors have considered the use of heuristics in their models in order to find “good” approximations of the original problem’s value functions. For instance in [167], a multi-period investment model is presented for capacity expansion in an uncertain environment. In this model a branch and bound algorithm is proposed as a lower bounding scheme and a heuristic method is used to determine the upper bounds. In addition, in [168], an inexact scenario-based
multistage stochastic integer programming method is developed for water resource management under uncertainty. Moreover, a sensitivity analysis based model for multistage stochastic problems with integer constrains is developed in [169], where they use approximations of the future expected value, as well as utilizing scenario clustering methods.

Advances in computer technology allow many models to utilize parallel optimization to reduce the solve-time by making use of the ability to run multiple smaller optimization problems simultaneously. Parallel decomposition algorithms for multistage models were first introduced in [145], where the sub-problems are solved in parallel, using a nonlinear interior point algorithm. In [170, 171], parallelization methods are developed for convex hydroelectric systems. Moreover, in [172], a parallel computing implementation is proposed that is specialized for mixed integer multistage models.

The SDDP algorithm is a powerful tool for solving convex multistage models, however, when tackling models with integer or binary variables, the standard SDDP cannot be used without introduction of heuristics, approximations or linear relaxation of integer terms. For instance, Flach et al., in [173], utilize the SDDP algorithm, in which the integrality is handled through a piece-wise linear approximation of the expected future value function. The model that is proposed in [174] is another implementation of SDDP for mixed integer models, where the non-convex constraints are approximated by McCormick envelopes.

Recently, Zou et al., in [175], introduced a comprehensive integer extension to SDDP, called stochastic dual dynamic integer programming (SDDiP), in which they assume binary state variables and convex value functions. In order to tackle the binary state variables, they propose a new class of cuts, named as Lagrangian cuts, which are derived from a Lagrangian relaxation of each stage’s sub-problems. Following the SDDiP method, Zou et al. implement their model for a unit commitment problem in which all state variables are binary [176]. Finally, in a recent work from Steeger et al., a stochastic multistage model is solved with non-convex value functions, through utilizing a combination of SDDP and Lagrangian relaxation. In this work, although they use an exact representation of the non-concave immediate revenue function, they use an approximate concave function for determining the future revenue.

One of the main applications of large-scale stochastic multistage optimization problems is in the power systems literature. Firstly, the practical energy models are large-scale optimization problems, as they are designed for real-world networks, where the electricity procurement for an entire country or a large region (i.e. PJM market) is optimized at each trading period. Secondly, dynamic (multistage) models are required in order to present mid- and long-term planning for power systems. These models could be from the perspective of the ISO for unit commitment optimization, dynamic water flow
planning for hydro-generators, or scheduling demand over a time horizon for major consumers (which is the focus of this thesis). Last but not least, due to the volatility in real-time demand, unpredictability of precipitation, and other factors, power system optimization models often involve stochastic optimization. The assumption of deterministic parameters is considered a non-trivial oversimplification for these models. The three discussed properties have led to development of a large class of optimization problems in both convex and non-convex settings. Linear multistage models are dominantly used for hydro-generation and other price-taker models. On the other hand, the mixed-integer models are developed to address the binary nature of day-ahead unit-commitment scheduling, as well as dynamic strategic behavior of large agents.

Linear multistage models in power systems have been widely influenced by work of Pereira and Pinto [149, 153] that introduced the SDDP algorithm for convex energy planning models. In their model, they approximate the expected-cost-to-go functions of stochastic dynamic programming by piece-wise linear functions, and test their proposed algorithm on a 39-reservoir energy system. In [177], the SDDP algorithm for hydro-thermal systems is further studied, in which they develop sampling strategies and stopping criteria for the SDDP algorithm.

Moreover, in [178], an interval-parameter multistage programming method, combined with scenario bundling is developed for a stochastic water flow planning model. Furthermore, Conejo et al. in [179], solve a linear multistage model from a consumer’s perspective approach to plan its demand in response to hourly electricity prices. In their paper, the price uncertainty is modeled through robust optimization techniques. Recently, in [180], a large-scale model is developed for electricity generation, storage, and transmission investments over a long planning horizon, where PH is used to decompose the stochastic multistage linear model.

Non-convex dynamic energy system models have been the focus of many researchers in the past two decades. When scheduling long-term energy procurement for a hydro-thermal generation system, integer variables are required for modeling of reservoir head effects [181], as well as addressing thermal capacity and ramp rates. In [161], an energy planning model is considered in which they utilize Lagrangian relaxation via maximization of the dual function by an approximate bundling method, while using sub-gradient information (based on the sub-gradient methods introduced in [182, 183]). In addition they use Lagrangian heuristics to find a feasible first-stage decision, and subsequently solve the market clearing problem for determining a nearly optimal first-stage decision. Similarly, in [184, 185], unit commitment problems for hydro-thermal generation are addressed through using heuristic (augmented) Lagrangian techniques combined with
bundling methods. Furthermore, Diaz et al., in [186], introduce a mixed integer nonlinear programming model for scheduling of the short-term integrated operation of a series of price-taking hydroelectric plants along a cascaded reservoir system in a deregulated electricity market. In this model they utilize nonlinear regression analysis in order to tackle the integrality.

In energy market optimization, modelling price-making agents may result in non-convex problems, often due to the nature of step function offer stacks. For instance, in [173], by utilizing the SDDP algorithm, a deterministic multistage model for price-making generators is developed, and simulated for the hydro-thermal system of El Salvador. Their proposed algorithm is based on SDDP algorithm, whereby, in order to tackle the non-convexity, they use a piece-wise linear approximation of the expected value function. Similarly, Baslis and Bakirtzis [187] offer a model for medium term stochastic scheduling of a price-making hydro-generator with pumped storage where the hydro-generator faces modified stochastic residual demand curves but over small number of scenarios and only three stages.

As discussed earlier, SDDP is widely used for multistage convex models in energy systems. When integrality is added to the power system optimization problems, SDDP may be utilized through additional heuristic methods. For example, in [174], in order to model hydro-thermal generation, the SDDP decomposition algorithm is applied to a mixed integer multistage stochastic optimization problem. In this method, the non-convex constraints that represent the production function of hydro plants are approximated by McCormick envelopes. Moreover, Granville et al. extended the SDDP algorithm through incorporating linearized transmission constraints [188]. Furthermore, Scott and Read in [189], utilize the constructive dual dynamic programming algorithm that they introduced in [155] to address multistage Cournot duopolies of hydro-generators.

The integer extension of SDDP, is also utilized for energy system models. Such as [176], where authors solve a dynamic programming formulation of a multistage stochastic unit commitment problem while assuming binary state variables. The strategic bidding problem of a price-making agent, has been recently addressed in [190], where they utilize SDDP, and tackle the non-linearities arising from the strategic behavior through dynamically convexifying the future value function via Lagrangian relaxation. This work is developed for aggregate strategic hydro-generators for El Salvador, Honduras, and Nicaragua markets. The authors, have also extended this model to allow for stage-wise dependency in [191].
Our Contributions

While there exists related literature on optimal behavior of price-making consumers and large-scale multistage strategic bidding, there is no work on large-scale multistage stochastic modelling of a price-making consumer who co-optimizes its demand bid and ILR offer whilst meeting a required level of demand over a finite time horizon. Here we compare the contribution of our model to the existing literature.

Firstly, while there exist multistage price-making major consumer models (presented in [109, 110]), they are not applicable for long-term planning over real-world energy markets, as they do not scale for large networks, such as NZEM. We offer methodologies that enable us to solve our real-world problem for a month-long time horizon, with half-hour periods.

Secondly, as discussed in the literature review, the works of Steeger et al. in [190, 191], address a price-making hydro-electric generator who strategically offers energy over a time-horizon. This model can be adapted for a price-making major consumer, where the total demand requirement for the consumer can be modelled the same way as the water storage constraint for the hydro-generator. However, their proposed model falls short to address the following aspects of our model. Firstly, their time horizon consists of 24 stages, which is much fewer than the number of trading periods we aim for (1440 stages). Secondly, the price-making hydro-generator that is studied in [190, 191] operates over the equivalent of a single-node network, i.e. their model is solved for a single demand level and resulting price of electricity at each trading period. On the other hand, the problem we tackle is implemented for the NZEM that operates over a large transmission network. Finally, the method that is used to tackle the non-convexity of the value function in [190, 191], convexifies the future expected value curves, which are approximations of the true value functions. Our method, however, tackles non-convexities through utilizing step-wise value curves. Moreover, [190, 191] study stage-wise independence and dependence of stochastic reservoir inflows, respectively. Our method also is applicable for both types of stage-wise dependent and independent models of stochasticity.

In the multistage integer stochastic optimization literature, there exist several methods to approximate the expected future value (EFV) function. Our method includes constructing value-consumption curves that present the marginal expected value, at each stage, for each level of the state variable. Similar attempts have been used in integer multistage settings, and here we address the novelty of our method, compared to the existing literature. Firstly, in [169], EFV curves are used for solving large-scale integer planning problems under uncertainty. In order to approximate the EFV curves an ad
hoc sensitivity analysis is used by truncating the Taylor series expansion of the objective function around the values of the linking variables. The contribution of our approach compared to this model is that we use the exact marginal utility values for given ranges of the values of the linking variables. Secondly, in our proposed solution method for the price-making major consumer problem, we have utilized the idea of demand curves for release that is introduced in [189] in a price-taking hydro-generation setting whereby we constructed Utility-Consumption (U-C) curves as look-up tables for the forward pass simulation. The novelty of our work is in constructing U-C curves for a price-maker agent. In addition, we offer a large-scale multistage model, over a long time-horizon, that takes into account uncertainty associated with the market, as well as allowing for co-optimization of strategic load with ILR.

**Chapter Structure**

In this chapter we offer solution methods for both price taker and price maker consumers. However, the emphasis of our results is on the price-making model. In Section 6.2, we lay out the structure of our multistage model and present a mathematical formulation for this model. In Section 6.3, we present the standard dynamic methods that could be used as a solution method for our model and discuss their limitations. In order to offer practical solution methods we develop Lagrangian decomposition methods for our multistage MIP model in Section 6.4. In order to build optimal policies through utilizing our decomposed model, we construct look-up tables in Section 6.5. In Section 6.6, we develop a heuristic model which enables us to implement our model over set of all possible contract levels. Furthermore, in Section 6.7 we calculate upper and lower bounds on the solutions of our optimization problem and compare it with our optimal policy’s results. Moreover, in Section 6.8, we lay out the algorithm to construct U-C curves. In order to comprehensively probe the strategic behavior of the major consumer under our proposed multistage optimal policy, we lay out a 3-node example in Section 6.9; in addition, we compare the optimal policies when a CfD is introduced to our model. Section 6.10 summarizes the findings of the chapter.
Specific Chapter Nomenclature

\( \mathcal{T} \) \hspace{1cm} The set of all time periods \( t \) in the time horizon.

\( \mathcal{S}_t \) \hspace{1cm} The set feasibility constraints at time period \( t \).

\( x_t \) \hspace{1cm} The consumption level value at time period \( t \).

\( \mathbf{x}_t \) \hspace{1cm} The vector of decision variables (i.e. consumption and ILR) at time period \( t \).

\( C_t \) \hspace{1cm} The cost function at time \( t \).

\( \mathcal{S}_t \) \hspace{1cm} The feasible set at stage \( t \), which may be non-convex.

6.2 Problem Structure

In this section we start with the single-stage profit maximizing MIP model that we laid out in Section 4.3. For sake of simplicity, and without loss of generality, we denote this single-stage co-optimization model as \([\text{MIP}]_{t,v}\) below. Note that, in this chapter, our single-stage MIP problem is indexed by marginal value of consuming electricity \( (v) \) and the time period \( (t \in \mathcal{T}) \).

\[
\begin{align*}
[\text{MIP}]_{t,v} \quad & \max \Pi_t(\mathbf{x}_t) = vx_t - C_t(\mathbf{x}_t) \\
& \text{s.t. } \mathbf{x}_t \in \mathcal{S}_t
\end{align*}
\]

One of the important parameters in \([\text{MIP}]_{t,v}\) is the marginal value of energy consumption \( (v) \). In chapter 4 we solved \([\text{MIP}]_{t,v}\) for each time period \( t \) separately according to the marginal value of electricity consumption \( (v) \). In this chapter, however, we aim to change our point-of-view from a marginal utility value perspective to a total consumption value target and plan the consumer’s load over a given time horizon. Hence, we lay out a model that optimizes the consumption and ILR levels for a strategic consumer, given a total consumption level \( G \) over a time horizon \( \mathcal{T} \); we denote this optimization problem \([\text{T-MIP}]\) as below:

\[
\begin{align*}
[\text{T-MIP}] \quad & \max \Pi(\mathbf{x}) = -\sum_{t \in \mathcal{T}} C_t(\mathbf{x}_t) + \sum_{t \in \mathcal{T}} vx_t \\
& \text{s.t. } \mathbf{x}_t \in \mathcal{S}_t \\
& \sum_{t \in \mathcal{T}} x_t = G, \quad \forall t \in \mathcal{T}
\end{align*}
\]
Chapter 6 Strategic Consumption over a Finite Time Horizon

Assume $x^*$ is the optimal solution to [T-MIP], hence $\sum_{t \in T} v x^*_t = vG$, which is a constant. This shows that the marginal value of consumption will not affect the optimal actions, since the total consumption value is the key parameter to control consumption at each time period\(^1\). In this model we consider this value to be exogenous to our model.

Given the observation above, we continue by defining [CM-MIP] as below. Note that the optimal solution to [T-MIP] solves [CM-MIP].

$$\text{[CM-MIP]} \quad Z(G) = \min \sum_{t \in T} C_t(x_t)$$

s.t. $x_t \in S_t \quad \forall t \in T$

$$\sum_{t \in T} x_t = G$$

In [CM-MIP], all stages are linked together via the total consumption constraint yielding a complex MIP model which we attempt to solve in this chapter. Given that this large problem cannot be solved with commercial solvers in a reasonable time-frame, decomposition methods are needed in order to find the optimal policy. In particular, we will implement a decomposition method which enables the flexibility to change the problem parameters (e.g. total consumption value) throughout the optimization period.

6.3 Dynamic Programming Solution Methods

In order to make our model computationally tractable, we aim to decompose our MIP in a way that we can solve each time period separately, while taking into account the effects of the actions in each time period on the other ones. At the first glance, the structure of [CM-MIP] fits well to be solved via dynamic programming (DP), with the “consumption to go” as the one dimensional state variable. In this section we lay out the standard DP setting for our model, as well as a modified heuristic version of DP, and discuss their properties and limitations.

6.3.1 Standard DP

In order to use the standard DP algorithm for our model, we utilize the Bellman equation (shown in Section 2.4). Firstly, we denote the stages by $t$ over the finite horizon $\mathcal{T}$. Furthermore, we define the state variables ($s \in \mathcal{O}_t$) as the consumption to go at stage $t$,

\(^1\) When the decision makers set the value $G$ for their firm, they already have taken into account the average marginal value of consumption over the designated time horizon.
and action \((a \in A_{t,s})\) as the amount of energy consumption by the strategic consumer at state \(s\) and stage \(t\). Lastly, we denote \(c_t(a)\) as the cost of taking action \(a \in A_{t,s}\) at stage \(t\) and state \(s\).

For the last trading period in our time horizon \((t = T)\), we must ensure that at the end of time horizon the total consumption requirement is met, therefore we set \(a = s\) which implies:

\[
A_{T,s} = \{s\} \quad \forall s \in \mathcal{O}_T. 
\] (6.1)

Using (6.1) we can calculate \(c_T(a)\) through solving \([CM-MIP]_t\) for \(t = T\) with the fixed consumption level \(x_t = a\), which we lay out as below:

\[
c_t(a) = \min \mathcal{C}_t(x_t) \\
\text{s.t.} \quad x_t \in \mathcal{S}_t \\
\quad \quad x_t = a
\]

We define the cost-to-go function at stage \(t\) and state \(s\), as \(C_t(s)\). For \(t = T\), we have \(C_T(s) = c_T(s)\). For all the stages \(t \neq T\), we use the Bellman equation (2.1) to calculate the cost-to-go function as below:

\[
C_t(s) = \min_{a \in A_{t,s}} \{c_t(a) + C_{t+1}(s - a)\}
\]

In this dynamic program, at each time period and price-state, a MIP needs to be solved to calculate the cost associated with each potential consumption action. When we implemented this standard approach for our full-scale MIP model, it did not perform efficiently as the number of price-states becomes very large for long time-horizons. For instance, for a monthly plan, and 1 MWh discretization step, this requires approximately 1.15 million MIP solves. Implementing this model for our strategic consumer results in a large problem that is not possible to solve over a long time-horizon.

In addition to the standard DP, we tried a heuristic semi-continuous DP model in which, instead of defining states by a consumption to go quantity, we assign a range of feasible consumption quantities to each state. The transition from stages would follow the principals of dynamic programming but the optimal action for each state is calculated through solving \([CM-MIP]\) over the state’s corresponding consumption region. This semi-continuous model not only decreases the number of states in each stage but also
allows us to compute the exact optimal consumption level at each state instead of a
discretized consumption value.

Our proposed heuristic DP model is computationally more tractable than the standard
DP, however, it does not scale for our real-world problem. Our goal is to offer a non-
discretized optimal demand schedule for a major consumer in a practical time-frame,
hyence, in the next section, we present alternative decomposition methods.

### 6.4 Lagrangian Decomposition

A commonly-used approach to decompose a master-problem with a constraint that links
the sub-problems together is Lagrangian relaxation. In order to decompose our model we
take advantage of the unique properties of our model, in addition, we adapt ideas from
decomposition methods for convex models. We start with a price-taking model (which
is convex), and we will then extend this methodology and apply it to the [CM-MIP]
model.

#### 6.4.1 Price Taker Consumer

In this section we define an LP, where a price-taking consumer minimizes its cost, given
a total consumption quantity \( G \) MWh is required, over the set of linear constraints \( S_t \),
\( \forall t \in T \).

\[
\begin{align*}
\text{[LP]} \quad & \min \sum_{t \in T} C_t(x_t) \\
\text{s.t.} \quad & x_t \in S_t \quad \forall t \in T \\
& \sum_{t \in T} x_t = G \quad [u]
\end{align*}
\]

In the next step, we use the Lagrangian relaxation method to reformulate the LP as
below:

\[
\begin{align*}
\text{[LP]} \quad & \min \sum_{t \in T} C_t(x_t) - u(\sum_{t \in T} x_t - G) \\
\text{s.t.} \quad & x_t \in S_t \quad \forall t \in T
\end{align*}
\]
Given $G$ is a constant, if we find the right $u = \hat{u}$ that solves [LP], we can separate [LP] by time periods. Hence for each $t \in T$ we have:

$$[LP]_t \min C_t(x_t) - \hat{u}x_t$$

s.t. $x_t \in S_t$.

Assuming strict convexity, $\hat{u}$ is essentially an increasing function of the consumption quantity $G$. The higher $\hat{u}$ is, the more the consumer will consume. In the next section, we transition to the price-maker consumer model by adapting this decomposition method for our MIP model, but as we will see, $\hat{u}$ will no longer be a function of $G$.

### 6.4.2 Price Maker Consumer

In the price-maker version, since the model is non-convex, we cannot acquire a dual variable on the total consumption constraint. However, by utilizing the idea of Lagrangian relaxation, we can reformulate our model, using a parameter which mirrors the properties of a multiplier. In order to build a decomposed MIP, we define $[U\text{-MIP}]_u$ as the problem with the total consumption constraint removed, and instead valued in the objective. The lemma below provides a sufficient condition where the optimal solution of $[U\text{-MIP}]_u$ solves $[CM\text{-MIP}]$.

$$[U\text{-MIP}]_u \max - \sum_{t \in T} C_t(x_t) + \sum_{t \in T} u x_t$$

s.t. $x_t \in S_t$ \quad \forall t \in T

**Lemma 6.1.** Let $x^*$ be the optimal solution to $[U\text{-MIP}]_u$, for some $u \geq 0$. Then $x^*$ solves $[CM\text{-MIP}]$, provided $G = \sum_{t \in T} x^*_t$.

**Proof.**

$$\Pi(x^*) = - \sum_{t \in T} C_t(x_t^*) + \sum_{t \in T} u x_t^* \geq - \sum_{t \in T} C_t(\hat{x}_t) + \sum_{t \in T} u \hat{x}_t$$

$\forall \hat{x}_t \in \mathcal{X}$

$$\Rightarrow \sum_{t \in T} C_t(\hat{x}_t) \geq \sum_{t \in T} u (\hat{x}_t - x_t^*) + \sum_{t \in T} C_t(x_t^*)$$

$$\Rightarrow \sum_{t \in T} C_t(\hat{x}_t) \geq \sum_{t \in T} C_t(x_t^*)$$

$\square$
Note that \([\text{U-MIP}]_u\) is decomposable to each time period (as \([\text{MIP}]_{t,v}\) and \([\text{U-MIP}]_{t,u}\)\(^2\) are the same). The next step is to find the right value of the parameter \(u\) in order to find the optimal actions, for valid given total consumption values \(G\). In order to make policies that result in finding the right value of \(u\), without relying on convexity or continuity, we introduce the Utility-Consumption (U-C) curves in the next section.

### 6.5 Utility Consumption Curves

In order to show the properties of the model with respect to the \(u\) parameter value and optimal consumption level, we construct look-up curves. The idea of using such curves is introduced in [189] for a hydro-scheduling setting, whereas in this paper we extend this concept to a price-making optimal consumption model.

In this section, in order to construct the U-C curves for each trading period \(t\) we solve \([\text{MIP}]_{t,u}\), \(\forall u \in [0, u_{\text{max}}]\), and plot the values of \(u\) versus its optimal consumption level, \(g(u, t) = x^*_t\) (shown in Figure 6.1). Mapping the consumption level to utility values is essentially a set valued mapping. In particular \([0, u_{\text{max}}]\) is partitioned by a finite number of sub-intervals (indexed by \(k\)) such that \(\forall u \in (u_k, u_{k+1})\), \(g(u)\) is constant.

As demonstrated in Figure 6.1 for each of the finite number of ranges of parameter \(u\), the optimal consumption remains constant within the range. Also we observe that by increasing \(u\), the consumption value jumps to a higher consumption level. Below we prove that the U-C curves are monotone increasing.

**Lemma 6.2.** Let \(x^*\) be the optimal solution to \([\text{MIP}]_{t,u}\), for some \(t\) and \(u\), and \(\Pi(x^*, u)\) be the optimal objective function value of \([\text{MIP}]_{t,u}\). Then \(\forall u \geq 0\), \(x^*(u)\) is monotone increasing.

\(^2\)Note that \([\text{MIP}]_{t,v}\) from Section 6.2 and \([\text{U-MIP}]_{t,u}\) are essentially the same optimization problem. Recall that the interpretation of parameter \(v\) in \([\text{MIP}]_{t,v}\) is the marginal value of consuming electricity; whereas the parameter \(u\) in \([\text{U-MIP}]_{t,u}\) is the multiplier on the total consumption constraint. Hence, although we have changed our perspective from a marginal utility point of view to a Lagrange multiplier value, these two parameters are identical.
Proof. We will first define $[\text{MIP}]_{t,u'}$, for some $u' > u$, where $\Pi(x^*, u')$ is the optimal objective value for $[\text{MIP}]_{t,u'}$; that is laid out below:

$$[\text{MIP}]_{t,u'} \max -C(x) + ux + (u' - u)x$$

s.t. $x \in S$.

Using the definition of $[\text{MIP}]_{t,u'}$ and $[\text{MIP}]_{t,u}$, we have:

$$\Pi(x^*, u) = \Pi(x^*, u') - (u' - u)x^*$$

and

$$\Pi(x^*, u') = \Pi(x^*, u') - (u' - u)x^*'.$$

Given $x^*$ is the optimal value of $x$ for $[\text{MIP}]_{t,u}$, then no objective function value higher than $\Pi(x^*, u)$ can be obtained for $[\text{MIP}]_{t,u}$. Therefore, we have:

$$\Pi(x^*, u) \geq \Pi(x^*, u') \implies \Pi(x^*, u') \geq \Pi(x^*, u') - (u' - u)(x^* - x^*)$$

Using the expression above we prove our lemma by contradiction. Suppose $x^{*'} < x^*$, therefore we have $(u' - u)(x^* - x^*) < 0$, this implies $\Pi(x^*, u') > \Pi(x^*, u')$ which in turn yields a contradiction.

\[ \square \]

In order to build U-C curves, we first define the order of $G$ values in $G$ from lowest to highest. Given the monotonicity of $G$ with respect to $u$, we know that for the first $G$ value in $G$ there will be a lower associated $u$ value than the second $G$ in $G$; this property enables us to easily build the whole curve. The next step is to find the optimal policy, via the aggregated U-C curve over all time periods. In order to make the aggregated curve, for any value of the parameter $u$, we compute the the sum of corresponding optimal consumption values over all the trading periods’ individual U-C curves in the time horizon. Therefore for each $u$ we have the total consumption values in the aggregated U-C curve $G(u) = \sum_{t \in T} g(t, u)$. Figure 6.2 shows an aggregated curve where the horizontal axis corresponds to total consumption values.

Given the method used for aggregating the U-C curve, the aggregated curves will have more pieces than the sum of individual time-periods’ curves. Here the points on the total consumption ($G$) axis, form the set of total consumption values $G = \{G_1, G_2, \ldots, G_5\}$ that we can choose from. After choosing $G$, we can look up our U-C curve and find the range of optimal $\hat{u}$ values for our model. Therefore we can solve each $[\text{U-MIP}]_{t,\hat{u}}$ separately, given $\hat{u}$. However in the case where $\hat{G} \notin G$, the optimal solution to $[\text{U-MIP}]_u$ is not equivalent to that of $[\text{CM-MIP}]$ as there is no $u = \hat{u}$ such that $\sum_{t \in T} x^*_t = \hat{G}$. Hence we cannot simply use the aggregate U-C curve to calculate the consumption in each time period.
The structure of our U-C curves indicate that to increase any designated $G \in \mathcal{G}$ value, we simply increase consumption in some trading periods, i.e. there is no decrease of consumption in any of the time-periods; this means that lower $G$ values are nested within higher $G$s. However, for $G \notin \mathcal{G}$ the nested property might not apply, and so the optimal solution could indicate decreasing consumption at one trading period and increasing consumption in another. In order to address this issue we present a heuristic method and bounds on the heuristic solution in the following sections.

### 6.6 Heuristics

In this section we introduce a heuristic algorithm to extract policies from the aggregate U-C curves, for any given value of $G$. First, for each time period’s U-C curve, we connect the vertical lines with virtual horizontal lines, to make step functions (as shown in Figure 6.3). This step-wise function has the same attributes as a supply function for a generator in the dispatch model. We view each time period as a node, and the total consumption value as the demand. We introduce [HEU], where we minimize utility $\times$ consumption.

![U-C curve demonstration for HEU](image)

**Figure 6.3:** U-C curve demonstration for [HEU]

\[
\text{[HEU]} \quad \min \sum_{t \in \mathcal{T}} \sum_{i \in \mathcal{I}_t} u_i^t x_i^t
\]

s.t. \[ \sum_{i \in \mathcal{I}_t} x_i^t = x_t \] \quad \forall t \in \mathcal{T}

\[ \sum_{t \in \mathcal{T}} x_t = G \] \quad \{[\hat{u}]\}

\[ 0 \leq x_i^t \leq q_i^t - q_i^{t-1} \] \quad \forall i \in \mathcal{I}, \forall t \in \mathcal{T}

\[ 0 \leq x_i^1 \leq q_i^1 \] \quad \forall t \in \mathcal{T}


Here $I_t$ is the index set that corresponds to optimal consumption values (and their corresponding $u$ value) in the U-C curve for each trading period $t \in T$ (edge of U-C curve’s steps in Figure 6.3). Once we have the U-C curves for each time period, solving the LP above has a trivial computational cost. When $\hat{G} \in G$ the optimal consumption values in [HEU] are are also optimal for [CM-MIP]. On the other hand, when $\hat{G} \notin G$ the optimal consumption values form a feasible solution to [CM-MIP]. However, [HEU]’s solution has advantages over other feasible consumption values. Suppose $\delta = \min_{G' \in G} \{ \hat{G} - G' \mid \hat{G} - G' > 0 \}$. In other words, in [HEU], we optimize time period(s) to which the consumption quantity $\delta$ should be allocated.

When we use [HEU] to solve our model with several time periods and total consumption value $\hat{G}$, many of time periods’ consumption values remain the same as the [U-MIP] model’s optimal consumption values for the marginal utility value $\hat{u}$. Thus we can use the stored optimal ILR quantity value of the [U-MIP]$_{t, \hat{u}}$ model for the time periods $t$ whose optimal consumption values remained unchanged. For those time periods $t$, whose optimal consumption values in [HEU] are different from that of [U-MIP]$_{t, \hat{u}}$, we solve [U-MIP]$_{t, \hat{u}}$ again but with a constraint on its consumption level, forcing it to be equal to the optimal value in [HEU], therefore we can determine the updated optimal ILR quantity for that time period.

Note that by increasing the number of stages, the relative gap between the total consumption values in $G$ decreases, which results in a lower relative optimality gap. In the next section we present bounds on gaps between the optimal solution and the heuristic policy.

### 6.7 Bounds

In order to demonstrate how close the optimal solution to the proposed heuristic solution is, we present upper and lower bounds on the optimal solution of [CM-MIP] (denoted by $Z(G)$). When $G_i \leq \hat{G} \leq G_{i+1}$, and $G_i, G_{i+1} \in G$, we have $Z(G_i) \leq Z(\hat{G}) \leq Z(G_{i+1})$. Without loss of generality we assume that $G_2 \leq \hat{G} \leq G_3$, and $G = \{G_1, G_2, \ldots, G_5\}$. Therefore the maximum optimality gap (for the problem with $G = \hat{G}$) is equal to $(Z(G_3) - Z(G_2))$. The red box in Figure 6.4 shows the proposed bounds in this case.

Given the structure of this problem, no better bounds can be derived without problem-specific information (prices and tranche widths etc.). The quality of these bounds depends on how small the gaps are between the total consumption quantities. When the time periods are not identical, the increase in number of time periods can lead to smaller gaps. In the case that the time horizon includes several identical time periods, the gaps
can stay large, but by using the heuristic method we can find optimal solution in between the gaps \((Z(\hat{G}_1))\) and \((Z(\hat{G}_2))\) in Figure 6.5). In the following corollary we prove that any solution of \([HEU]\) in which each \(x_t = \sum_{i \in I} x^i_t\) is equal to a \(q_t\) in the corresponding time period (a vertex solution), is an optimal solution to our original cost-minimizing problem.

**Corollary 6.3.** *Any optimal vertex solution to [HEU] is an optimal solution to [CM-MIP].*

**Proof.** Any optimal vertex solution to [HEU] yields a total consumption value \(G := \sum_{t \in T} x^*_t\). Lemma 6.1 then establishes the result that \(x^*\) solves \([CM-MIP]\). □

This means that if the total consumption quantity \(\hat{G}\) is not in \(G\) and it is used as the total consumption parameter in [HEU] and returns a vertex solution, then this solution solves [CM-MIP]. Furthermore, from Corollary 6.3, we have the statement below:

There exists an ordered set \(G' = \{G_1, G_2, ..., G_m\}\) whereby [HEU] returns a vertex solution. Since we know that the maximum gap between any pair of total consumption values in \(G'\) is the maximum consumption in one time period \((G_{i+1} - G_i \leq \max\{q_j\})\), this means that, for large total consumption values (e.g. for optimization over long time-horizons), the size of this approximation as a percentage of the total consumption becomes insignificant.

### 6.8 U-C Algorithm

In this section we put the steps developed in the previous sections together in an algorithm to produce a (near) optimal consumption schedule for the large consumer of interest.

**Deterministic U-C algorithm: DUCA**
1. **Construct individual TP (Time Period) U-C curves** \( g(u, t) \)
   Solve \([\text{MIP}]_{t,u}\) to find \( x_t = x_t^* \) set \( g(u, t) = x_t^* \), \( \forall u \in [0, u_{\text{max}}], \forall t \in T \).

2. **Make the aggregated U-C curve** \( G(u) \)
   \[ G(u) = \sum_{t \in T} g(u, t), \quad \forall u \in [0, u_{\text{max}}]. \]

3. **For a given total consumption value** \( \hat{G} \), **find a corresponding** \( u \) **value**
   If \( \exists \hat{u} \in [0, u_{\text{max}}] \) where \( G(\hat{u}) = \hat{G} \):
   - Directly use \( \hat{u} \) value to determine optimal consumption values at each TP
     \[ \hat{x}_t = g(\hat{u}, t), \quad \forall t \in T. \]
   Else:
   - Use the heuristic model to determine consumption values at each TP
     Solve \([\text{HEU}]\) with \( G = \hat{G} \). If \( x_t^* \) is the optimal consumption values of \([\text{HEU}]\)
     \( \rightarrow \hat{x}_t = x_t^* \) \( \forall t \in T \).

4. **Return consumption values and corresponding ILR quantities**
   For each time period return \( \hat{x}_t \).

Note that the construction methodology for our aggregate U-C curve ensures that, in step 3 of the algorithm above, we choose an optimal, or near optimal, consumption level.

In Chapter 7, we generalize this to the stochastic expected U-C curves construction algorithm.

### 6.9 Multistage Co-optimization Example

In order to demonstrate the impact of multistage co-optimization, we lay out an illustrative example over a 3-node network. The 3-node network illustrated in Figure 6.6 contains a loop (impedance equals 1.0 for all arcs) with buses A, B and C. In this example we assume that there are two generators, one at bus B, and one at bus C. We also assume there is one inelastic consumer at bus C, and one strategic consumer at bus A.

The line capacities are shown in Figure 6.6.

The maximum amount of aggregated reserve and energy available from generators B and C are 50 and 75 MW, respectively. Table 6.1 presents the generators’ offer stacks data.

---

3The optimal ILR quantity is computed simultaneously with the optimal consumption value for each \( u \) at time period \( t \). These values can be stored to be used in this step of the algorithm.
In this example, we lay out a multistage model which is solved over twelve time periods, where the time horizon starts with off-peak time periods, then it transitions to shoulder (i.e. medium consumption), peak and again shoulder trading periods, respectively. Here we assume that the difference in time periods is only in the inelastic demand at bus C and the reserve requirement, while the energy and reserve offer stacks are identical in all time periods. Table 6.2 presents the values of demand and reserve parameters associated with each time period.

<table>
<thead>
<tr>
<th>Time Period</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bus C Inelastic Demand (MW)</td>
<td>13</td>
<td>18</td>
<td>14</td>
<td>23</td>
<td>27</td>
<td>28</td>
<td>52</td>
<td>53</td>
<td>53</td>
<td>23</td>
<td>27</td>
<td>21</td>
</tr>
<tr>
<td>Reserve Requirement (MW)</td>
<td>18</td>
<td>21</td>
<td>17</td>
<td>36</td>
<td>31</td>
<td>39</td>
<td>51</td>
<td>69</td>
<td>70</td>
<td>35</td>
<td>42</td>
<td>36</td>
</tr>
</tbody>
</table>

Table 6.2: Inelastic demand and reserve levels

We assume that the major consumer is required to consume a total of 240 MWh of electricity over the time horizon. In order to probe the impacts of co-optimization of energy and reserve we present two versions of our model. Firstly, we consider a major consumer who chooses its consumption quantity, and has the ability to offer ILR. Secondly, we consider a major consumer who only decides on its consumption level and cannot offer ILR; we call these two versions v1 and v2, respectively. Utilizing our decomposition method, we build the individual trading period U-C curves, and subsequently, the aggregated U-C curves for v1 and v2 (Figure 6.7).
Utilizing the U-C curves above, we can derive the U-C policy as well as the heuristic method that was presented in Section 6.6. In Table 6.3 we lay out the consumption level values of our proposed policies for both v1 and v2. Note that the solution of normal U-C policy is to over-consume energy; the heuristic policy however, aims to consume the exact amount of energy throughout the time horizon (the instances when consumption levels differ between the two policies are highlighted in red).

<table>
<thead>
<tr>
<th>Time Period</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>U-C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>v1 243</td>
</tr>
<tr>
<td>Heuristic</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>v1 240</td>
</tr>
<tr>
<td>U-C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>v2 259</td>
</tr>
<tr>
<td>Heuristic</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>v2 240</td>
</tr>
</tbody>
</table>

**Table 6.3:** Consumption levels for normal U-C policy vs heuristic policy

From the policies that are presented in Table 6.3, we can observe that in peak time periods (7–9), the strategic consumer’s demand is higher in v1 compared to v2. This is because in v1 with higher consumption levels the strategic consumer, capable of offering ILR, can offer more ILR and take advantage of high reserve prices at peak time periods. On the other hand, in v2, high peak prices push the consumption down and lead to more energy consumption in off-peak time periods.

In Figure 6.8 we depict the realized energy and reserve prices, for both v1 and v2, that are derived from the heuristic U-C policy. In addition, we compare our consumption policies
with a fixed policy. We report on the fixed consumption policy in which the strategic
agent consumes 20MWh at each trading period (and co-optimizes ILR, similar to v1).
In Figure 6.8, we also demonstrate the realized energy prices of the fixed consumption
model for v1 at each trading period.

![Figure 6.8: Energy and reserve prices in v1 and v2](image)

As shown in Figure 6.8, compared to the fixed consumption policy, our U-C policies
prevent high energy prices at peak time periods (7–9) and consume more at off-peak
trading periods (1–3). In addition, reserve prices in v2 are never less than v1 meaning
that offering ILR by the consumer may lower reserve prices. In order to further compare
our policy’s performance we compare the results with the optimal solution that is derived
from directly solving [CM-MIP] for v1 and v2. In Table 6.4, we present the total cost
of implementing each presented policy.

<table>
<thead>
<tr>
<th>Policy</th>
<th>Model</th>
<th>Total cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>Heuristic</td>
<td>v1</td>
<td>2617.6</td>
</tr>
<tr>
<td>Optimal</td>
<td>v1</td>
<td>2604.1</td>
</tr>
<tr>
<td>Heuristic</td>
<td>v2</td>
<td>3494.5</td>
</tr>
<tr>
<td>Optimal</td>
<td>v2</td>
<td>3476.5</td>
</tr>
</tbody>
</table>

**Table 6.4: Total cost comparison**

### 6.9.1 Multistage Co-optimization with CfD

In this subsection we lay out an example in which a strategic major consumer, who is
able to bid energy and offer ILR, also participates in the hedge market through having a
CfD. In order to comprehensively track the change in the strategic behavior when a CfD
is introduced to the model, we use the same 3-node setting as was laid out in Section
6.9. In our CfD model the objective function is changed to contain the CfD payments;
we call the new objective function $z'$ as in (6.2).
\[ z' = U(y^d) - \pi^d y^d + \pi^r y^r + (\pi^d - \pi^c) y^c \]  \hspace{2cm} (6.2)

In this example we have set the contract quantity \( y^c = 20 \) and the contract price \( \pi^c = 15 \).

In Table 6.5 we demonstrate the optimal consumption, ILR, energy price and reserve price values for the Section 6.9’s example with and without CfD. Note that, in order to accurately compare the two models (with CfD and without CfD), we have obtained the optimal values from directly solving the [MIP] model over the time horizon.

<table>
<thead>
<tr>
<th>Time Period</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal ( y^d )</td>
<td>17</td>
<td>19</td>
<td>16</td>
<td>29</td>
<td>25</td>
<td>24</td>
<td>10</td>
<td>9</td>
<td>9</td>
<td>26</td>
<td>25</td>
<td>31</td>
</tr>
<tr>
<td>Normal ( y^r )</td>
<td>17</td>
<td>19</td>
<td>16</td>
<td>26</td>
<td>21</td>
<td>24</td>
<td>10</td>
<td>9</td>
<td>9</td>
<td>25</td>
<td>25</td>
<td>26</td>
</tr>
<tr>
<td>CfD ( y^d )</td>
<td>18</td>
<td>21</td>
<td>17</td>
<td>26</td>
<td>21</td>
<td>24</td>
<td>13</td>
<td>12</td>
<td>12</td>
<td>25</td>
<td>25</td>
<td>26</td>
</tr>
<tr>
<td>CfD ( y^r )</td>
<td>18</td>
<td>21</td>
<td>17</td>
<td>26</td>
<td>21</td>
<td>24</td>
<td>11</td>
<td>9</td>
<td>10</td>
<td>25</td>
<td>25</td>
<td>26</td>
</tr>
<tr>
<td>Normal ( \pi^d )</td>
<td>10</td>
<td>13</td>
<td>10</td>
<td>15</td>
<td>15</td>
<td>15</td>
<td>17.5</td>
<td>17.5</td>
<td>17.5</td>
<td>15</td>
<td>15</td>
<td>15</td>
</tr>
<tr>
<td>Normal ( \pi^r )</td>
<td>2.5</td>
<td>2.5</td>
<td>2.5</td>
<td>3.5</td>
<td>3.5</td>
<td>3.5</td>
<td>3.5</td>
<td>7.8</td>
<td>7.8</td>
<td>7.8</td>
<td>3.5</td>
<td>3.5</td>
</tr>
<tr>
<td>CfD ( \pi^d )</td>
<td>13</td>
<td>15</td>
<td>13</td>
<td>15</td>
<td>15</td>
<td>15</td>
<td>314.5</td>
<td>314.5</td>
<td>314.5</td>
<td>15</td>
<td>15</td>
<td>15</td>
</tr>
<tr>
<td>CfD ( \pi^r )</td>
<td>2.5</td>
<td>2.5</td>
<td>2.5</td>
<td>3.5</td>
<td>3.5</td>
<td>3.5</td>
<td>300</td>
<td>300</td>
<td>300</td>
<td>3.5</td>
<td>3.5</td>
<td>3.5</td>
</tr>
</tbody>
</table>

**Table 6.5: Consumption levels for normal policy vs CfD policy**

As shown in Table 6.5, the CfD policy would consume more than the normal policy in peak trading periods (TP7–TP9), and as a result the peak prices are much higher in the CfD model. The large difference between the spot price \( y^d \) and the contract price in the peak time periods increases the CfD payments, in addition, the higher reserve prices increase the ILR revenue for the strategic consumer.
6.10 Conclusions

In this chapter, we have developed a model that solves a large-scale extended time-horizon strategic electricity consumption scheduling problem. The major consumer in this model purchases electricity from the spot market and is also able to offer ILR. We assumed that in addition to co-optimizing energy and reserve, the major consumer has a total consumption requirement over a time horizon. Initially, we analyzed the price-taking consumer model, and by using this convex problem’s intuition, we probed the properties of a price-making major consumer (modeled via a MIP), who observes the impacts of its decisions on the market while meeting a total production (consumption) requirement.

In order to be able to solve such a large MIP over an extended time-horizon, we used Lagrangian methods to decompose the multistage stochastic demand and ILR co-optimization. We proved that our decomposition method obtains the optimal solution for a given set of total consumption values. However, in order to expand our solution for all possible total consumption values we presented a heuristic method that yields near optimal solutions for our problem. Furthermore, we computed upper and lower bounds for our proposed approximate solutions.

In order to utilize our decomposition method to solve our multistage problem, we introduced the notion of U-C curves as look-up tables. Our method determines the consumer’s optimal demand and ILR offer by adjusting the marginal value of consuming electricity to the type of trading period.

Moreover, we laid out a comprehensive example to probe the impacts of co-optimizing energy consumption and ILR offer in a multistage setting. We compared the aggregated U-C curve of an energy-only setting with an energy and ILR co-optimized model. In addition, we compared the consumption values in our proposed U-C policy with a fixed consumption policy. We observed that the peak trading periods’ energy prices in the U-C policy are lower than those of fixed policy energy prices. Furthermore, we discussed the inclusion of CfDs in the major consumer’s objective function, and extended our example to a multistage CfD model. By comparing the optimal consumption values in CfD and non-CfD model, we demonstrated that a CfD can change the strategic behavior, particularly in peak trading periods. We observed that the consumer may choose to consume high levels of energy in order to increase its CfD earnings and this can lead to price spikes at peak trading periods.

In this chapter we assumed deterministic market parameter values, however, in the real world, we would only have estimates of such values. In order to account for the
uncertainty, in Chapter 7, we will develop mathematical models that provide stochastic multistage decision making tools for major consumers of energy.
Chapter 7

Stochastic Strategic Consumption over a Finite Time Horizon

In Chapter 6, we addressed the deterministic multi-stage strategic consumption, and presented a decomposition algorithm based on Lagrangian methods. In this chapter, we introduce uncertainty into our model and adjust our solution methods to account for stochasticity in our optimization problem.

We represent our model with a scenario tree, in order to account for different realizations of market conditions at each decision-making stage. In order to properly model uncertainty, it is necessary to recognize how the uncertainty is revealed with respect to the timing of decisions (i.e. wait-and-see and here-and-now) in the optimization model. In this chapter, we examine how uncertainty in our model parameters affects the optimal consumption/ILR policy. Specifically, we present a hybrid stochastic modeling scheme, accounting for both short-term and medium-term variations in market conditions. In addition, we focus on methods that enable us to reduce the size of our scenario tree while maintaining the important information. In order to facilitate this, we lay out a problem-specific scenario clustering scheme which we implement on a large-scale real-world problem.

7.1 Introduction

The size of a multi-stage stochastic program’s scenario tree increases exponentially with respect to the length of the time horizon, thus large-scale realistic models, may well become computationally intractable. One of the methods used to tackle the curse of dimensionality is to utilize scenario clustering techniques to reduce the size of the scenario
tree. In this section, we briefly address the existing literature on scenario tree reduction for power system models and present a tailor-made clustering method in order to make our multi-stage stochastic optimization model tractable.

Generating scenarios that accurately represent a stochastic process is an important component of a realistic multi-stage model. Multivariate regression models have been widely used for estimating distributions for water flow, demand and electricity prices over a time period, in order to optimally manage hydro-thermal generation systems [192–194]. In these models, many sample paths of the historical data are fitted to (multi)normal probability distributions.

On the other hand, there exists a large number of stochastic processes with probability distributions that cannot easily be described by standard parametric models. In order to address these types of distributions, empirical methods are used for scenario generation and the corresponding probabilities are derived from the historical data [195, 196]. Non-parametric stochastic models are also utilized for electric power system models. For instance, in [197], electricity demand scenarios are generated for a unit commitment generation model.

There exists extensive research on sampling methods of multi-stage models, such as studies presented in [198, 199]. The Monte Carlo sampling algorithm is the most commonly used method for generating random samples for stochastic processes of large-scale stochastic programming problems. Given such a random sample, the expected value function of the model can be approximated by its sample average function; this is known as the Sample Average Approximation (SAA) method [200]. In [201], a Monte Carlo simulation approach for stochastic discrete optimization problems is developed, using the SAA concept; however, this is only applied to two-stage models. In addition, Shapiro et al. use a deterministic-equivalent SAA approach, in conjunction with some variance reduction techniques, to converge to solutions to two-stage problems [202]. However, they show that this approach is difficult to extend to a multi-stage setting.

One of the main questions that needs to be addressed, when constructing a sample based on a real-world multistage problem, is how many sampled scenarios are required in order to adequately represent the parametric or non-parametric stochastic process. In [203], this problem is addressed through developing sequential importance sampling schemes for dynamic stochastic programs based on expected value of perfect information (EVPI) processes. Subsequent to generating sufficient data-paths for a stochastic model, the scenario tree might need to be reduced so that the problem can be solved in a practical time-frame.
There exists a wide range of developed work on scenario tree reduction for linear multistage models. One of the most common approaches is through generating a subset of scenarios with a probability measure that is the closest to the initial distribution of scenarios. For instance, in [204, 205], by utilizing the Kantorovich metric and the notion of representative scenarios, near-optimal reduced models are presented. Moreover, in [206, 207], efficient algorithms are developed that determine approximately optimal solutions using similar reduced probability measures. Moreover, the use of reduced probability measures for convex stochastic models is extended for two-stage integer models in [208].

In addition to modeling the scenario reduction policies, the detailed consequences of such data processing algorithms on the quantitative stability of linear multistage stochastic programs are studied [199, 209]. Moreover, in [210], a stability-theory based heuristic model is presented that utilizes forward or backward algorithms for generating recursively reduced scenarios trees.

In the discussed sample processing methods that are applied to multi-stage problems, the stochastic process models a time-dependent exogenous parameter over the time horizon. For instance, demand or price processes are most commonly used in power system models, where the exogenous electricity prices or demand levels are approximated through representative scenarios along the time horizon. However, in our model, given the price-making nature of the studied agent, the possible future scenarios cannot be laid out through approximating a few exogenous parameters, and the accuracy of such parametric stochastic process would not be sufficient for our model. Therefore, we present a tailor-made clustering method in order to address the uncertainty within our model’s endogenous pricing process.

Chapter Structure

We start by setting out the mathematical program corresponding to our stochastic multistage model in Section 7.2. Next, by extending our deterministic decomposition method, we present stochastic solution methodologies for our mathematical model in Section 7.3. Similar to the deterministic case, we start by setting out a price-taker model’s Lagrangian relaxation method and develop a similar methodology for our price-making major consumer model. In Section 7.4 we construct the expected U-C curves with both stage-wise dependent and independent uncertainty. In addition, similar to the deterministic model, we present a heuristic model which enables us to utilize the data from expected U-C curves for all possible total consumption values. In the second part of this chapter we focus on modelling uncertainty. In Section 7.5, we present a
hybrid stochastic model which combines wait-and-see and here-and-now decision making properties. In addition, we reduce the size of our model’s scenario tree by utilizing a tailor-made scenario clustering method. Furthermore, in Section 7.6, we conduct an experiment for a major consumer in New Zealand Electricity Market and report on numerical results. Section 7.7 summarizes the chapter.

Specific Chapter Nomenclature

Scenario Tree Formulation

$\mathcal{T}$ The set of all time periods $t \ (\mathcal{T} = \{1, 2, \ldots, T\})$.

$\rho_n$ The probability corresponding to node $n$ in the scenario tree.

$\rho_{n,l}$ The probability of reaching node $l$ from node $n$ in the tree.

$C_n$ The cost function at node $n$ in the scenario tree.

$\mathcal{N}$ The set of all nodes in the scenario tree.

$\mathcal{N}_t$ The set of nodes in time period $t$.

$\mathcal{S}_n$ The feasible set of actions at node $n$.

$P_m$ The vector of nodes that form the path that terminates at leaf node $m$.

$x_n$ The electricity consumption at node $n$ of the tree.

$x_n$ The vector of actions at node $n$ of the tree.

Stochastic U-C Curves Formulation

$u_n(q)$ The marginal value of electricity corresponding to consumption value $q \in [0, q^{\text{max}}]$, based on the U-C curve associated with the node $n \in \mathcal{N}_t$.

$g(u, n)$ The consumption value ($q$) at marginal utility $u$, based on the U-C curve associated with the node $n \in \mathcal{N}_t$.

$\Xi_n$ The set of all consumption values that shape the vertical steps in the U-C curve associated with the node $n \in \mathcal{N}_t$.

$Q_n'$ The set of optimal (corner point) consumption-to-go values at node $n \in \mathcal{N}_t$.

$Q_n'$ The set of optimal consumption-to-go values at stage $t$, where $Q_n' = \bigcup_{n \in \mathcal{N}_t} Q_n$.

$\mathcal{K}_n$ The set of future consumption to go and present node’s $(n)$ optimal consumption values that have equal marginal cost values.
Stage-wise independent

\( u'(q) \)  The set of \( u \) values that correspond to optimal (corner point) future consumption-to-go values \( (q \in Q^t) \) that are observed by each node at stage \( t - 1 \).

Stage-wise dependent

\( u'_n(q) \)  The set of \( u \) values that correspond to optimal (corner point) future consumption-to-go values \( (q \in Q^t) \) that are observed by node \( n \) at stage \( t - 1 \).

Scenario Clustering Formulation

\( \mathcal{L} \)  The set of all designated consumption levels \( l \).

\( \pi^\omega_t(l) \)  The energy price corresponding for each consumption level \( l \in \mathcal{L} \), in scenario \( \omega \), and stage \( t \).

\( r \)  Indicative value of each price range \( p_r \).

\( \Gamma \)  The set of all types of stages \( \gamma \).

\( c^\omega_t \)  A vector demonstrating the cluster of scenario \( \omega \) at stage \( t \).

\( C_\gamma \)  The set of clusters for each type of time period \( \gamma \).

### 7.2 Problem Structure

Similar to the deterministic case, we seek to minimize expected total cost of consumption over the time horizon. In order to facilitate mathematical exposition, we define a scenario tree [210] and we use the **Scenario Tree Formulation** notation to index variables by their corresponding node in the tree. We denote this stochastic problem \([S-MIP]\); this optimization problem seeks to minimize expected cost over all paths of the scenario tree \((P_m)\) as given below:

\[
[S-MIP] \quad \min \sum_{t \in T} \sum_{n \in N_t} \rho_n C_n(x_n) \\
\text{s.t. } x_n \in X_n \quad \forall n \in N_t, \forall t \in T \\
\sum_{n \in P_m} x_n = G \quad \forall m \in N_T
\]
Given the size and complexity of the MIP above, in order to solve this problem to global optimum, we need to decompose our model. In the next section, similar to the deterministic case, we first seek insight from the price-taking model and by taking advantage of the unique properties of our MIP, we prove that we can use a similar decomposition method for the stochastic price-making model under certain circumstances.

### 7.3 Stochastic Decomposition

We can extend the decomposition methodology of the deterministic case here, but in addition to decomposition over time periods, we need to also decompose the problem across all the tree nodes at each time period.

![Nodes in the tree](image1)

![Paths of the tree](image2)

In order to implement the optimal consumption strategy, the stochastic and non-anticipative nature of the scenario tree must be respected. We will therefore need suitably aggregated U-C curves that encode this information. To do so, we are guided by the principle that the optimal value of consumption in each period is precisely the quantity that will yield a marginal cost of consumption now, equal to the expected marginal cost of consumption in the future in presence of convexity. We illustrate this principle initially on the convex problem that captures the price-taking consumer’s planning problem, and extend it to the price-making model.

#### 7.3.1 Price Taker Consumer

We start modelling uncertainty by a wait-and-see approach to the stage-problem, where the realized node (scenario) is known before taking an action, although the future is
Chapter 7 Stochastic Strategic Consumption over a Finite Time Horizon

uncertain. Recall the price-taking consumer from Section 6.4.1, but now in a multistage setting, formulated as a stochastic dynamic program and in the absence of any integrality requirements. Utilizing a backward induction scheme, we lay out the following stochastic LP algorithm (SLP-A) to define each node’s \((n \in \mathcal{N}_t)\) optimization problem, denoted by \([D-LP]_t^n\). Here, at each stage \(t\) the linear program \([D-LP]_t^n\) minimizes cost, with total consumption to go \((g_t)\) as the state variable.

\textbf{SLP-A}

For period \(T\) and any leaf node \(m \in \mathcal{N}_T\) we have:

\[
[D-LP]_T^m C_T^m(g_T) = \min C(x_T^m) \\
\text{s.t. } x_T^m = g_T. \\
[u_T^m]
\]

For each period \(t \in \{1, 2, ..., T - 1\}\) with any node (in the scenario tree) \(n \in \mathcal{N}_t\) we have:

\[
[D-LP]_t^n C_t^n(g_t) = \min C(x_t^n) + \sum_{l \in \mathcal{N}_{t+1}} \rho_{n,l} C_{t+1}^l(g_{t+1}^n) \\
\text{s.t. } x_t^n + g_{t+1}^n = g_t. \\
[u_t^n]
\]

Similar to the deterministic \([LP]\), we define the multiplier on the total consumption constraint, \(u_t^n\) which is indexed by stage \(t\), and node \(n\). We aim to use the properties of this parameter in the price maker model to extract optimal policies. Hence we examine the role of this parameter in the convex setting first.

\textbf{Theorem 7.1.} Consider \(u_t^n\), the dual on the consumption constraint at stage \(t\) defined in \([D-LP]_t^n\). Then

\[
\frac{\partial \sum_{l \in \mathcal{N}_{t+1}} \rho_{n,l} C_{t+1}^l(g_{t+1}^n)}{\partial g_{t+1}^n} = u_t^n, \text{ and} \\
\frac{\partial C(x_t^n)}{\partial x_t^n} = u_t^n.
\]

\textbf{Proof.} Simply apply optimality conditions to \([D-LP]_t^n\) to obtain the above. \(\square\)

The equations above show that \(u_t^n\) can be used to control the optimality condition, which indicates that at each node of the tree the marginal cost equals marginal cost-to-go. In the next part, we use this property, to define the marginal utility vs. marginal cost relationship in the price maker model. This approach is explained in detail in section 7.4, but prior to that we make a short detour to deal with the discrete nature of the optimal consumption quantities.
7.3.2 Price Maker Consumer

In the price maker stochastic model, as well as the deterministic version, we are faced with the problem of corner point solutions. Here, the decomposed model’s optimal solution is only equivalent to that of the original problem for a set of discrete total consumption levels, where there exists a different set for each path \( m \in \mathcal{N}_T \). In order to address this issue, we introduce \([S\text{-MIP-}\epsilon^+]\), where \( \epsilon^+_m \geq 0 \) is added to the total consumption on each path. We will focus attention on \([S\text{-MIP-}\epsilon^+]\), instead; although \([S\text{-MIP-}\epsilon^+]\) is not exactly equivalent to \([S\text{-MIP}]\), the deviation of its optimal solution from that of \([S\text{-MIP}]\) is small enough for the purposes of our model (large total consumption value over a long time-horizon).

\[
[S\text{-MIP-}\epsilon^+] \quad \min \sum_{t \in T} \sum_{n \in \mathcal{N}_t} \rho_n C_n(x_n) \\
\text{s.t.} \quad \sum_{n \in \mathcal{P}_m} x_n = G + \epsilon^+_m \quad \forall m \in \mathcal{N}_T \\
x_n \in \mathcal{S}_n \quad \forall n \in \mathcal{N}_t, \forall t \in T.
\]

The next step is to use Lagrangian methods to decompose \([S\text{-MIP-}\epsilon^+]\). We define \([L\text{-MIP}]_\mu\), where \( \mu \) is the vector of Lagrangian multipliers (\( \mu_m \) is the Lagrange multiplier on path \( m \)'s total consumption constraint). Furthermore, in the objective function of \([L\text{-MIP}]_\mu\), the total consumption constraint corresponding to each path \( m \) is weighted with the probability of occurrence of path \( m \) (\( \rho_m \)). Note that \([L\text{-MIP}]_{x,\mu}\) is decomposable by time-period.

\[
[L\text{-MIP}]_\mu \quad \min L(x, \mu) = \sum_{n \in \mathcal{N}} \rho_n C_n(x_n) - \sum_{m \in \mathcal{N}_T} \sum_{n \in \mathcal{P}_m} \rho_m \mu_m x_n + \sum_{m \in \mathcal{N}_T} \sum_{n \in \mathcal{P}_m} \rho_m \mu_m (G + \epsilon^+_m) \\
\text{s.t.} \quad x_n \in \mathcal{S}_n \quad \forall n \in \mathcal{N}.
\]

Here, the last term in the objective function is constant and therefore will not impact the optimal consumption values. If we solve \([L\text{-MIP}]_\mu \quad \forall \mu_m \in (0, \mu^{\text{max}}_m)\), we obtain a set of corresponding total consumption values for each path \( m \), hence we can find the smallest total consumption \( \hat{G}_m \), where \( \hat{G}_m = G + \epsilon^+_m \), \( \epsilon^+_m \geq 0 \). Here, \( \hat{G}_m \), and therefore \( \epsilon^+_m \) are functions of \( \mu_m \). Note that the structure of our model for each path guarantees that \( \epsilon^+_m < C \), where \( C \) is the maximum consumption at one time period (as shown for the deterministic case in Section 6.7).
Lemma 7.2. Let \( \mathbf{x}^* \) be the vector of optimal consumption values of \([L-MIP]\) such that 
\[
\sum_{n \in P_m} x^*_n := G + \epsilon^*_m \text{ and } \epsilon^*_m \geq 0.
\]
Then \( \mathbf{x}^* \) solves \([S-MIP-\epsilon^+]\).

Proof. We use the Lagrangian sufficiency theorem (presented in Section 2.5) to prove that \( \mathbf{x}^* \) is optimal for \([S-MIP-\epsilon^+]\). First, note that the feasible regions of both optimization problems are identical. Hence \( \mathbf{x}^* \) is a feasible solution of \([S-MIP-\epsilon^+]\), and 
\[
L(\mathbf{x}^*, \mu^*) \leq L(\hat{\mathbf{x}}, \mu^*), \forall \hat{\mathbf{x}} \mid \hat{\mathbf{x}}_n \in S_n.
\]

In the next step we calculate the expected value of \( u \) parameter (that is used in the aggregated U-C curve), based on indexing the problem by paths, whereby we have:

\[
\sum_{n \in N} \rho_n u^*_n x^*_n = \sum_{m \in N_T} \sum_{n \in P_m} \rho_n u^*_n x^*_n = \sum_{m \in N_T} \sum_{n \in P_m} \rho_m \mu_m x^*_n. \tag{7.1}
\]

Hence we can calculate the optimal marginal utilities, \( u^*_n \), that meet (7.1).

\[
\sum_{n \in N} \rho_n u^*_n x^*_n = \sum_{m \in N_T} \sum_{n \in P_m} \rho_m \mu_m x^*_n = \sum_{n \in N} \sum_{m \in N_T | n \in P_m} \rho_m \mu_m x^*_n \Rightarrow u^*_n \left( \sum_{m \in N_T | n \in P_m} \rho_m \right) = \sum_{n \in N} x^*_n \left( \sum_{m \in N_T | n \in P_m} \rho_m \mu_m \right) \Rightarrow u^*_n = \frac{\sum_{m \in N_T | n \in P_m} \rho_m \mu_m}{\sum_{m \in N_T | n \in P_m} \rho_m}
\]

In the next section, we will lay out the construction of the expected U-C curves with an algorithm based on the equation above.

### 7.4 Expected Utility Consumption Curves

In this section, by utilizing the intuition from the properties of the Lagrange multipliers in price taker model, and Lemma 7.2, we build expected aggregate U-C curves to find (near) optimal stochastic policies. We denote by \( u_n(q) \), the marginal value of electricity consumption at level \( q \), at the tree node \( n \) (note that \( u \) is a set valued map and for some \( q_s \) there is an interval of corresponding marginal costs \( u \)). Similarly, we define the function \( g(u, n) \) which indicates the consumption value \( (q) \) at node \( n \), with marginal utility \( u \). As discussed above, in order to make the optimal choice of consumption in a given period, we must know the expected cost of future consumption. Clearly, this must be reflected through a single (expected aggregated) U-C curve, as we can not anticipate
the future. In the last time period, this task is straightforward, the expected U-C curve for trading period \((t = T)\) is constructed by assigning \(\sum_{n \in \mathcal{N}_t} \rho_n u_n(q)\) to quantity \(q\) (see Figure 7.3). This step is referred to as “vertical expectation”, and the resulting curve as expected cost to go curve.

\[
\hat{u}_2 \\
\hat{u}_1
\]

\[
\nu_n(q)
\]

\[
\nu_n(q) + \rho u_n(q)
\]

**Figure 7.3:** Vertical expectation step of constructing aggregated expected U-C curves

By stepping backward, for each node \(n \in \mathcal{N}_t\) in the current stage \((t = T - 1)\), we are in a position to map out the marginal cost of consumption for any quantity \(q\) by assessing which is the cheaper of the two options, namely consume now or leave it for the future. This process will naturally deliver updated U-C curves for each scenario in the current period as depicted in Figure 7.4. We label this procedure “the horizontal aggregation”. Once the updated curves of all nodes of stage \(t\) are available, we proceed to the vertical expectation step, which results in the expected cost to go curve for period \(t - 1\). We do this recursively as outlined in the SUC-construction algorithm below. At the end of the algorithm, we are equipped with a look-up table for optimal consumption at each stage.

\[
\nu_n(q)
\]

**Figure 7.4:** Horizontal aggregation step of constructing aggregated expected U-C curves

---

1For the sake of simplicity, we start our U-C curve construction method with a stage-wise independent setting, and then will lay out our stage-wise dependent method. Note that the assumption of stage-wise independence, depending on the model, might interfere with the scenario tree structure. We acknowledge that for this model stage-wise dependency is an oversimplification, therefore, our case-study simulations are based on our stage-wise dependent model.
Given the step-wise form of the U-C curves, we can implement the above procedures in a relatively computationally inexpensive manner, by first acquiring the set of \( q \) values at the edge of each step. We can find the corresponding \( u \) values for all \( q \) levels between two consecutive edges. Let \( \Xi_n \) denote the set of all consumption values that shape the vertical steps in the U-C curve of node \( n \). We also define the set of optimal (corner point) consumption-to-go values at node \( n \) as \( \mathcal{Q}_n \) (note that for \( t = T \), \( \mathcal{Q}_n = \Xi_n \)). We also denote the set of optimal consumption-to-go values at stage \( t \) by \( \mathcal{Q}^t = \bigcup_{n \in \mathcal{N}_t} \mathcal{Q}_n \).

**SUC-C-I**

1. Set \( t = T \) and compute \( u^t(q) = \sum_{n \in \mathcal{N}_t} \rho_n u_n(q) \), \( \forall q \in \mathcal{Q}^t \) (Figure 7.3).
2. \( t \rightarrow t - 1 \).
3. \( \forall n \in \mathcal{N}_t: \mathcal{K}_n := \{ (q_t, \hat{q}_{t+1}) | u_n(q_t) = u^{t+1}(\hat{q}_{t+1}), \forall q_t \in \Xi_n \cup \mathcal{Q}^{t+1} \} \) (Figure 7.4).
4. \( \forall n \in \mathcal{N}_t: \nu_n := u_n(q_t) | q = q_t + \hat{q}_{t+1}, \forall (q_t, \hat{q}_{t+1}) \in \mathcal{K}_n \) (Figure 7.4).
5. \( u^t(q) := \sum_{n \in \mathcal{N}_t} \rho_n \nu_n(q) \), \( \forall q \in \mathcal{Q}^t \) (Figure 7.3).
6. If \( t > 1 \), go to step 3. If \( t = 1 \), stop.

### 7.4.1 Stage-wise dependency

In this subsection we will demonstrate that, by utilizing our proposed decomposition method, we can also model the uncertainty in our SDP as a time in-homogeneous Markovian process. In order to model the stage-wise dependency, we first define the transition probabilities\(^2\). We use the notation \( \rho_{n,l} \) to show the probability of transitioning from node \( n \) to node \( l \) in the tree. In addition, we define an artificial node \( \hat{n} \) for stage zero as \( \mathcal{N}_0 = \{ \hat{n} \} \), where \( \rho_{\hat{n},l} \) is the probability of being at node \( l \) at stage one. We use these transition probabilities to construct the U-C curves with conditional expectation. Hence, such U-C enables us to address stage-wise dependency, in transitions within trading periods as laid out below:

**SUC-C-D**

1. Set \( t = T \).

\(^2\)In Section 7.5 we will show how we calculate the transition matrix, for our model.
2. \(\forall n \in \mathcal{N}_{t-1}: u^t_n(q) = \sum_{m \in \mathcal{N}_t} \rho_{n,m} u^t_m(q), \ \forall q \in \mathcal{Q}^t.\)

3. \(t \rightarrow t - 1.\)

4. \(\forall n \in \mathcal{N}_t: \mathcal{K}_n = \{(q_t, \hat{q}_{t+1})|u_n(q_t) = u^{t+1}_n(\hat{q}_{t+1}), \forall q_t \in \Xi_n \cup \mathcal{Q}^{t+1}\}.\)

\[Q_n = \{q|q = q_t + \hat{q}_{t+1}, \forall (q_t, \hat{q}_{t+1}) \in \mathcal{K}_n\}\]

5. \(\forall n \in \mathcal{N}_t: \nu_n(q) = u_n(q_t)|q = q_t + \hat{q}_{t+1}, \forall (q_t, \hat{q}_{t+1}) \in \mathcal{K}_n\)

6. \(\forall n \in \mathcal{N}_{t-1}: u^t_n(q) = \sum_{m \in \mathcal{N}_t} \rho_{n,m} \nu_m, \ \forall q \in \mathcal{Q}^t.\)

7. If \(t > 1,\) go to step 3. If \(t = 1,\) \(u^t_n\) is the function corresponding to the aggregated expected U-C curve in node \(n,\) at stage one.

Note that similar to the deterministic setting, in the stochastic model, we are faced with the corner point solutions that form the expected aggregated U-C curves. Therefore, in the next section, we lay out a heuristic solution method that deals with the discrete nature of optimal total consumption quantities.

### 7.4.2 Heuristics

In the stochastic setting, we can also use the heuristic algorithm used in the deterministic model. We use the intuition from the market clearing problem to solve the [S-U-MIP] problem for any given \(\hat{G}.\)

At each stage of solving the dynamic programming we are faced with two curves. One is the current time-period’s utility-demand curve and the other one is the curve corresponding to the expected future. In each curve, we connect the vertical lines with horizontal lines, to turn them to step functions.

As these step-wise functions (current stage and the expected future) have the same attributes as a supply function for a generator in the dispatch model, we view this model as a two node network with no flow constraints, and assume that the total consumption value is the demand. We introduce an LP, in which we minimize utility \(\times\) consumption, given the total consumption constraint.

\[
[S\text{-HEU}] \quad \min \sum_{n \in \{\text{Now, Future}\}} \sum_{i \in \mathcal{L}_n} u^t_n x^t_n \\
\text{s.t.} \sum_{n \in \{\text{Now, Future}\}} \sum_{i \in \mathcal{L}_n} x^t_n = \hat{G} \quad [\hat{u}] 
\]
\[0 \leq x_{\text{Now}}^i \leq q^i - q^{i-1} \quad \forall i \in I_{\text{Now}}/\{1\} \quad (q^i, q^{i-1} \in \Xi_n)\]

\[0 \leq x_{\text{Future}}^i \leq q^i - q^{i-1} \quad \forall i \in I_{\text{Future}}/\{1\} \quad (q^i, q^{i-1} \in \Xi^{t+1})\]

Here the index “Now” refers to the current node \(n \in N_t\) of the scenario tree (which is in the current stage \(t\)), therefore \(I_{\text{Now}}\) denotes the indexes of the set of \(u\) values that correspond to the optimal consumption levels in the U-C curve corresponding to node \(n \in N_t\) \((u_n(q), \forall q \in \Xi_n)\). Similarly, the index “Future” refers to the expected future as is observed by the current node in the tree. Hence \(I_{\text{Future}}\) denotes the indexes for the set of \(u\) values that correspond to all optimal (corner point) future consumption-to-go values that are observed by node \(n\) \((u^{t+1}(q), \forall q \in \Xi^{t+1})\).

The optimal consumption values in [S-HEU] form a feasible solution to [S-CM-MIP]. The optimal solution to [S-HEU] solution has advantages over other feasible consumption values as discussed in Section 6.6.

### 7.4.3 Stochastic U-C Policy Implementation

In this subsection we lay out the algorithm that we use to obtain near optimal policies with utilization of U-C curves, when given a total consumption value \(\hat{G}\) over the time horizon \(T = \{1, 2, \ldots, T\}\).

**Stochastic U-C algorithm: SUCA**

1. **Construct the aggregated expected U-C curve from Sections 7.4 or 7.4.1**
   
   Calculate \(v_n(q), \forall q \in Q, \forall n \in N_t, \forall t \in T\).

2. **Start forward pass**
   
   Set \(t = 1\) and set consumption to go \(G = \hat{G}\).

3. **For the realized node of tree \(n \in N_t\), find the corresponding \(u\) value**
   
   If \(G \in Q_n\):
   
   - Directly use \(v_n(G)\) to determine optimal consumption value\(^4\) at stage \(t\)
     
     \[\hat{x}_t = g(v_n(G), n)\].

   Else:

\(^3\)For stage-wise dependent model we have \(u^{t+1}_n(q), \forall q \in Q^{t+1}\)

\(^4\)The optimal ILR quantity is computed simultaneously with the optimal consumption value for each node \(n\) and for each \(u\) value. These values can be stored to be used in this step of the algorithm.
– Use the heuristic model to determine the consumption value at stage $t$

Solve $[\text{S-HEU}]$ with $G = \hat{G}$. Given $x_{\text{Now}}^*$ is the optimal consumption value of $[\text{HEU}]$ for present stage $\rightarrow \hat{x}_t = x_{\text{Now}}^*$.

4. Update consumption to go value in the forward pass

Set $G = G - \hat{x}_t$.

If $t \neq T$:

– Go to next step

Set $t = t + 1$ and go to step 3.

Else:

– End of simulation

The algorithm above is the last step in simulating (near) optimal policies for our multistage problem in a forward pass. In the next section, in order to comprehensively examine our multistage stochastic price-making consumer model, we will focus on the practical aspects of modelling uncertainty. First, we will lay out a scenario clustering method which enables us to reduce the size of the scenario tree. Secondly, we discuss different types of uncertainty in our specific problem and how we can incorporate them in our model.

### 7.5 Modelling Uncertainty

In this section we address scenario sampling methods, scenario trees and the structure of uncertainty. The section is divided into two parts. Firstly, we present a tailor-made clustering method in order to address the uncertainty within our model’s endogenous pricing process. Secondly, we probe different types of uncertainty and their impact on the scenario tree layout.

#### 7.5.1 Scenario Clusters

In order to construct a price process that adequately represents the future, we apply the same concept used in our bi-parametric sensitivity analysis (introduced in Section 5.4.5). The idea is to only focus on the impacts of the OPF and the uncertainty in the market participants’ actions, on the strategic node. Each scenario impacts our model’s objective function through energy and reserve price at the strategic node; these prices are functions
of the consumers' actions as well as those of other market participants (generators and other consumers). Given such endogenous nature of the strategic node’s prices, the information that we need from each scenario is the strategic’s node energy and reserve price for all possible actions of the strategic consumer. We take a price sensitivity analysis approach to define representative scenarios. Without loss of generality, and in order to reduce the number of scenario clusters, we only consider energy prices to be of importance in our clustering method.

In this section, \( \Omega_t \) denotes the set of all outcomes (scenarios) restricted to stage \( t \). In effect this is similar to \( \mathcal{N}_t \), where each \( n \in \mathcal{N}_t \) corresponds to a scenario \( \omega \in \Omega_t \). We assume that by solving the OPF model for scenario \( \omega \) and stage \( t \), with the strategic consumption level \( l \), the realized energy price at the strategic node would be \( \pi_l^\omega \). We denote this assumption with function \( \pi_l^\omega(l) \), which demonstrates the energy price corresponding to each consumption level \( l \in \mathcal{L} \), in scenario \( \omega \), at stage \( t \). In order to further quantify our sensitivity analysis, we first define the price ranges and assign an index \( (r) \) to each price range \( (p_r) \). Table 7.1 presents an example of possible price ranges for our model.

\[
\begin{array}{cccccc}
  r & 1 & 2 & 3 & 4 & 5 \\
  p_r & 0-20 & 20-40 & 40-60 & 60-100 & \geq 100 \\
\end{array}
\]

Table 7.1: Price ranges

At the next step we assign the price range indices to each consumption level \( (l) \). Therefore, we denote the cluster of scenario \( \omega \) at stage \( t \) \( (c_l^\omega) \), as a vector with \( |\mathcal{L}| \) elements, where \( (c_l^\omega)_l = r|\pi_l^\omega(l) \in p_r \). Defining scenarios by vectors enables us to utilize the idea of representative scenarios. Therefore, given that each unique vector represents a cluster, scenarios that result in the same vector of price-ranges can be considered to be in the same cluster. In addition to defining the clusters, we must calculate the relative frequency of each cluster occurring for each type of time period. This can be done through counting the number of members of each cluster over the sample space. Using these clusters we can also address stage-wise dependency in the price process. We find the probability of transitioning from one cluster in time period \( t \) to different clusters in \( t + 1 \) and build the corresponding transition matrix. In subsection 7.6.1, we present the scenario clusters and the transition matrix for our case study.

### 7.5.2 Scenario Realizations

In this subsection we discuss different structures of uncertainty for our mathematical model. So far we used a wait-and-see approach in defining our model. Figure 7.5 shows
a stylized stage-wise independent\(^5\) wait-and-see decision making process where, at each stage, the realized scenario is known, even though all future stages’ scenario that would be realized are uncertain. Here each node is referenced to by its stage-scenario pair.

\[ x_t = y_n^{\omega} \quad \forall n \in N, \forall \omega \in \Omega_t \quad (7.2) \]

Depending on the nature of uncertainty, we may wish to lay out our model with a here-and-now decision process. In this setting, we know which scenario \( \omega \) is realized only after we make a decision. In order to capture such uncertainty, at each stage, \( t \), we need to solve a MIP equivalent to [S-MIP] (from Section 5.2) where the the optimal consumption maximizes the expected profit of the major consumer (with marginal utility value \( u \)) over a set of scenarios for that stage (\( \Omega_t \)). However, in this model, we need to include an additional set of constraints that enforces the optimal consumption values of all scenarios to be equal, as shown below:

\[ x_t = y_n^{\omega} \quad \forall n \in N, \forall \omega \in \Omega_t \quad (7.2) \]

Here \( y_n^{\omega} \) is the optimal consumption value for scenario \( \omega \in \Omega_t \). We denote [S-MIP] with the additional set of constraints 7.2 as \( [H-MIP]_{t,u} \).

We solve the stochastic \( [H-MIP]_{t,u} \), \( \forall t \in \mathcal{T} \), which maximizes expected profit over \( \omega \in \Omega_t \) where \( \Omega_t \) is the set of possible future scenarios for stage \( t \). For a stage-wise independent model, in order to construct the corresponding U-C curves, we solve \( [H-MIP]_{t,u} \) \( \forall u \in [0,u^{max}] \) the same way as the deterministic model. Note that, in the here-and-now model, all the uncertainty is dealt with through the \( [H-MIP]_{t,u} \) optimization problem, therefore, the scenario tree for a stylized stage-wise independent model will consist of a single path, as illustrated by Figure 7.6.

\(^5\)In this section, for the sake of simplicity of demonstration, we assume that in each stage of the stylized scenario trees we have independent and identically distributed (iid) scenarios.
Although wait-and-see or here-and-now methods are useful in modelling specific types of stochasticity, in practice, decision making under uncertainty is a combination of wait-and-see and here-and-now actions, given the difference in our ability of predicting different uncertain parameters. Therefore, we combine the here-and-now and wait-and-see approaches into a hybrid method, in order to present a more realistic stochastic process.

We first utilize the type of uncertainty that cannot be accurately predicted to make scenarios for [H-MIP]. In the case of our model, such uncertainties arise from real-time fluctuations in demand or wind. In addition, we incorporate the wait-and-see uncertainty that follows known patterns and can be anticipated, to build the set of states in each stage of the scenario tree for the stochastic model [S-MIP]. Such uncertainties are built into our model as the estimated generation offers, as well as average demand and prices, depending on the time-of-day, day-of-week and season of a trading period.

Figure 7.7 demonstrates a sample scenario tree for the hybrid model in a stage-wise independent setting. Here each node in the tree is denoted with its stage, wait-and-see scenario and here-and-now scenario respectively. For our stochastic optimization model, the hybrid scenario tree can also illustrate our implementation of scenario clusters. In Figure 7.7 each box in the scenario tree represents a unique type of time period, which we refer to as a cluster. Each cluster consists of scenarios that share similar patterns of generation offers and total demand. Given that market participants have a good estimate of the type of trading period they are in (in terms of generation offers and total demand), before making their decision, we define clusters in a wait-and-see setting, as shown (by rectangles) in Figure 7.7. On the other hand, the real-time fluctuations of the market need to be addressed through a here-and-now stochastic model, within each cluster. The individual scenarios are demonstrated by circles in Figure 7.7, where the scenarios in each rectangle belong to the same cluster.

This hybrid method is applicable for models with stage-wise independent or dependent uncertainty; therefore we can construct the expected U-C curves as described in SUC-C-I (in Section 7.4) and SUC-C-D (in subsection 7.4.1). In the next section we utilize our illustrated hybrid method in our case study of a large consumer of energy in New Zealand.
7.6 Case Study

In this section, we implement our stochastic model for the New Zealand aluminum smelter (NZAS), an industrial consumer of energy in the NZEM. The strategic consumer’s load is large enough to affect the energy price at its node. Also, it is capable of offering ILR, and can thereby influence the reserve price in the South Island. We solve our model for a given total consumption level to determine our consumption policy over a time-horizon based on our proposed method. The expected cost of this policy is compared to the operating policy of NZAS which consumes a fixed quantity over all trading periods.

In order to present a comprehensive report on the policy performance, we limit our example to analysis of one season, winter. Hence, we construct our sample space using historical data from winters 2016–2018. Our proposed algorithm for solving stochastic multistage demand-response allows us to solve our model over the time horizon of up to one month in a reasonable time-frame. In what follows, we lay out our sampling method and construct the scenario clusters (as illustrated in subsection 7.5.1).

7.6.1 Scenario Clusters for NZAS

In order to build scenario clusters, we sample multiple scenarios from the sample space (winters 2016–2018) for each stage $t$ over the time horizon $T$. We distinguish the type of a stage ($\gamma$) by the trading period of the day ($\text{TP} \in \mathcal{P}$), the type of day ($d \in \mathcal{D}$), as weekend or weekday, and the season ($s \in \mathcal{S}$). Hence there exist $|\mathcal{P}| \times |\mathcal{D}| \times |\mathcal{S}|$ types of stages; we call the set of all types of stages as $\Gamma$. Depending on the length of the time horizon, the number of types of stages that are included in the model is different. For instance, in our monthly scheduling model, we target one season $|\mathcal{S}| = 1$, therefore we have $(|\Gamma| = 48 \times 2 \times 1 = 96)$. Note that two stages can share the same $\gamma$ (e.g. 2 p.m. in all weekdays of winter). Utilizing the above definitions, we assign a sample space $\Theta_\gamma$, $\forall \gamma \in \Gamma$, where each sampled scenario $\theta \in \Theta_\gamma$ is derived from the historical data of trading periods of type $\gamma$.

Moreover, $\forall \gamma \in \Gamma$, we run the vSPD (the OPF problem for the NZEM) $\forall \theta \in \Theta_\gamma$, for different designated levels of consumption $l \in \mathcal{L}$ for our strategic consumer (the NZAS in NZEM) and report on the corresponding energy prices at the strategic node (Tiwai Point). Using this method, we can determine the kind of each scenario, in terms of realized prices corresponding to different strategic consumption levels. Table 7.2 shows an example of the change in price by increasing consumption of the strategic consumer, for a group of sample scenarios.
### Table 7.2: Price sensitivity analysis

<table>
<thead>
<tr>
<th>Date and Time</th>
<th>Parameters</th>
<th>Load</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>d</td>
<td>TP</td>
</tr>
<tr>
<td>5/07/2016 0:00</td>
<td>w</td>
<td>w-d</td>
</tr>
<tr>
<td>5/07/2016 0:30</td>
<td>w</td>
<td>w-d</td>
</tr>
<tr>
<td>5/07/2016 1:00</td>
<td>w</td>
<td>w-d</td>
</tr>
<tr>
<td>5/07/2016 1:30</td>
<td>w</td>
<td>w-d</td>
</tr>
<tr>
<td>5/07/2016 2:00</td>
<td>w</td>
<td>w-d</td>
</tr>
</tbody>
</table>

In order to further quantify the sensitivity analysis, we assign an indicative value to each price range. Table 7.1, presents the price ranges for our case study.

Utilizing Table 7.1’s price range definitions, we assign the price range numbers to each consumption level and make Table 7.3. Note that we picked 5 price ranges, since it was a good trade-off between accuracy and practicality. In Table 7.3, we view each row as a vector where each unique vector is a cluster. For example, the first two scenarios of this sample are in one cluster, with a corresponding cluster vector [2,3,3].

### Table 7.3: Price range clusters

<table>
<thead>
<tr>
<th>Date and Time</th>
<th>Parameters</th>
<th>Load</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>d</td>
<td>TP</td>
</tr>
<tr>
<td>5/07/2016 0:00</td>
<td>w</td>
<td>w-d</td>
</tr>
<tr>
<td>5/07/2016 0:30</td>
<td>w</td>
<td>w-d</td>
</tr>
<tr>
<td>5/07/2016 1:00</td>
<td>w</td>
<td>w-d</td>
</tr>
<tr>
<td>5/07/2016 1:30</td>
<td>w</td>
<td>w-d</td>
</tr>
<tr>
<td>5/07/2016 2:00</td>
<td>w</td>
<td>w-d</td>
</tr>
</tbody>
</table>

We derive the set of clusters \( C_\gamma \) through the above algorithm \( \forall \gamma \in \Gamma \); here each cluster \( c \in C_\gamma \) consists of a subset of sample scenarios \( \Theta_\gamma^c \). Furthermore, \( \bigcup_{c \in C_\gamma} \Theta_\gamma^c = \Theta_\gamma \) and all subsets \( \Theta_\gamma^c \) are pairwise disjoint. Using the above definitions, we integrate the notion of a hybrid scenario tree with our clusters (as described in subsection 7.5.2). In our model’s scenario tree, at each stage \( t \) with type \( \gamma \), we have \( |C_\gamma| \) nodes, and represent each node with the [H-MIP]_t model, that is solved over the scenario set \( \Theta_\gamma^c \) (in the next subsection we will use the solution of [H-MIP]_t to build the corresponding U-C curves). In addition to defining the clusters, we calculate the probability distribution of the clusters \( c \) at each type of stage \( \gamma \) (as illustrated in subsection 7.5.1) and denote it by \( \varphi_\gamma^c \).
7.6.2 Expected U-C Curves for NZAS

In order to construct the expected U-C curve over the time horizon $T$ for our case study, we first need to calculate the individual U-C curves$^6$ for each $t \in T$. Thereafter, for our hybrid model, each individual U-C curve corresponding to stage $t$ (where $t$ is of type $\gamma$), and cluster $c$ is calculated through iteratively solving [H-MIP] over set of scenarios $\Theta_{t, c}$, which is the stochastic optimization program corresponding to cluster $c$.

When solving our multi-stage model, we focus on which stage ($t \in T$) we are at, instead of which type of stage we are facing. Therefore, we use the function $\psi(t) = \gamma$ that returns the type of stage $t$ and update our notation as follows; we assign a scenario set $\Omega_t$ to each stage $t$, which consists of all clusters $c \in C_{\psi(t)}$. Therefore, each cluster $c \in C_{\psi(t)}$ corresponds to one scenario $\omega \in \Omega_t$; in other words, we can view each $\omega \in \Omega_t$ as the representative scenario of its corresponding cluster $c \in C_{\psi(t)}$. Moreover, the probability of occurrence of $\omega \in \Omega_t$ in a stage-wise independent setting ($\rho^\omega_{\omega}$) is equal to that of its corresponding cluster ($\phi^c_{\psi(t)}$). Similarly, when we assume stage-wise dependency for the transition probabilities, we have: $\rho^\sigma_\omega = \phi^{b,c}_{\psi(t)}$, where $\sigma$ is the realized scenario at stage $t - 1$. Note that for trading periods that are transitioning from a weekday to weekend and vice versa we set the sample set accordingly in order to maintain the information on transition probabilities. In this case study, for each trading period $t \in T$, our sample space consists of 50 trading periods of type $\psi(t)$. Note that the number of clusters for each $t$ is different.

For the stage-wise independent model, by utilizing the SUC-C-I algorithm that was described in Section 7.4, we construct the expected U-C curves over the designated time horizon. In addition, in order to construct expected U-C curves with stage-wise dependency, we utilize the SUC-C-D algorithm that was described in subsection 7.4.1. Here we use $\rho^\sigma_\omega$ to address the probability of transitioning from scenario $\sigma$ at stage $t - 1$ to $\omega$ at stage $t$. Note that $\rho^\sigma_\omega$ is equivalent to $\rho_n,l$, which is the probability of going from node $n \in \mathcal{N}_{t-1}$ in the tree to node $l \in \mathcal{N}_t$ in the tree, where $n$ and $l$ are the tree nodes that correspond to $\sigma$ and $\omega$, respectively.

The expected U-C curve provides us with a comprehensive look-up table. Thereafter, when given a total consumption $G$ over time horizon $T$, we can derive a near-optimal action from the expected U-C curve. In the next section, we simulate our proposed policy over random out-of-sample paths and report on the expected costs resulting from our optimal policies versus a fixed-consumption policy.

---

$^6$We calculate individual U-C curves as described in Section 6.5, but, instead of solving [MIP]$_{t,u}$, we solve [H-MIP]$_{t,u}$. We use a bisection search to determine the optimal consumption corresponding to marginal utility values in $(0, u^{max})$, where $\Delta u = 1$ is the smallest interval for $u$ values.
7.6.3 Policy Simulation Algorithm

In this section we use the algorithm that was laid out in subsection 7.4.3 (SUCA) to simulate our proposed policy. In order to address the fluctuations in demand and intermittent generation resources, we conduct an out-of-sample simulation in which we take up on a hybrid stochastic scheme where we assume that, before decision making, the type (cluster) of the realized scenario are known (wait-and-see), but the details of the particular scenario is realized after decision making (here-and-now). To simulate out-of-sample scenarios for our policy we first build a random sample path. For the stage-wise independent setting, at each stage $t$, we randomly choose a scenario $\hat{\omega}_t$ from $\Omega_t$, with the probability $\rho_{\hat{\omega}_t}$. Furthermore, for the stage-wise dependent model, $\forall t \neq 1$, given $\hat{\sigma}$ is the picked scenario at $t - 1$, we randomly choose scenario $\hat{\omega}_t$ from $\Omega_t$, with probability $\rho_{\hat{\omega}_t}^7$.

For $t = 1$ the probability of choosing $\hat{\omega}$ is calculated the same way as the stage-wise independent model.

---

**Figure 7.8:** Optimal stochastic policy out-of-sample simulation algorithm
Using this sample path (denoted by $\hat{\Omega}$), for each stage $t$, we have a randomly picked cluster ($c \in C_{\psi(t)}$) that corresponds to $\hat{\omega}_t$. Thereafter, given $\hat{\omega}_t$, we can use an out-of-sample scenario $\varpi_t$, that belongs to the cluster $c$ (corresponding to $\hat{\omega}_t$), as the realized scenario at stage $t$. Therefore in our forward pass, we employ an action that is extracted from the U-C curves, and apply it to the out-of-sample scenarios. Figure 7.8 lays out the steps of our simulation algorithm for NZAS. Here, the blue boxes demonstrate the exogenous setting of each simulation, in addition gray arrows are used to show the stages needed for the calculation of cost for out-of-sample scenarios.

Over the remainder of this section, we will report on the average cost of 200 different sample paths. We present numerical results for short-term (daily), mid-term (weekly) and long-term (monthly) demand scheduling.

### 7.6.4 Daily Scheduling

In this section we simulate our daily consumption policies using random sample paths and compare it with a fixed policy, in which the consumption rate stays the same throughout the time horizon. Note that in order to simulate the fixed policy, we run our model for the out-of-sample scenarios with the given fixed consumption rate and compare the total cost with our policy. In our case study, the total consumption requirement for one day is set to 13728 MWh\(^8\), hence for the fixed policy, the consumption rate is 572 MW for all time periods\(^9\).

In Table 7.4, we report on the average total expected cost over 200 paths, for the stage-wise dependent (first row) and for the independent U-C (second row) policy and the fixed policy.

<table>
<thead>
<tr>
<th>Expected cost ($)</th>
<th>Average Price ($/MWh)</th>
</tr>
</thead>
<tbody>
<tr>
<td>U-C Policy[D]</td>
<td>966,225</td>
</tr>
<tr>
<td>U-C Policy[I]</td>
<td>1,045,437</td>
</tr>
<tr>
<td>Fixed Policy</td>
<td>1,166,420</td>
</tr>
</tbody>
</table>

**Table 7.4: Daily scheduling comparison**

From Table 7.4 we observe that the stage-wise dependent optimal policy yields the lowest average cost over our simulations. Note that we have used the same set of (out-of-sample) sample paths for all three types of policies; in addition, our sample paths are generated using the transition probability that was described in subsection 7.6.2. The

---

\(^8\)Note that in the NZEM trading periods are half-hours. Therefore, if the consumer’s consumption rate is 100MW, during one trading period it consumes 50 MWh.

\(^9\)572 MW is the fixed rate that was used by the NZAS in the historical data that we have used.
stage-wise independent model still has an advantage over the simple fixed policy, where the average nodal price for the NZAS has decreased by $13.5.

![Graph showing consumption and price comparison](image)

**Figure 7.9: Policy comparison**

In order to further illustrate the policies’ performance, we choose a random sample path and report on each stage’s optimal actions. In this example, we compare our policy with a fixed policy, where the same level of consumption applies to each stage. In Figure 7.9 we present a comparison between optimal actions, followed by a plot showing the realized prices in each stage, given the consumption level. This shows that our policy is able to utilize the off peak time periods and consume more, such as TP9. Furthermore, our policy adjusts the consumption in peak time periods and reduces its level in order to alleviate spike prices, TP17 is an example of such time periods, where the fixed policy results in much higher price than our policy’s realized price. In this example, we used the actions corresponding to stage-wise dependent U-C policy and the fixed policy. In the next part, we demonstrate the difference between the stage-wise dependent and independent approaches in a similar setting.

**Stage-wise Dependency and Independency**

Electricity prices are affected by changes in temperature, rainfall, or in the event of generators’ or transmission’s shutdowns. Generally fluctuations of electricity prices that are caused by natural phenomena continue for a few hours. Therefore, the beginning of an unforeseen rise or fall of the price could be used as a signal for adjusting the
agent’s actions in the trading periods that follow it. We address this property in our stage-wise dependent model. In this section, we present an example, that illustrates the advantages of using a stage-wise dependent model. In this example we used a sample path that consists of scenarios with potentially higher than average prices. (Such sample path could be representative of a very cold winter day.) We simulate both our stage-wise dependent and independent U-C policies on one such sample path and report on the optimal consumption levels and realized prices for all trading periods over that day in Figure 7.10.

As shown in Figure 7.10, the stage-wise dependent policy anticipates the prospect of persisting high prices whereas, in the stage-wise independent model, the probability of lower priced scenarios in the future is high enough for the consumer to delay its consumption to later time periods in the day; this leads to very high prices towards the end of the time horizon for the stage-wise independent model.

7.6.5 Weekly Scheduling

In this section, we simulate our proposed policy on a week-long time horizon. We first report on the performance of our method, in comparison with the fixed consumption policy. Secondly, we discuss the length of time horizon, and how the strategic consumer can decrease its total cost if it schedules demand over a longer time period. Note that
in the weekly scheduling simulation, the time horizon consists of $48 \times 7 = 336$ stages. The algorithm used to employ our U-C curves as the marginal value's look-up table is the same as depicted in Figure 7.8.

Table 7.5 compares the U-C and fixed policy, when simulated for 200 random sample paths. For these simulations, the total consumption equals $336 \times 0.5 \times 572 = 96096\,\text{MWh}^{10}$.

<table>
<thead>
<tr>
<th></th>
<th>Expected cost ($)</th>
<th>Average Price ($/\text{MWh})</th>
</tr>
</thead>
<tbody>
<tr>
<td>U-C Policy</td>
<td>6,677,076</td>
<td>67.23</td>
</tr>
<tr>
<td>Fixed Policy</td>
<td>8,042,006</td>
<td>82.91</td>
</tr>
</tbody>
</table>

Table 7.5: Weekly scheduling comparison

The longer the time horizon for demand scheduling is, the longer it takes to construct the U-C curves. On the other hand, the extended time horizon allows us to take into account further future information in our present actions, and to have more opportunity to shift load to low price periods. In this subsection, we lay out an example in which we compare the performance of two policies; the first one is solving the stochastic demand scheduling problem through utilizing U-C curves, for random sample paths, over a week-long time horizon, with $G = 96096\,\text{MWh}$. In the second example, we use the same sample paths that we used in the first example, but here we solve a sequence of seven one-day long scheduling problems. The total consumption required for each of these policies is the same, $(96096/7 = 13728)$. Table 7.6 presents the comparison of the two experiments.

<table>
<thead>
<tr>
<th>Schedule</th>
<th>Mon</th>
<th>Tues</th>
<th>Wed</th>
<th>Thur</th>
<th>Fri</th>
<th>Sat</th>
<th>Sun</th>
</tr>
</thead>
<tbody>
<tr>
<td>Weekly</td>
<td>13714</td>
<td>14258</td>
<td>12731</td>
<td>14014</td>
<td>13584</td>
<td>13961</td>
<td>13831</td>
</tr>
<tr>
<td>Daily</td>
<td>13728</td>
<td>13728</td>
<td>13728</td>
<td>13728</td>
<td>13728</td>
<td>13728</td>
<td>13728</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Schedule</th>
<th>Mon</th>
<th>Tues</th>
<th>Wed</th>
<th>Thur</th>
<th>Fri</th>
<th>Sat</th>
<th>Sun</th>
</tr>
</thead>
<tbody>
<tr>
<td>Weekly</td>
<td>864,854</td>
<td>1,097,349</td>
<td>794,562</td>
<td>869,550</td>
<td>748,289</td>
<td>777,241</td>
<td>837,680</td>
</tr>
<tr>
<td>Daily</td>
<td>867,953</td>
<td>1,020,440</td>
<td>980,425</td>
<td>848,277</td>
<td>836,657</td>
<td>748,979</td>
<td>822,674</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Schedule</th>
<th>Mon</th>
<th>Tues</th>
<th>Wed</th>
<th>Thur</th>
<th>Fri</th>
<th>Sat</th>
<th>Sun</th>
</tr>
</thead>
<tbody>
<tr>
<td>Weekly</td>
<td>58.7</td>
<td>72.9</td>
<td>60.2</td>
<td>61.1</td>
<td>52.4</td>
<td>54.3</td>
<td>56.6</td>
</tr>
<tr>
<td>Daily</td>
<td>59.5</td>
<td>71.0</td>
<td>68.8</td>
<td>60.9</td>
<td>58.3</td>
<td>53.8</td>
<td>56.0</td>
</tr>
</tbody>
</table>

Table 7.6: Weekly scheduling comparison

Table 7.6 shows the benefits of having flexibility in total consumption over one day versus one week. Here we compare the total load in one day for the two policies, while the aggregated amount for the whole week stays fixed. The results demonstrate that the weekly scheduling is more able to adjust consumption to take advantage of lower priced days than the daily scheduling policy. The total cost in daily scheduling over the 7 days is $6,125,408 which is 2.2\%$ higher than that of the weekly scheduling, which is $5,989,527$. Moreover, the last two rows of this table show the time-weighted price at the strategic node, under the two types of scheduling. We observe that nodal price of

---

10Note that our stages are half-hours
higher priced days (like Wednesday for this simulation) goes lower when we are able to shift load from day to day in the weekly scheduling. We further illustrate the results in Figure 7.11, where we lay out the detailed consumption levels and corresponding prices for Wednesday and Thursday.

As shown in Figure 7.11, the weekly scheduling scheme reduces total consumption on Wednesday, but compensates for it later. On the other hand, the daily scheduling is set to consume the fixed total consumption and therefore on Wednesday the average price becomes about $8 more than that of the weekly scheduling model. On Thursday, weekly schedule’s total consumption (14014MWh) is more than the total amount of daily scheduling (13728MWh), but the difference in the average price of energy is only $0.2.

7.6.6 Monthly Scheduling

The length of the period of a production plan primarily depends on the duration of the contracts with its clients. The length of a firm’s manufacturing schedule, can also be affected by sustained and considerable increases or decreases in the price of its product. For instance, if a firm observes that the local or global price of its product is constantly dropping, it might want to set a lower production level for the next scheduling period.
In order to practically address the contract length, and the price signals, we have chosen the duration of one month as the longest time-horizon for our case study on NZAS’s demand scheduling.

In this section we present numerical results for monthly demand-scheduling with our U-C policy and compare it with a fixed consumption policy. Note that for our case study the number of trading periods in one month is \((48 \times 7 \times 4 = 1344)\). Solving a stochastic multi-stage program over a large time-horizon raises concerns on both the accuracy of generated policies and the practicality of implementation.

The first issue is due to the unavailability of reliable information for the time-periods that are far in the future. Therefore, the parameter estimates for the scenarios at the time periods which are close to the end of a long time-horizon can become inaccurate. However, as we approach these time periods in our forward pass, we could gain more useful information on the parameter values associated with these TPs. For instance, the weather forecast becomes more accurate as it gets closer to the date, resulting in more accurate estimates of reservoir levels and temperature.

Secondly, building the U-C curve becomes more cumbersome as the planning period becomes longer. When large number of U-C curves are aggregated, the width of the steps in the aggregated expected U-C curve become smaller, which requires larger numbers of calculations in each recursion and much larger memory space.

In order to address the accuracy and practicality issues, we propose a time adaptive aggregation method for building U-C curves. Given that the information on future time periods evolves as we move forward in time, we build the individual U-C curves for the farther time periods with less detail. Hence, when constructing U-C curves for scenarios in distant time periods, the intervals of \(u\) values in \([0, u_{\text{max}}]\) are larger than those of the time periods which are closer to the start of our time-horizon.

In order to implement the time-adaptive U-C curve approach for the monthly scheduling, after constructing the first aggregated expected U-C curve (at day 0), we run our forward pass and, at the end of each week, we update the aggregate U-C curve that is corresponding to the remaining weeks. Table 7.7 presents the interval updates at the end of each week in the forward pass.

<table>
<thead>
<tr>
<th>Day</th>
<th>Day 0</th>
<th>Day 7</th>
<th>Day 14</th>
<th>Day 21</th>
</tr>
</thead>
<tbody>
<tr>
<td>week 1</td>
<td>1</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>week 2</td>
<td>1</td>
<td>1</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>week 3</td>
<td>10</td>
<td>1</td>
<td>1</td>
<td>-</td>
</tr>
<tr>
<td>week 4</td>
<td>10</td>
<td>10</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 7.7: Bisection search \(u\) intervals for monthly scheduling
In the monthly scheduling case study, we used 100 random sample paths, using the historical data for winter 2016 and 2017. In order to probe the impacts of updating U-C curves, we conduct the following experiments on these 100 sample paths and report the average results. The first experiment is our standard U-C policy and assumes sufficient and consistent availability of information for all the four weeks in the horizon. In the second experiment we assume that, for each sample path, less information is available for weeks three and four at the beginning of simulation. Therefore the U-C curve is based on less detailed future data. Here, we implement the time-adaptive U-C policy, and use Table 7.7 to construct and update our aggregated U-C curves. Note that the same sample paths are used in these two experiments. Table 7.8 shows that the weekly adaptive policy significantly outperforms the fixed consumption policy in terms of expected costs. In addition, in Table 7.8 we compare the updating mechanism with the standard U-C policy. Here, we observe that the expected cost in the updated U-C policy is only 0.5% higher than the standard U-C policy, which is an acceptable trade-off for reducing the problem size whilst also enabling the updating of future scenario distributions.

<table>
<thead>
<tr>
<th></th>
<th>Expected cost ($)</th>
<th>Average Price ($/MWh)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standard U-C Policy</td>
<td>21,160,509</td>
<td>62.6</td>
</tr>
<tr>
<td>Weekly Updated U-C Policy</td>
<td>21,276,065</td>
<td>62.7</td>
</tr>
<tr>
<td>Fixed Policy</td>
<td>25,499,180</td>
<td>82.4</td>
</tr>
</tbody>
</table>

Table 7.8: Monthly scheduling comparison

7.7 Conclusions

In this chapter we presented stochastic multistage solution methods for price-making major consumers of electricity. Our method solves the co-optimized demand scheduling problem that can capture both stage-wise dependent and independent uncertainty. With the introduction of Expected U-C curves, we decomposed our large stochastic MIP to time periods and also scenarios. In addition, we reduced the size of our model’s scenario tree by utilizing a tailor-made scenario clustering method.

We constructed expected aggregated U-C curves via utilizing the Lagrangian multiplier on the total consumption constraint as the optimality control parameter that also captures non-anticipativity constraints. In order to obtain the expected aggregated U-C curve (our look-up table), we computed vertical expectation of marginal cost of scenarios at each step in addition to horizontal aggregation of U-C curves with backward transitions. Furthermore, we presented a stochastic heuristic model that provides near-optimal solutions for our non-convex problem.
Moreover, we implemented our optimization method for NZAS, a large consumer of energy in New Zealand. Our policy simulation over out-of-sample scenarios showed the advantage of our proposed method, which resulted in an average 18% cost reduction compared to the fixed consumption policy. We also showed that, by extending the time-horizon, our method takes advantage of the higher flexibility in consumption level which leads to lower total cost. For instance, weekly scheduling saves about 2%, when compared to daily scheduling for the same total consumption over the seven days. By utilizing our proposed method, we also solved the multi-stage demand planning over one month for NZAS. Our algorithm allows for incorporating stage-wise dependent scheduling, as well as updating and tuning the look-up table as the information about the future evolves in the forward pass.

In this chapter, we provided major manufacturers with a mid-term cost-minimizing model, however, the results of our simulations can also be used for long-term policy making purposes. By sensitivity analysis on the expected costs associated with U-C policy simulations, a strategic manufacturer can determine the expected optimal total consumption level for its future production contracts.

Finally, we showed that the strategic behavior of a major consumer who purchases electricity from the wholesale market and has flexible load, not only decreases its own costs but also lowers the spot price of electricity, which can also benefit smaller consumers.
Chapter 8

Concluding Remarks

In this thesis we studied a strategic major consumer that simultaneously optimizes energy bids and ILR offers in a deregulated and co-optimized energy and reserve market. Depending in the size of the consumer and the structure of the market, a major consumer may optimally bid energy and offer reserve as a price-taking or price-making consumer. In this thesis, we have discussed and modeled the optimal behavior of both a price-taking major consumer and a price-making consumer, with an emphasis on the latter.

A price taking consumer bids/offer its true utility/cost functions to a welfare maximizing model, and, if all agents are price takers, a competitive equilibrium could be achieved through economic dispatch. In this model however, the joint utility and cost functions of energy and reserve may not be incorporated due to the step-wise shape of stacks in the market. In order to explore this issue, we consider an example in which we calculate the opportunity cost arising from the co-optimized bid and offers to find the loss in efficiency.

In order to model a large manufacturer’s participation in the market more accurately, we developed a strategic consumption model in which the strategic agent anticipates the impacts of its actions on the market prices and, simultaneously, co-optimizes its consumption level and ILR offer in order to maximize its profit. Utilizing this model, we began our analysis by studying the impacts of strategic co-optimization through stylized examples. Firstly, we showed that demand response may lead to lower energy prices; in particular through alleviating price spikes. In addition, in our example, we demonstrated that reserve prices are also affected by the strategic behavior of the major consumer. While this strategic behavior is designed to gain high ILR revenue for the consumer, we also observed that, due to the co-optimization of energy consumption and reserve procurement and the linking constraint between these two variables, the consumer may
choose to consume more energy than it would in absence of a capability to offer ILR. It
does this so that it can offer more ILR and, as a result, gain higher revenue from ILR.

In addition to a pure demand response model, we studied co-optimized models of strate-
gic consumption that are combined with CfDs. In one example, we compared the optimal
policies for a strategic consumer with and without CfDs; this showed that introducing
CfDs into the profit maximizing problem significantly changes the behavior of the major
consumer. We observed that a CfD may encourage the major consumer to increase spot
prices through its consumption level instead of decreasing its consumption; this effect
is due to the structure of the CfD payments, where the consumer is paid the difference
between the spot price and the contract price for a fixed contract volume. Hence it may
be beneficial for the strategic agent to push the prices higher, since it does not face the
consequences of purchasing electricity at high prices (as long as it still consumes less
than the contract quantity).

The volatility of demand and generation offers within electricity markets necessitates
the development of tools for decision making under uncertainty. In order to address
this issue, we set out a stochastic co-optimization model that computed the optimal
(in expectation) energy bid and reserve offer stacks for each trading period. In order
to implement our model for a large-scale problem, we used reformulation methods to
reduce the solution time. Of the several reformulations that we tried, the bi-parametric
sensitivity analysis method had the greatest impact on reducing the solution time. This
enabled us to solve our large-scale NZEM model to a zero optimality gap for more than
20 scenarios in one hour, compared with the standard MIP reformulation that could
only solve up to six scenarios (on average) in one hour.

In addition to the application of the bi-parametric reformulation method for our problem,
this algorithm can be extended to any bi-level optimization problem in which some
of the right-hand sides of the lower-level problem are considered as decision variables
in the upper-level problem. The resulting bi-linear terms in the objective are then
optimized over the set of equilibrium constraints stemming from the lower-level problem.
In addition, the bi-parametric sensitivity analysis can be extended to higher dimensions;
however developing the mathematical formulation for the general use of this method is
not necessary for our application.

By implementing our stochastic co-optimization model for NZAS and utilizing historical
data, we simulated our proposed optimal policy over out-of-sample historical scenarios.
The simulation process consisted of obtaining the optimal energy bid and reserve of-
er stacks and submitting them to the OPF problem for out-of-sample scenarios. We
reported on realized dispatch quantities and corresponding prices. By comparing the
results of our optimal policy with a fixed consumption policy, we demonstrated the advantages of our method.

In the final part of this thesis, we presented a multistage cost-minimizing problem, that addressed large manufacturers’ contracts with their clients in terms of total production levels throughout a given time horizon. Firstly, by utilizing the structure of our multi-stage problem, we developed Lagrangian methods to decompose our model into each stage’s co-optimization problem. Secondly, by implementing heuristic methods we further tackled the non-convexities arising from the MIP model and enhanced the performance of our policies. In addition, in order to incorporate stochasticity to our multi-stage model while maintaining the computational tractability of our model, we presented a problem-specific scenario clustering method.

Lastly, we implemented our cost-minimizing model for NZAS, utilizing historical data. We implemented our model by producing policies for daily, weekly and monthly scheduling and reported on average expected cost over 200 random sample paths. Our proposed optimization method enables us to update our forecast of the future at any stage of our simulation and to adjust actions based on new information. Such information can include updated estimates on lake levels, temperature and precipitation, all of which may be incorporated in our model through changing the probability of occurrence of certain types of scenarios, or by introducing new scenarios to the future sample space.

Our stochastic strategic optimization methods not only offer operational decision making tools for major manufacturers, but also help them in a macro policy-making stage. For instance, a manufacturer can combine our strategic utility-maximizing and cost-minimizing models, to estimate appropriate production contract levels for different times of the year. In addition, a major consumer of electricity can use our presented optimization tools in order to estimate the endogenous future electricity prices and the resulting costs. This can be used as a factor in determining strategies such as their proper manufacturing capacity, future investments, or potential closures.
Part IV

Appendices
Appendix A

Gurobipy: Bi-parametric Sensitivity Analysis Region Construction

```python
from __future__ import print_function
from pulp import *
from gurobipy import *
import offer_stack as offer_stack
import random as random
import pickle as pickle
import csv as csv
import gom_plot as gom_plot
from pprint import pprint
from numpy import linspace
import random
import matplotlib.pyplot as plt
import datetime
import numpy
from gurobipy import *
import numpy as np
from scipy import linalg
from matplotlib.path import Path
from matplotlib import collections, colors, transforms
import matplotlib.patches as patches
import time

ZERO = 1e-5

day=11

def outputPoints(points):
    for i in range(len(points)):
        print(points[i][0], end='	')
        print(points[i][1])

def printSolution():
    if m.status == GRB.Status.OPTIMAL:
        print('%sCost: %g' % (m.objVal))
        print('%sBuy:')
        buy = m.getAttr('x', buy)
        for f in foods:
            if buy[f] > 0.0001:
                print('%s %g' % (f, buys[f]))
    else:
        print('No solution')

def removeRedundant(points):
    m3 = Model("redundant")
```
alpha={}
output=

x=m3.addVar(lb=-GRB.INFINITY)
y=m3.addVar(lb=-GRB.INFINITY)
for i in range(len(points)):
    alpha[i] = m3.addVar(lb=0.0,ub=1.0)

m3.update()
m3.addConstr(quicksum(alpha[i] for i in range(len(points))) == 1)
m3.addConstr(x==quicksum(alpha[i]*points[i][0][0] for i in range(len(points))))
m3.addConstr(y==quicksum(alpha[i]*points[i][0][1] for i in range(len(points))))
m3.update()
c2=-1000000
while 1:
    print(c2)
    m3.setObjective(x+c2*y, GRB.MAXIMIZE)
m3.update()
m3.optimize()
    for i in range(len(points)):
        print(alpha[i].x)
        if(alpha[i].x>=0.999):
            if points[i] not in output:
                output.append(points[i])
            break
        if(m3.SAObjUp[1]<10000000):
            c2=m3.SAObjUp[1]+0.001
        else:
            break
    c2=1000000
    while 1:
        print(c2)
        m3.setObjective(-x+c2*y, GRB.MAXIMIZE)
m3.update()
m3.optimize()
        for i in range(len(points)):
            print(alpha[i].x)
            if(alpha[i].x>=0.999):
                if points[i] not in output:
                    output.append(points[i])
                break
            if(m3.SAObjLow[1]>-10000000):
                c2=m3.SAObjLow[1]-0.001
            else:
                break
    return output

def reduceRegions(stored,p):
    linked=[]
    output=[]
    for i in range(len(stored)):
        if i not in linked:
            linked.append(i)
            temp2=stored[i]
            for j in range(i,len(stored)):
                if(stored[i][0][1][0]>=stored[j][0][1][0]-0.001 and stored[i][0][1][0]<=stored[j][0][1][0]+0.001)
                    and (stored[i][0][1][1]>=stored[j][0][1][1]-0.001 and stored[i][0][1][1]<=stored[j][0][1][1]+0.001)):
                        temp2=stored[i]+stored[j]
            linked.append(j)
            temp2=removeRedundant(temp2)
            output.append(temp2)
    return output

def findRegions(b1,b2):
    queue=[[b1,b2],[0,0],[0,0],[0,0]]
    stored=[]
    bases=[]
    Pr=[]
    j=0
    while len(queue)>0 and j<1000:
        j+=1
        b=queue.pop(0)
        [points,search,prices]=findRegion(b)
        if(len(points)<>0):
            temp=[]

        return output
for i in range(len(points)):
    temp.append([points[i], prices[i]])
stored.append(temp)

for i in range(len(search)):
    search[i].append([prices[0], prices[1]])

print(search)
queue = search + queue
print("Queue length:", end=' ')
print(len(queue))
return stored

def findRegion(b):
    [Binv, basisLB, basisUB, bOffset, prices] = solveLP(b)
    if len(basisLB) == 0:
        return [[], [], 0]
    [points, search] = findPoints(b[0][0], b[0][1], Binv, basisLB, basisUB, bOffset)
    outputPoints(points)
    return [points, search, prices]

def get_expr_coos(expr, var_indices):
    for i in range(expr.size()):
        dvar = expr.getVar(i)
        yield expr.getCoeff(i), var_indices[dvar]

def get_matrix_coo(m):
    matrix = []
    dvars = m.getVars()
    constrs = m.getConstrs()
    for i in range(len(dvars)):
        matrix.append([])
        for j in range(len(constrs)):
            if i == j:
                matrix[i+len(dvars)].append(1)
            else:
                matrix[i+len(dvars)].append(0)
    return matrix

def findExtremes(Binv, basisLB, basisUB, bOffset):
    m2 = Model("region")
    extremes = []
    extremes2 = []
    b1 = m2.addVar(lb=-GRB.INFINITY, name="consumption")
    b2 = m2.addVar(lb=-GRB.INFINITY, name="ilr")
    for i in range(len(Binv)):
        if (Binv[i, 0] != 0.0 or Binv[i, 1] != 0.0):
            m2.addConstr(Binv[i, 0] * b1 + Binv[i, 1] * b2 <= basisUB[i] - bOffset[i])
            m2.addConstr(Binv[i, 0] * b1 + Binv[i, 1] * b2 >= basisLB[i] - bOffset[i])
    m2.update()
    c2 = -1000000
    while 1:
        m2.setObjective(b1 + c2 * b2, GRB.MAXIMIZE)
        m2.update()
        m2.optimize()
        if (m2.SAObjUp[1] < 10000000):
            c2 = m2.SAObjUp[1] + 0.001
        else:
            break
    c2 = -1000000
    while 1:
        m2.setObjective(-b1 + c2 * b2, GRB.MAXIMIZE)
        m2.update()
        m2.optimize()
def findPoints(b1, b2, bInv, basisLB, basisUB, bOffset):
    points = []
    extremes, extremes2 = findExtremes(Binv, basisLB, basisUB, bOffset)
    for i in range(len(extremes)):
        points.append([extremes[i][0], extremes[i][1]])
    for i in range(len(extremes2)):
        points.append([extremes2[i][0], extremes2[i][1]])
    for i in range(len(points)):
        points[i][0] += b1
        points[i][1] += b2
    search = []
    for i in range(len(points)):
        x = (points[i][0] + points[(i+1)%len(points)])[0] / 2
        y = (points[i][1] + points[(i+1)%len(points)])[1] / 2
        delx = points[(i+1)%len(points)][1] - points[i][1]
        dely = points[i][0] - points[(i+1)%len(points)][0]
        length = pow(delx * delx + dely * dely, 0.5) / 0.01
        delx /= length
        dely /= length
        x += delx
        y += dely
        if x <= 800 and y <= x and x >= 0 and y >= 0:
            search.append([[x, y], [points[i][0], points[i][1]], [points[(i+1)%len(points)][0], points[(i+1)%len(points)][1]]])
    return [points, search]

def createLP(sce):
    n_d = [sce]
    trans_cap = arc_cap
    d = demands
    r = reserves
    d_x = {}
    r_x = {}
    f = {}
    d_p = {}
    d_y = {}
    r_p = {}
    r_y = {}
    d_tranches = {}
    d_prices = {}
    d_quantities = {}
    r_tranches = {}
    r_prices = {}
    r_quantities = {}
    lambd = {}
    pi = {}
    rpi = {}
    theta = {}
    dem_cons = {}
    res_cons = {}
    for i in n_d:
        (d_tranches[i], d_prices[i], d_quantities[i]) = offer_stack.SplitDataByNode(  
            nodes[i], d_tranche[i], d_price[i], d_quantity[i], d_M[i]  
        )
        d_tranches[i] = dict(zip(  
            nodes[i], d_tranches[i]  
        ))
        d_prices[i] = dict(zip(  
            nodes[i], d_prices[i]  
        ))
        d_quantities[i] = dict(zip(  
            nodes[i], d_quantities[i]  
        ))
        for n in nodes[i]:
            d_sort_list = zip(  
                d_prices[i][n], d_tranches[i][n], d_quantities[i][n]  
            )
            list.sort(d_sort_list)
            (d_prices[i][n], d_tranches[i][n], d_quantities[i][n]) = zip(d_sort_list)
        d_x[i] = {}
        f[i] = {}
        lambd[i] = {}
        theta[i] = {}
        d_y[i] = {}
r_y[i]{}, pi[i]{}, rpi[i]{}

dem_cons[i] = {}
res_cons[i] = {}

(r_tranches[i], r_prices[i], r_quantities[i]) = offer_stack.SplitDataByNode( nodes[i], r_tranche[i], r_price[i], r_quantity[i], r_M[i] )

r_tranches[i] = dict(zip( nodes[i], r_tranches[i] ))
r_prices[i] = dict(zip( nodes[i], r_prices[i] ))
r_quantities[i] = dict(zip( nodes[i], r_quantities[i] ))

for n in nodes[i]:
  r_sort_list = zip(r_prices[i][n], r_tranches[i][n], r_quantities[i][n] )
  list.sort(r_sort_list)
  (r_prices[i][n], r_tranches[i][n], r_quantities[i][n] ) = zip(*r_sort_list)

r_x[i]{}, r_y[i]{}

for n in gen_nodes[i]:
  d_x[i][n]{}
  d_x[i][n][t] = m.addVar(lb = 0.0, ub = d_quantities[i][n][t],
         name = "d_x_%s_%s_%s" % (t,n,i))

for n in nodes[i]:
  r_x[i][n]{}
  r_x[i][n][t] = m.addVar(lb = 0.0, ub = r_quantities[i][n][t],
         name = "r_x_%s_%s_%s" % (t,n,i))

for n in strategic_nodes:
  d_y[i][n] = m.addVar(lb = 0.0, ub = 800
         , name = "Strategic_electricity_Offer_%s_%s" % (n,i))
  r_y[i][n] = m.addVar(lb = 0.0, ub = 800
         , name = "Strategic_reserve_Offer_%s_%s" % (n,i))

for a in arc_id[i]:
  f[i][a] = m.addVar(lb = -trans_cap[i][a], ub = trans_cap[i][a]
         , name = "f_%s_%s" % (a,i))

for b in buses[i]:
  theta[i][b] = m.addVar(lb = 0, ub = 10000
         , name = "PowerAngle_%s_%s" % (i,b))

m.update()

for n in strategic_nodes:
  m.addConstr(d_y[i][n]==0.0,name="fixed_consumption")
  m.addConstr(r_y[i][n]==0.0,name="fixed_ILR")
  m.addConstr(r_y[i][n]<=d_y[i][n],name="energy-reserve")
  m.update()

Generator_Bathtub_Constraint_1(m, d_x[i] , r_x[i], bt1_rhs[i]
    , gen_nodes[i], d_tranches[i], r_tranches[i],i)

dem_cons[i] = d_Meet_Demand(d_tranches[i],m, d_y[i], d_x[i], d[i]
    , beta[i], f[i], nodes[i],gen_nodes[i]
    , strategic_nodes, buses[i], arc_id[i], arc_from[i], arc_to[i], i)

res_cons[i] = r_Meet_Demand(r_tranches[i], m, r_y[i], r_x[i],r[i]
    , beta[i], nodes[i], strategic_nodes, buses[i], i, islands , I[i])

# Kirchoff's loop flow laws
print ("Implementing loop flows for demand scenario %s...")
Kirchoff_Loop_Law(m, f[i], X[i], theta[i], arc_id[i], arc_from[i], arc_to[i], i)

j=0.0

for i in n_d:
  d_p[i]={}
  r_p[i]={}
  for n in gen_nodes[i]:
    for t in range(len(d_tranches[i][n])):
      d_p[i,n,t]=d_prices[i][n][t]+j/10000000
      j=j+1.0
  for n in nodes[i]:
    for t in range(len(r_tranches[i][n])):
      r_p[i,n,t]=r_prices[i][n][t]+j/10000000
      j=j+1.0

objective = (quicksum(d_p[i,n,t]*d_x[i][n][t] for i in n_d
    for n in gen_nodes[i] for t in range(len(d_tranches[i][n])))
  + quicksum(r_p[i,n,t]*r_x[i][n][t] for i in n_d
    for n in nodes[i] for t in range(len(r_tranches[i][n]))) )

m.setObjective(objective)

m.update()

def findVertical(regions):
  output[]
  for i in range(len(regions)):
def findAdjacent(regions):
    output = []
    for i in range(len(regions)):
        x = regions[i][0][0] + regions[(i + 1) % len(regions)][0][0]
        y = regions[i][0][1] + regions[(i + 1) % len(regions)][0][1]
        delx = regions[(i + 1) % len(regions)][0][0] - regions[i][0][0]
        dely = regions[i][0][1] - regions[(i + 1) % len(regions)][0][1]
        length = pow(delx * delx + dely * dely, 0.5) / 0.01
        delx /= length
        dely /= length
        x += delx
        y += dely
        output.append([x, y], [regions[i][0][0], regions[i][0][1]], [regions[(i + 1) % len(regions)][0][0], regions[(i + 1) % len(regions)][0][1]])
    return output

def solveAdjacent(b):
    constrs[0].RHS = b[0][0]
    constrs[1].RHS = b[0][1]
    m.update()
    # Solve
    m.optimize()
    if m.status == 3:
        return [], [], 0, 0, 0
    prices = [-constrs[0].pi, constrs[1].pi]
    basis = m.VBasis + m.CBasis
    if basis in bases:
        print('duplicate')
        return [], [], 0, 0, 0
    bases.insert(0, basis)
    entering = []
    leaving = []
    if oldVBasis != []:
        for i in range(len(entering)):
            if oldVBasis[i] == m.VBasis[i]:
                if m.VBasis[i] == 0:
                    entering.append(i)
elif oldVBasis[i]==0:
    leaving.append(i)
for i in rangeC:
    if oldCBasis[i]==m.CBasis[i]:
        if m.CBasis[i]==0:
            entering.append(i+numV)
    else:
        leaving.append(i+numV)
if oldVBasis==[] and len(entering)==<5 and len(leaving)==len(entering):
for i in range(len(entering)):
    print("PIVOTING")
    leave=pos_b[leaving[i]]
    Bmatrix[leave]=a[entering[i]]
    if entering[i+numV]!=m.CBasis[i+numV]:
        basisLB[leave]=m.LB[entering[i+numV]]
        basisUB[leave]=m.UB[entering[i+numV]]
    else:
        basisLB[leave]=0.0
        if constrs[entering[i+numV]].Sense=='=':
            basisUB[leave]=0.0
        else:
            basisUB[leave]=1000000000.0
    pos_b[entering[i]]+=leave
    pos_b[leaving[i]]=-1
for i in rangeV:
    if pos_b[i]!=-1:
        xB[pos_b[i]]=dvars[i].x
for i in rangeC:
    if pos_b[i+numV]!=-1:
        xB[pos_b[i+numV]]=constrs[i].Slack
    else:
        print("REFRESHING")
        Bmatrix=[]
        countb=0
        pos_b=[]
        basisLB=[]
        basisUB=[]
        xB=[]
        print('A')
        for i in rangeV:
            if m.VBasis[i]==0:
                xB.append(dvars[i].x)
                Bmatrix.append(a[i])
                basisLB.append(m.LB[i])
                basisUB.append(m.UB[i])
                pos_b.append(countb)
                countb+=1
            elif m.VBasis[i]==-1:
                pos_b.append(-1)
            else:
                pos_b.append(-1)
        print('B')
        for i in rangeC:
            if m.CBasis[i]==0:
                xB.append(constrs[i].Slack)
                pos_b.append(countb)
                countb+=1
                Bmatrix.append(a[i+numV])
                basisLB.append(0.0)
                basisUB.append(100000000.0)
            elif constrs[i].Sense=='<':
                basisLB.append(-100000000.0)
                basisUB.append(0.0)
            else:
                basisLB.append(0.0)
                basisUB.append(0.0)
            else:
                pos_b.append(-1)
        Binv=linalg.solve(np.transpose(np.matrix(Bmatrix)),I2)
    oldVBasis=m.VBasis
    oldCBasis=m.CBasis
    return [Binv,basisLB,basisUB,xB,prices]
Generation at strategic nodes less than the strategic generation capacity.

```python
def Consumer_utility():
    u = 220
    return u

def Generator_Costs(m, d_y, gen_costs, strategic_nodes, d_gen_cap, gen_c, gen_q, scen_no):
    for i in strategic_nodes:
        m.addConstr(d_y[i] == quicksum(gc_alpha[i,j] * round(gen_q[i][j]) for j in gc_set), 
                    'Match_Cost_Quantity_%s' % i)
        m.addConstr(gen_costs[i] == quicksum(gc_alpha[i,j] * round(gen_c[i][j]) for j in gc_set), 
                    'Generation_Costs_%s' % i)
        m.addConstr(quicksum(gc_alpha[i,j] for j in gc_set) == 1, 
                    'Sum_alpha(gc)_%s' % i)
        m.addSOS(GRB.SOS_TYPE2, [gc_alpha[i,j] for j in gc_set])

def d_Meet_Demand(d_tranches, m, d_y, d_x, d, beta, f, nodes, gen_nodes, 
                    strategic_nodes, buses, arc_id, arc_from, arc_to, scen_no):
    for b in buses:
        m.addConstr(quicksum( -beta[n,b]*d_y[n] for n in strategic_nodes) 
                    == quicksum( beta[n,b]*d[n] for n in nodes) 
                    - quicksum( beta[n,b]*d_q[n] for n in nodes) 
                    + quicksum( f[a] for a in arc_id if arc_from[a] == b) 
                    - quicksum( f[a] for a in arc_id if arc_to[a] == b) 
                    + 0.5*quicksum( line_loss[a] for a in arc_id if arc_from[a] == b) 
                    + 0.5*quicksum( line_loss[a] for a in arc_id if arc_to[a] == b) 
                    , 'd_Meet_Demand_%s_%s' % (b, scen_no))
```

Meet nodal demands, as well as conservation of flow at each bus. Includes losses.

```python
def r_Meet_Demand(r_tranches, m, r_y, r_x, r, beta, nodes, strategic_nodes, 
                    buses, scen_no, islanes, I):
    for l in islanes:
        m.addConstr(quicksum( -r_y[n] for n in strategic_nodes if Find_Island(buses, beta, n, I) == l) 
                    == r[l] + quicksum( r_x[n][t] for n in nodes if Find_Island(buses, beta, n, I) == l 
                    for t in range(len(r_tranches[n]))) 
                    , 'r_Meet_Demand_%s_%s' % (l, scen_no))
```

Meet nodal demands, as well as conservation of flow at each bus. Includes losses.
Applies a piecewise linear approximation for losses. The losses are derived from a series of convex cuts but are implemented as a piecewise linear function. The condition of convexity is important for the dual variable phi to remain correct in the formulation. This is why Line_Loss_Calculations needs to be called twice - once to set up the piecewise points, and then again to ensure the values refer to the convex implementation for the a, b, and phi’s.

\[(ll_pieces, ll_points, f_values, ll_values, ll_a, ll_b) = Line_Loss_Calculations_Piecewise(arc_id, R, fixed_loss, trans_cap, arc_has_losses, num_loss_tranches)\]

1. for a in arc_id:
   1.   for k in ll_points[a]:
   2.       f_alpha[a,k] = m.addVar(0.0,1.0,name = "alpha(f)")
   3.       m.update()
   4.       m.addConstr( line_loss[a] == quicksum(ll_values[a,k] * f_alpha[a,k] for k in ll_points[a] ), "Piecewise_Loss_%s" % a)
   5.       m.addConstr( f[a] == quicksum(f_values[a,k] * f_alpha[a,k] for k in ll_points[a] ), "Piecewise_Loss_%s" % a)
   6.       m.addConstr( quicksum(f_alpha[a,k] for k in ll_points[a]) == 1, "Sum_alpha(f)_%s" % a)
   7.       m.addSOS(GRB.SOS_TYPE2, [f_alpha[a,k] for k in ll_points[a]])
   8.       # Complimentary Slackness
   9.       for k in ll_pieces[a]:
   10.      m.addConstr(phi[a,k] <= 5000*(f_alpha[a,2*k] + f_alpha[a,2*k+1]), "Phi_Orthogonality_%s_%s" % (a,k))
   11.     m.addSOS(GRB.SOS_TYPE2, [phi[a,k] for k in ll_pieces[a]])

\[(ll_pieces, ll_points, f_values, ll_values, ll_a, ll_b) = Line_Loss_Calculations_Convex(arc_id, R, fixed_loss, trans_cap, arc_has_losses, num_loss_tranches)\]

\[\text{def } Line_Loss_Calculations_Piecewise(arc_id, R, fixed_loss, trans_cap, arc_has_losses, num_loss_tranches):\]

\[\text{Sets up the flow points and loss points for piecewise line losses.}\]

\[\text{loss_gradients = []}
\text{loss_intercepts = []}
\text{loss_values = []}
\text{flow_values = []}
\text{points = []}
\text{pieces = []}\]

\[\text{for } a \text{ in arc_id:}\]

\[\text{# Set up flow points - doubled up for each interior point}\]

\[\text{if arc_loss[a]:}\]

\[\text{flow_points = [-trans_cap[a], 0.0, 0.0, trans_cap[a]]}\]

\[\text{else:}\]

\[\text{flow_points = [-trans_cap[a], 0.0, 0.0, trans_cap[a]]}\]

\[\text{# In the piecewise implementation, n flow points implied n/2 flow pieces.}\]

\[\text{points[a] = range(len(flow_points))}\]

\[\text{pieces[a] = range(len(flow_points)/2)}\]

\[\text{# Set up loss points corresponding to the flow points}\]

\[\text{if arc_loss[a]:}\]

\[\text{ll_val, ll_a, ll_b) = Line_Loss_Function(flow_points, R[a], fixed_loss[a])}\]

\[\text{else:}\]

\[\text{ll_val = [0.0]*len(flow_points)}\]

\[\text{ll_a = [0.0]*len(flow_points))}\]

\[\text{for } k \text{ in points[a]:}\]

\[\text{flow_values[a,k] = flow_points[k]}\]

\[\text{loss_values[a,k] = ll_val[k]}\]

\[\text{for } k \text{ in points[a]:}\]

\[\text{loss_gradients[a,k] = ll_a[k]}\]

\[\text{loss_intercepts[a,k] = ll_b[k]}\]

\[\text{return (pieces, points, flow_values, loss_values, loss_gradients, loss_intercepts)}\]

\[\text{def Line_Loss_Calculations_Convex(arc_id, R, fixed_loss, trans_cap, arc_loss, num_loss_tranches=0):}\]

\[\text{Sets up the flow points and loss points for convex line losses.}\]
loss_gradients = {}  
loss_intercepts = {}  
loss_values = {}  
flow_values = {}  
pieces = {}  
points = {}  
for a in arc_id:
    # Set up flow points
    if arc_loss[a]:
        flow_points = []
        flow_points.extend( linspace(-trans_cap[a], 0.0, num_tranches[a]+1) )
        flow_points.pop(-1)
        flow_points.extend( linspace(0.0, trans_cap[a], num_tranches[a]+1) )
    else:
        flow_points = [-trans_cap[a], 0.0, trans_cap[a]]
    # In the convex implementation, n flow points implied n-1 flow pieces.
    points[a] = range(len(flow_points))
    pieces[a] = range(len(flow_points)-1)
    # Set up loss points corresponding to the flow points
    if arc_loss[a]:
        (ll_val, ll_a, ll_b) = Line_Loss_Function(flow_points, R[a], fixed_loss[a])
    else:
        ll_val = [0.0]*len(flow_points)
        ll_a = [0.0]*(len(flow_points)-1)
    for k in points[a]:
        flow_values[a,k] = flow_points[k]
        loss_values[a,k] = ll_val[k]
    for k in pieces[a]:
        loss_gradients[a,k] = ll_a[k]
        loss_intercepts[a,k] = ll_b[k]
return (pieces, points, flow_values, loss_values, loss_gradients, loss_intercepts)

def Line_Loss_Function(flow, R, C):
    '''
    Generates a set of points, l, corresponding to the losses
    for each quantity, and calculates the gradients and
    intercepts for lines between each l.
    INPUTS: flow - a list of flow points
    R - resistance on a line
    C - fixed loss on a line
    RETURNS: l - a list of line loss values
            a - a list of gradients for each piece
            b - a list of intercepts for each piece
    '''
    l = []
    a = []
    b = []
    for f in flow:
        l.append( R * f * f + C)
    for i in range(len(flow)-1):
        # Check if two adjacent flows are equal
        if (abs(flow[i+1] - flow[i]) < ZERO):
            a.append( 0.0 )
            b.append( l[i] )
        else:
            a.append( (l[i+1] - l[i])/(flow[i+1] - flow[i]) )
            b.append( l[i] - a[i]*flow[i] )
    return (l,a,b)

def Consumer_Bathtub_Constraint(m, d_y , r_y , CBT_RHS , strategic_nodes, scen_no):
    for n in strategic_nodes:
        m.addConstr( d_y[n] - r_y[n] >= CBT_RHS, "Consumer_Bathtub_%s_%s" %(n,scen_no))
        m.update()

def Generator_Bathtub_Constraint_1(m, d_x , r_x , bt1_rhs, gen_nodes, d_tranches, r_tranches, i) :
    for n in gen_nodes:
        m.addConstr((sum(d_x[n][t] for t in range(len(d_tranches[n])) ) + sum( 
            r_x[n][t] for t in range(len(r_tranches[n])))) - bt1_rhs[n])
        m.addConstr("Bathtub_%s_%s" %(n,i))
        m.update()

def Arc_Stationarity(m, pi, eta1, eta2, lambd, X, arc_id, arc_from, arc_to, scen_no):
    ...  
    First order KKT condition for arc flows.
    ...  
    for a in arc_id:
```

m.addConstr(-pi[arc_from[a]] + pi[arc_to[a]] - eta1[a] + eta2[a] + lambd[a] * X[a] == 0,

  "Arc_Stationarity_%s" % (a, scen_no))


def d_Price_Stationarity(m, gen_nodes, d_tranches , d_prices, d_p , ug , bt_dual1,i):

  '''
  First order KKT condition for arc gen tranches.
  '''
  for n in gen_nodes:
    for t in range(len(d_tranches[n])):
      m.addConstr(d_prices[n][t] - d_p[n] + ug[n][t] + bt_dual1[n], GRB.GREATER_EQUAL , 0,

        "d_Price_Stationarity_%s_%s_%s" %(n,t,i))


def r_Price_Stationarity(m, nodes, gen_nodes, r_tranches, r_prices, r_p , rug

  , bt_dual1, islands, buses , beta ,I,scen_no):

  '''
  First order KKT condition for arc res tranches.
  '''
  for l in islands:
    for n in nodes:
      for t in range(len(r_tranches[n])):
        if Find_Island(buses, beta, n , I)==l:
          if n in gen_nodes:
            m.addConstr(r_prices[n][t] - r_p[l] + rug[n][t] + bt_dual1[n], GRB.GREATER_EQUAL , 0,

            "r_Price_Stationarity_%s_%s_%s" %(n,t, scen_no))
          else:
            m.addConstr(r_prices[n][t] - r_p[l] + rug[n][t], GRB.GREATER_EQUAL , 0,

            "r_Price_Stationarity_%s_%s_%s" %(n,t, scen_no))


def Arc_Stationarity_Losses(m, pi, eta1, eta2, lambd, X, arc_id, arc_from

  , arc_to, phi, ll_a, ll_pieces, scen_no):

  '''
  First order KKT condition for arc flows including convex losses.
  '''
  for a in arc_id:
    m.addConstr(-pi[arc_from[a]] + pi[arc_to[a]] - eta1[a] + eta2[a] + lambd[a] * X[a]

    - quicksum( ll_a[a,k]*phi[a,k] for k in ll_pieces[a])

    == 0, "Arc_Stationarity_%s" %a)


def Loss_Stationarity(m, pi, phi, arc_id, arc_from, arc_to, ll_pieces, scen_no):

  '''
  First order KKT condition for the convex line losses.
  '''
  for a in arc_id:
    m.addConstr(0.5*(pi[arc_from[a]] + pi[arc_to[a]])

    == quicksum( phi[a,k] for k in ll_pieces[a]), "Loss_Stationarity_%s" %a)


def Kirchoff_Loop_Law(m, f, X, theta, arc_id, arc_from, arc_to, scen_no):

  '''
  Kirchoff's loop law giving relations between arc flows and bus power angles.
  Alternative to the sum of voltage differences in a loop = 0.
  '''
  for a in arc_id:
    if not((a=="BEN_HAY1.1") or (a=="BEN_HAY2.1")

      or (a=="HAY_BEN1.1") or (a=="HAY_BEN2.1")):
      m.addConstr( X[a] * f[a]

      == theta[arc_from[a]] - theta[arc_to[a]], "Kirchoff's_Law_%s" %a)


def Power_Angle_Stationarity(m, lambd, buses, arc_id, arc_from, arc_to, scen_no):

  '''
  First order KKT condition for the power angles at buses.
  '''
  for b in buses:
    m.addConstr(quicksum(lambd[a] for a in arc_id if (arc_to[a]==b)

      and not((arc_from[a]=="580" and b=="420")

      or (arc_from[a]=="420" and b=="580")))

    - quicksum(lambd[a] for a in arc_id

      if (arc_from[a]==b) and not((arc_to[a]=="580"

      and b=="420") or (arc_to[a]=="420"

      and b=="580")) == 0.0, "Power_Angle_Stationarity_%s" %(b, scen_no))


def Flow_Complementary_Slackness(m, price, eta1, eta2, trans_cap, f, arc_id, scen_no):

  '''
  Complementary slackness conditions on arc flows at capacity or at reverse capacity.
  '''
```
# Flow Constraints Complementary Slackness

z_eta1 = {}  
z_eta2 = {}

for a in arc_id:
    z_eta1[a] = m.addVar(vtype = GRB.BINARY, name = "Eta1Master_%s_%s" %(a,scen_no))
    z_eta2[a] = m.addVar(vtype = GRB.BINARY, name = "Eta2Master_%s_%s" %(a,scen_no))

m.update()

for a in arc_id:
    m.addConstr(eta1[a] <= 5000* z_eta1[a],
        "Flow_Capacity_1.%s_%s" %(a,scen_no))
    m.addConstr(trans_cap[scen_no][a] - f[a] <= (2*trans_cap[scen_no][a] ) * (1-z_eta1[a]),
        "Flow_Capacity_2.%s_%s" %(a,scen_no))
    m.addConstr(eta2[a] <= 5000* z_eta2[a],
        "Reverse_Flow_Capacity_1.%s_%s" %(a,scen_no))
    m.addConstr(trans_cap[scen_no][a] + f[a] <= (2*trans_cap[scen_no][a] ) * (1-z_eta2[a]),
        "Reverse_Flow_Capacity_2.%s_%s" %(a,scen_no))
    m.addConstr(z_eta1[a] + z_eta2[a] <= 1)

return (z_eta1, z_eta2)

def BT_Complementary_Slackness(m, d_x,r_x, bt_dual1, bt1_rhs, scen_no, gen_nodes, d_tranches , r_tranches):
    z_bt1 = {}  
    for n in gen_nodes:
        z_bt1[n] = m.addVar(vtype = GRB.BINARY, name = "BinaryBTDual_%s_%s" %(n,scen_no))

m.update()

for n in gen_nodes:
    m.addConstr((bt1_rhs[n] - (sum(d_x[n][t] for t in range(len(d_tranches[n])) ) + sum( r_x[n][t] for t in range(len(r_tranches[n]))) ) - 6000*z_bt1[n]),GRB.LESS_EQUAL,0, name = "Comp1_BinaryBTDual_%s_%s" %(n,scen_no))
    m.addConstr((bt_dual1[n]- 6000*(1- z_bt1[n])),GRB.LESS_EQUAL,0, name = "Comp2_BinaryBTDual_%s_%s" %(n,scen_no))

m.update()

def d_Tranche_Complementary_Slackness(m, d_x,d_quantities,d_prices, d_p, ug , bt_dual1, scen_no, gen_nodes, d_tranches):
    d_z1_tranche= {}  
    d_z2_tranche= {}
    for n in gen_nodes:
        d_z1_tranche[n]={}
        d_z2_tranche[n]={}
        for t in range(len(d_tranches[n])):
            d_z1_tranche[n][t] = m.addVar(vtype = GRB.BINARY, name = "d_BinaryTrancheComplementarity1_%s_%s_%s" %(n,t,scen_no))
            d_z2_tranche[n][t] = m.addVar(vtype = GRB.BINARY, name = "d_BinaryTrancheComplementarity2_%s_%s_%s" %(n,t,scen_no))

m.update()

for n in gen_nodes:
    for t in range(len(d_tranches[n])):
        m.addConstr(d_quantities[n][t] - d_x[n][t] - d_quantities[n][t]*(1-d_z1_tranche[n][t]) ,GRB.LESS_EQUAL,0,
            "Tranche_Comp_Slack_1.%s.%s_%s" %(n,t,scen_no))
        m.addConstr(ug[n][t]- 6000*(1- z_bt1[n]),GRB.LESS_EQUAL,0,
            "Tranche_Comp_Slack_2.%s.%s_%s" %(n,t,scen_no))
        m.addConstr( d_x[n][t] - d_quantities[n][t]*(1-d_z2_tranche[n][t]) ,GRB.LESS_EQUAL,0,
            "Tranche_Comp_Slack_3.%s.%s_%s" %(n,t,scen_no))
        m.addConstr((d_prices[n][t] - d_p[n] + ug[n][t] + bt_dual1[n] - 6000*d_z2_tranche[n][t] ),GRB.LESS_EQUAL,0,
            "Tranche_Comp_Slack_4.%s.%s_%s" %(n,t,scen_no))

m.update()

return( d_z1_tranche,d_z2_tranche)

def d_improve_Big_M( m, scen_no, gen_nodes, d_tranches, d_z1_tranche, d_z2_tranche):
    for n in gen_nodes:
        for t in range(len(d_tranches[n])):
            if t!= len(d_tranches[n])-1 :
                m.addConstr(d_z1_tranche[n][t]-d_z1_tranche[n][t+1] ,GRB.GREATER_EQUAL,0,
                    "d_Improve_Big_M_1.%s.%s_%s" %(n,t,scen_no))
                m.addConstr(d_z2_tranche[n][t]-d_z2_tranche[n][t+1] ,GRB.LESS_EQUAL,0,
                    "d_Improve_Big_M_2.%s.%s_%s" %(n,t,scen_no))

m.update()
```python
def r_Tranche_Complementary_Slackness(m, r_x, r_quantities, r_prices, r_p, rug, bt_dual1, scen_no, nodes, gen_nodes, r_tranches, islands, buses, beta, I):
    r_z1_tranche = {}
    r_z2_tranche = {}
    for n in nodes:
        r_z1_tranche[n] = {}
        r_z2_tranche[n] = {}
        for t in range(len(r_tranches[n])):
            r_z1_tranche[n][t] = m.addVar(vtype = GRB.BINARY, name = "r_BinaryTrancheComplementarity1_%s_%s_%s" % (n, t, scen_no))
            r_z2_tranche[n][t] = m.addVar(vtype = GRB.BINARY, name = "r_BinaryTrancheComplementarity2_%s_%s_%s" % (n, t, scen_no))
        m.update()
        for n in nodes:
            for t in range(len(r_tranches[n])):
                m.addConstr(r_quantities[n][t] - r_x[n][t] - r_quantities[n][t] * (1 - r_z1_tranche[n][t]), GRB.LESS_EQUAL, 0,
                           "r_Tranche_Comp_Slack_1.%s.%s_%s" % (n, t, scen_no))
                m.addConstr((rug[n][t] - 6000 * r_z1_tranche[n][t]), GRB.LESS_EQUAL, 0,
                           "r_Tranche_Comp_Slack_2.%s.%s_%s" % (n, t, scen_no))
                m.addConstr(r_x[n][t] - r_quantities[n][t] * (1 - r_z2_tranche[n][t]), GRB.LESS_EQUAL, 0,
                           "r_Tranche_Comp_Slack_3.%s.%s_%s" % (n, t, scen_no))
            for l in islands:
                if Find_Island(buses, beta, n, I) == l:
                    if n in gen_nodes:
                        m.addConstr((r_prices[n][t] - r_p[l] + rug[n][t] + bt_dual1[n] - 6000 * r_z2_tranche[n][t]), GRB.LESS_EQUAL, 0,
                                   "r_Tranche_Comp_Slack_4.%s.%s_%s" % (n, t, scen_no))
                else:
                    m.addConstr((r_prices[n][t] - r_p[l] + rug[n][t] - 6000 * r_z2_tranche[n][t]), GRB.LESS_EQUAL, 0,
                                   "r_Tranche_Comp_Slack_4.%s.%s_%s" % (n, t, scen_no))
        m.update()
    return (r_z1_tranche, r_z2_tranche)

def r_improve_Big_M(m, scen_no, nodes, r_tranches, r_z1_tranche, r_z2_tranche):
    for n in nodes:
        for t in range(len(r_tranches[n])):
            if t != len(r_tranches[n]) - 1:
                m.addConstr(r_z1_tranche[n][t] - r_z1_tranche[n][t + 1], GRB.GREATER_EQUAL, 0,
                            "r_Improve_Big_M_1.%s.%s_%s" % (n, t, scen_no))
                m.addConstr(r_z2_tranche[n][t] - r_z2_tranche[n][t + 1], GRB.LESS_EQUAL, 0,
                            "r_Improve_Big_M_2.%s.%s_%s" % (n, t, scen_no))
            m.addConstr(sum(r_z1_tranche[n][t] + r_z2_tranche[n][t] for t in range(len(r_tranches[n]))), GRB.LESS_EQUAL, len(r_tranches[n]),
                        "r_Improve_Big_M_3.%s.%s" % (n, scen_no))
            m.addConstr(sum(r_z1_tranche[n][t] + r_z2_tranche[n][t] for t in range(len(r_tranches[n]))), GRB.GREATER_EQUAL, len(r_tranches[n]) - 1,
                        "r_Improve_Big_M_4.%s.%s" % (n, scen_no))
        m.update()

def d_IP_Constraints(m, strategic_nodes, d_y, d_p, n_d):
    # Integer programming constraints to ensure that marginal price and residual demand points are monotonic
    #
    # Enumeration Constraints
    d_z = {}
    for n in strategic_nodes:
        d_z[n] = {}
        for i in n_d:
            for j in n_d:
                if j != i:
                    d_z[n][i, j] = m.addVar(vtype = GRB.BINARY, name = "d_IP_Z_Var_%s_%s_%s" % (n, i, j))
        m.update()
    for n in strategic_nodes:
        for i in n_d:
            for j in n_d:
                if i != j:
                    m.addConstr(d_y[i][n] <= d_y[j][n] + 8000 * d_z[n][i, j], name = "d_IP_Constraints1_%s_%s_%s" % (n, i, j))
                m.addConstr(d_p[j][n] <= d_p[i][n] + 5000 * d_z[n][i, j], name = "d_IP_Constraints2_%s_%s_%s" % (n, i, j))
```

Appendix A Gurobipy: Bi-parametric Sensitivity Analysis Region Construction

---

```python
def r_Tranche_Complementary_Slackness(m, r_x, r_quantities, r_prices, r_p, rug, bt_dual1, scen_no, nodes, gen_nodes, r_tranches, islands, buses, beta, I):
    r_z1_tranche = {}
    r_z2_tranche = {}
    for n in nodes:
        r_z1_tranche[n] = {}
        r_z2_tranche[n] = {}
        for t in range(len(r_tranches[n])):
            r_z1_tranche[n][t] = m.addVar(vtype = GRB.BINARY, name = "r_BinaryTrancheComplementarity1_%s_%s_%s" % (n, t, scen_no))
            r_z2_tranche[n][t] = m.addVar(vtype = GRB.BINARY, name = "r_BinaryTrancheComplementarity2_%s_%s_%s" % (n, t, scen_no))
        m.update()
        for n in nodes:
            for t in range(len(r_tranches[n])):
                m.addConstr(r_quantities[n][t] - r_x[n][t] - r_quantities[n][t] * (1 - r_z1_tranche[n][t]), GRB.LESS_EQUAL, 0,
                           "r_Tranche_Comp_Slack_1.%s.%s_%s" % (n, t, scen_no))
                m.addConstr((rug[n][t] - 6000 * r_z1_tranche[n][t]), GRB.LESS_EQUAL, 0,
                           "r_Tranche_Comp_Slack_2.%s.%s_%s" % (n, t, scen_no))
                m.addConstr(r_x[n][t] - r_quantities[n][t] * (1 - r_z2_tranche[n][t]), GRB.LESS_EQUAL, 0,
                           "r_Tranche_Comp_Slack_3.%s.%s_%s" % (n, t, scen_no))
            for l in islands:
                if Find_Island(buses, beta, n, I) == l:
                    if n in gen_nodes:
                        m.addConstr((r_prices[n][t] - r_p[l] + rug[n][t] + bt_dual1[n] - 6000 * r_z2_tranche[n][t]), GRB.LESS_EQUAL, 0,
                                   "r_Tranche_Comp_Slack_4.%s.%s_%s" % (n, t, scen_no))
                else:
                    m.addConstr((r_prices[n][t] - r_p[l] + rug[n][t] - 6000 * r_z2_tranche[n][t]), GRB.LESS_EQUAL, 0,
                                   "r_Tranche_Comp_Slack_4.%s.%s_%s" % (n, t, scen_no))
        m.update()
    return (r_z1_tranche, r_z2_tranche)

def r_improve_Big_M(m, scen_no, nodes, r_tranches, r_z1_tranche, r_z2_tranche):
    for n in nodes:
        for t in range(len(r_tranches[n])):
            if t != len(r_tranches[n]) - 1:
                m.addConstr(r_z1_tranche[n][t] - r_z1_tranche[n][t + 1], GRB.GREATER_EQUAL, 0,
                            "r_Improve_Big_M_1.%s.%s_%s" % (n, t, scen_no))
                m.addConstr(r_z2_tranche[n][t] - r_z2_tranche[n][t + 1], GRB.LESS_EQUAL, 0,
                            "r_Improve_Big_M_2.%s.%s_%s" % (n, t, scen_no))
            m.addConstr(sum(r_z1_tranche[n][t] + r_z2_tranche[n][t] for t in range(len(r_tranches[n]))), GRB.LESS_EQUAL, len(r_tranches[n]),
                        "r_Improve_Big_M_3.%s.%s" % (n, scen_no))
            m.addConstr(sum(r_z1_tranche[n][t] + r_z2_tranche[n][t] for t in range(len(r_tranches[n]))), GRB.GREATER_EQUAL, len(r_tranches[n]) - 1,
                        "r_Improve_Big_M_4.%s.%s" % (n, scen_no))
        m.update()

def d_IP_Constraints(m, strategic_nodes, d_y, d_p, n_d):
    # Integer programming constraints to ensure that marginal price and residual demand points are monotonic
    #
    # Enumeration Constraints
    d_z = {}
    for n in strategic_nodes:
        d_z[n] = {}
        for i in n_d:
            for j in n_d:
                if j != i:
                    d_z[n][i, j] = m.addVar(vtype = GRB.BINARY, name = "d_IP_Z_Var_%s_%s_%s" % (n, i, j))
        m.update()
    for n in strategic_nodes:
        for i in n_d:
            for j in n_d:
                if i != j:
                    m.addConstr(d_y[i][n] <= d_y[j][n] + 8000 * d_z[n][i, j], name = "d_IP_Constraints1_%s_%s_%s" % (n, i, j))
                    m.addConstr(d_p[j][n] <= d_p[i][n] + 5000 * d_z[n][i, j], name = "d_IP_Constraints2_%s_%s_%s" % (n, i, j))
```
for i in n_d:
    for j in n_d:
        if j > i:
            m.addConstr(d_z[n][i,j] + d_z[n][j,i] == 1,
                        name = "d_IP_Constraints3_%s_%s_%s" % (n,i,j))

def r_IP_Constraints(m, strategic_nodes, r_y, r_p, n_d, islands, buses, beta, I):
    r_z = {}  
    for n in strategic_nodes:
        r_z[n] = {}
        for i in n_d:
            for j in n_d:
                if i != j:
                    r_z[n][i,j] = m.addVar(vtype = GRB.BINARY,
                                            name = "r_IP_Z_Var_%s_%s_%s" % (n,i,j))
    m.update()
    for n in strategic_nodes:
        for l in islands:
            if Find_Island(buses, beta, n, I) == l:
                for i in n_d:
                    for j in n_d:
                        if j != i:
                            m.addConstr(r_y[i][n] <= r_y[j][n] + 8000*r_z[n][i,j],
                                        name = "r_IP_Constraints1_%s_%s_%s" % (n,i,j))
                            m.addConstr(r_p[i][l] <= r_p[j][l] + 5000*r_z[n][i,j],
                                        name = "r_IP_Constraints2_%s_%s_%s" % (n,i,j))
                for i in n_d:
                    for j in n_d:
                        if j > i:
                            m.addConstr(r_z[n][i,j] + r_z[n][j,i] == 1,
                                            name = "r_IP_Constraints3_%s_%s_%s" % (n,i,j))
    def Tranche_Constraints(m, nodes, d_y, p, tranche_no, n_d):
        Qa = {}
        Pa = {}
        gammaP = {}
        gammaQ = {}
        for n in nodes:
            Qa[n] = {}
            Pa[n] = {}
            gammaP[n] = {}
            gammaQ[n] = {}
            Qa[n][0] = 0
            Pa[n][0] = 0
            Pa[n][num_tranches+1] = 2000
            for i in range(0,num_tranches+1):
                Qa[n][i] = m.addVar(name = 'Residual_Demand_Level_%s_%s' % (n,i))
                Pa[n][i] = m.addVar(name = 'Marginal_Price_Level_%s_%s' % (n,i))
                gammaP[n][i] = {}  
                gammaQ[n][i] = {}
                for j in n_d:
                    if i != 0:
                        gammaP[n][i][j] = m.addVar(vtype = GRB.BINARY,
                                                    name = 'gammaP'+str(i)+'_'+str(j))
                        gammaQ[n][i][j] = m.addVar(vtype = GRB.BINARY,
                                                    name = 'gammaQ'+str(i)+'_'+str(j))
            Pa[n][num_tranches+2] = m.addVar(vtype = GRB.BINARY,
                                                name = 'Pa'+str(i))
        m.update()
    for n in strategic_nodes:
        for l in islands:
            if Find_Island(buses, beta, n, I) == l:
                for i in n_d:
                    for j in n_d:
                        if j > i:
                            m.addConstr(r_z[n][i,j] + r_z[n][j,i] == 1,
                                            name = "r_IP_Constraints3_%s_%s_%s" % (n,i,j))
    def Tranche_Constraints(m, nodes, d_y, p, tranche_no, n_d):
        Qa = {}
        Pa = {}
        gammaP = {}
        gammaQ = {}
        for n in nodes:
            Qa[n] = {}
            Pa[n] = {}
            gammaP[n] = {}
            gammaQ[n] = {}
            Qa[n][0] = 0
            Pa[n][0] = 0
            Pa[n][num_tranches+1] = 2000
            for i in range(0,num_tranches+1):
                Qa[n][i] = m.addVar(name = 'Residual_Demand_Level_%s_%s' % (n,i))
                Pa[n][i] = m.addVar(name = 'Marginal_Price_Level_%s_%s' % (n,i))
                gammaP[n][i] = {}  
                gammaQ[n][i] = {}
                for j in n_d:
                    if i != 0:
                        gammaP[n][i][j] = m.addVar(vtype = GRB.BINARY,
                                                    name = 'gammaP'+str(i)+'_'+str(j))
                        gammaQ[n][i][j] = m.addVar(vtype = GRB.BINARY,
                                                    name = 'gammaQ'+str(i)+'_'+str(j))
            Pa[n][num_tranches+2] = m.addVar(vtype = GRB.BINARY,
                                                name = 'Pa'+str(i))
        m.update()
for i in range(1,num_tranches+1):
    temp.append(gammaP[n][i][j])
    m.addSOS(GRB.SOS_TYPE1, temp)
for i in range(0,num_tranches+1):
    for j in n_d:
        m.addConstr(p[j][n]>=Pa[n][i]-4000*(1-gammaQ[n][i][j]))
        m.addConstr(p[j][n]<=Pa[n][i+1]+4000*(1-gammaQ[n][i][j]))
        m.addConstr(y[j][n]>=Qa[n][i]-4000*(1-gammaQ[n][i][j]))
        m.addConstr(y[j][n]<=Qa[n][i]+4000*(1-gammaQ[n][i][j]))
for i in range(1,num_tranches+1):
    for j in n_d:
        m.addConstr(p[j][n]>=Pa[n][i]-4000*(1-gammaP[n][i][j]))
        m.addConstr(p[j][n]<=Pa[n][i]+4000*(1-gammaP[n][i][j]))
        m.addConstr(y[j][n]>=Qa[n][i-1]-4000*(1-gammaP[n][i][j]))
        m.addConstr(y[j][n]<=Qa[n][i]+4000*(1-gammaP[n][i][j]))

for i in range(1,num_tranches):
    m.addConstr(Pa[n][i]<=Pa[n][i+1])
    m.addConstr(Qa[n][i]<=Qa[n][i+1])
    m.addConstr(Qa[n][0] == 0)

def Get_BT1_RHS_Data(tp , f_bt1rhs):
    f1 = csv.reader(open(f_bt1rhs, 'r'))
    f1.next()
    f1.next()
    bt1_rhs = {}
    for bt1rhs_csv in f1:
        if (bt1rhs_csv[0] == tp):
            bt1_rhs[bt1rhs_csv[1]] = float(bt1rhs_csv[2])
    return bt1_rhs

def Get_Reserve_Data(tp, f_reserve):
    f1 = csv.reader(open(f_reserve, 'r'))
    f1.next()
    islands = ['NI', 'SI']
    reserves = {}
    for i in islands:
        reserves[i] = 0.0
    for reserve_csv in f1:
        if (reserve_csv[1] == tp):
            reserves[reserve_csv[2]] = float(reserve_csv[4])
    return (islands, reserves)

def Get_Node_Data(tp, f_node, f_demand):
    ... Returns node data from vSPD data for a particular trading period. ...
    Return node data from vSPD data for a particular trading period.
    INPUTS:
    tp - Relevant trading period
    f_node - File name for "TradePeriodNode" data
    f_demand - File name for "TradePeriodNodeDemand" data
    OUTPUTS:
    nodes - List of nodes
    demands - Dict of demands (NODES)

    ...
appendix A Gurobipy: Bi-parametric Sensitivity Analysis Region Construction

return (nodes, demands)

def Get_Gen_Nodes(tp , f_gen_node):
    f1 = csv.reader(open(f_gen_node, 'r'))
    f1.next()
    f1.next()
    gen_nodes = []
    dem_nodes = []
    for gen_node_csv in f1:
        # Check trading period
        if (gen_node_csv[0] == tp):
            if (gen_node_csv[2] == "t1"):
                if(gen_node_csv[3] == "i_GenerationMWOffer"):
                    gen_nodes.append(gen_node_csv[1])
    return gen_nodes

def Get_Dem_Nodes(tp , f_dem_node):
    f1 = csv.reader(open(f_dem_node, 'r'))
    f1.next()
    f1.next()
    dem_nodes = []
    for dem_node_csv in f1:
        # Check trading period
        if (dem_node_csv[0] == tp):
            dem_nodes.append(dem_node_csv[1])
    return dem_nodes

def Get_Bus_Data(tp, f_busisland):
    
    Returns bus data from vSPD for a particular trading period.
    INPUTS:
    tp - Relevant trading period
    f_busisland - File name for "TradePeriodBusIsland" data
    OUTPUTS:
    buses - list of buses
    I - Dictionary mapping each bus to "NI" or "SI", North or South Island.

    f1 = csv.reader(open(f_busisland, 'r'))
    # Headers
    f1.next()
    f1.next()
    buses = []
    I1 = []
    # Get list of buses
    for csv_bus in f1:
        if (csv_bus[0] == tp):
            buses.append(csv_bus[1])
            I1.append(csv_bus[2])
    I ={}
    I = dict(zip(buses,I1))
    return (buses, I)

def Get_Bus_Allocation_Factor(tp, nodes, buses, f_nodebusallocation):
    
    Constructs the beta matrix describing which buses appear at which nodes,
    as well as the respective allocation factors.
    INPUTS:
    tp - Relevant trading period
    nodes - List of nodes
    buses - List of buses
    f_nodebusallocation - File name for "TradePeriodNodeBusAllocationFactor" data
    OUTPUTS:
    beta - node-bus allocation factor dictionary.

    f1 = csv.reader(open(f_nodebusallocation, 'r'))
    # Headers
    f1.next()
    f1.next()
    # Initialise beta to be all zeros, so all key values are present
    beta = {}
    for n in nodes:
        for b in buses:
            beta[n,b] = 0
    # Populate beta using vSPD data
    for csv_busallocation in f1:
if (csv_busallocation[0] == tp):
    beta[csv_busallocation[1], csv_busallocation[2]] = float(csv_busallocation[3])
return (beta)

def Get_Arc_Data(tp, f_branchdefn, f_branchstatus, f_branchcapacity, f_branchparam):
    '''
    Returns arc data from vSPD data for a particular trading period.
    
    INPUTS:
    tp - Relevant trading period
    f_branchdefn - File name for "TradePeriodBranchDefn" data
    f_branchstatus - File name for "TradePeriodBranchOpenStatus" data
    f_branchcapacity - File name for "TradePeriodBranchCapacity" data
    f_branchparam - File name for "TradePeriodBranchParam" data
    
    OUTPUTS:
    arc_id - List of arc identifiers (names)
    arc_from - Dict of arc heads {ARC_ID}
    arc_to - Dict of arc tails {ARC_ID}
    arc_cap - Dict of arc capacities {ARC_ID}
    fixed_loss - Fixed loss on each arc {ARC_ID}
    R - Resistance on each arc {ARC_ID}
    X - Reactance on each arc {ARC_ID}
    num_tranches - Number of loss tranches on each arc {ARC_ID}
    '''
    f1 = csv.reader(open(f_branchdefn, 'r'))
    f2 = csv.reader(open(f_branchstatus, 'r'))
    f3 = csv.reader(open(f_branchcapacity, 'r'))
    f4 = csv.reader(open(f_branchparam, 'r'))
    # Headers
    f1.next()
    f1.next()
    f2.next()
    f2.next()
    f3.next()
    f3.next()
    f4.next()
    f4.next()
    arc_id = []
    arc_from = []
    arc_to = []
    arc_cap = []
    fixed_loss = []
    R = []
    X = []
    num_tranches = []
    for csv_branchdefn in f1:
        status_csv = f2.next()
        capacities_csv = f3.next()
        # Check open status
        if (csv_branchdefn[0] == tp) & (status_csv[2] == "0"):
            # Arcs
            arc_id.append( csv_branchdefn[1] )
            arc_from.append( csv_branchdefn[2] )
            arc_to.append( csv_branchdefn[3] )
            arc_cap.append( float(capacities_csv[2]) )
            # Fixed losses
            fixed_loss.append( 0.0 * float(param_csv[3]) )
            # Resistance
            param_csv = f4.next()
            R.append( 0.1 * float(param_csv[3]) )
            # Reactance
            param_csv = f4.next()
            if (abs(float(param_csv[3])) < ZERO):
                X.append( 1.0 / float(param_csv[3]) )
            else:
                X.append( 1.0 / float(param_csv[3]) )
        else:
            param_csv = f4.next()
            num_tranches.append( 0 )
        f1.next()
arc_cap = dict(zip(arc_id, arc_cap))
fixed_loss = dict(zip(arc_id, fixed_loss))
R = dict(zip(arc_id, R))
X = dict(zip(arc_id, X))
num_tranches = dict(zip(arc_id, num_tranches))

return (arc_id, arc_from, arc_to, arc_cap, fixed_loss, R, X, num_tranches)

def Get_Energy_Offer_Data(tp, f_energyoffer, strategic_nodes):
    
    Returns offer data from uSPD for a particular trading period.

    INPUTS:
    tp - Relevant trading period
    f_energyoffer - File name for “TradePeriodEnergyOffer” data
    strategic_nodes - A list of strategic node names - offers at these nodes will be skipped.

    OUTPUTS:
    offer_data - A dictionary indexed by {TRANCHES} containing a tuple of
                  (price, quantity, tranch_node) for each tranch.
    
    
    f = csv.reader(open(f_energyoffer, 'r'))
    f.next()
    f.next()
    d_tranches = []
    d_prices = []
    d_quantities = []
    d_tranch_nodes = []
    for csv_offer in f:
        if (csv_offer[0] == tp):
            # Ignore offers on strategic nodes
            if not(csv_offer[1] in strategic_nodes):
                d_quantities.append( float(csv_offer[4]) )
                csv_offer = f.next()
                d_prices.append( float(csv_offer[4]) )
                d_tranch_nodes.append(csv_offer[1])
        return dict(zip( d_tranches, zip(d_prices, d_quantities, d_tranch_nodes) ))

def Get_Reserve_Offer_Data(tp, f_sustainedILRoffer, f_sustainedPLSRoffer, f_sustainedTWDRoffer, strategic_nodes):
    
    Returns reserve offer data from uSPD for a particular trading period.

    INPUTS:
    tp - Relevant trading period
    f_sustainedILRoffer - File name for “TradePeriodReserveOffer” data
    strategic_nodes - A list of strategic node names - offers at these nodes will be skipped.

    OUTPUTS:
    reserve_offer_data - A dictionary indexed by {TRANCHES} containing a tuple of
                         (price, quantity, tranch_node) for each tranch.
    
    
    sum_reserve = 0
    f1 = csv.reader(open(f_sustainedILRoffer, ‘r’))
    f1.next()
    f1.next()
    f2 = csv.reader(open(f_sustainedPLSRoffer, ‘r’))
    f3 = csv.reader(open(f_sustainedTWDRoffer, ‘r’))
    f2.next()
    f2.next()
    f3.next()
    f3.next()
    r_tranches = []
    r_prices = []
    r_quantities = []
    r_tranch_nodes = []
    for csv_offer in f1:
        if (csv_offer[0] == tp):
            # Ignore offers on strategic nodes
            if not(csv_offer[1] in strategic_nodes):
                r_quantities.append( float(csv_offer[4]) )
                sum_reserve = sum_reserve + float(csv_offer[4])
                csv_offer = f1.next()
                r_prices.append( float(csv_offer[4]) )
                r_tranch_nodes.append(csv_offer[1])
    for csv_offer in f2:
        if (csv_offer[0] == tp):
            # Ignore offers on strategic nodes
            if not(csv_offer[1] in strategic_nodes):
if not(csv_offer[1] in strategic_nodes):
    r_tranches.append(csv_offer[0] + " T" + csv_offer[2])
    r_quantities.append(float(csv_offer[4]))
    sum_reserve = sum_reserve + float(csv_offer[4])
    csv_offer = f3.next()
    r_prices.append(float(csv_offer[4]))
    r_tranch_nodes.append(csv_offer[1])

for csv_offer in f3:
    if (csv_offer[0] == tp):
        # Ignore offers on strategic nodes
        if not(csv_offer[1] in strategic_nodes):
            r_tranches.append(csv_offer[0] + " T" + csv_offer[2])
            r_quantities.append(float(csv_offer[4]))
            sum_reserve = sum_reserve + float(csv_offer[4])
            csv_offer = f3.next()
            r_prices.append(float(csv_offer[4]))
            r_tranch_nodes.append(csv_offer[1])

return (dict(zip(r_tranches, zip(r_prices, r_quantities, r_tranch_nodes))))

def Find_Node(nodes, beta, b):
    '''
    Returns the node name of a bus
    '''
    for n in nodes:
        if beta[n,b] > 0.0:
            return n

def Find_Island(buses, beta, n, I):
    '''
    Returns the island of a node
    '''
    for b in buses:
        if beta[n, b] > 0.0:
            return I[b]
            break

def Strategic_Consumer_Data():
    strategic_nodes = ['TWI2201']
    d_gen_cap = {}  # consumer's capacity in each node
    r_gen_cap = {}  
    for n in strategic_nodes:
        d_gen_cap[n] = 800
        r_gen_cap[n] = 800
    gen_q = {}  
    for n in strategic_nodes:
        if n in strategic_nodes:
            gen_q[n] = linspace(0.0, d_gen_cap['TWI2201'], 10)
    gen_c = {}  
    for n in strategic_nodes:
        gen_c[n] = polynomial(0.05, 20.0, 100, gen_q['TWI2201'])
    return (strategic_nodes, d_gen_cap, gen_q, gen_c, r_gen_cap)

def polynomial(a, b, c, x):
    y = []
    for i in x:
        y.append(c - a*i*i + b*i)
    return y

def Contract_Data(nodes, buses, beta, I, demands):  
    strategic_retail_share = 0.18
    strategic_industrial_share = 0.18
    north_retail = 0.66  # Percentage of retail demand in NI
    south_retail = 0.42  # Percentage of retail demand in SI
    contracts = []
    contract_quantity = []
    N = {}  
    for n in nodes:
        contract_name = "CONTRACT_" + n
        contracts.append(contract_name)
        island = Find_Island(buses, beta, n, I)
        if island == "NI":
            amount = (demands[n] * strategic_retail_share * north_retail
                      + demands[n] * strategic_industrial_share * (1.0-north_retail))
            N[n] = amount / (strategic_retail_share + strategic_industrial_share)
elif island == "SI":
    amount = (demands[n] * strategic_retail_share + demands[n] * strategic_industrial_share)
    contract_quantity[contract_name] = amount

N[contract_name] = n
return (contracts, contract_quantity, N)

def Get_incumbent_data(scen_no, f_incumbentvalues, strategic_nodes):
    f = csv.reader(open(f_incumbentvalues, 'r'))
    f.next()
    scen_st = str(scen_no)
    d_y = []
    d_pi = []
    r_y = []
    r_pi = []
    for csv_value in f:
        if (csv_value[0] == scen_st):
            d_y.append(float(csv_value[1]))
            d_pi.append(float(csv_value[2]))
            r_y.append(float(csv_value[3]))
            r_pi.append(float(csv_value[4]))
    return (d_y,d_pi,r_y,r_pi)

def plotRegions(stored,emax,rmax):
    fig = plt.figure()
    ax = fig.add_subplot(111)
    for i in range(len(stored)):
        verts=
        p=stored[i][1][1]/emax
        p2=stored[i][1][1]/rmax
        if p>1.0:
            p=1.0
        if p2>1.0:
            p2=1.0
        codes=[Path.MOVETO]
        for j in range(len(stored[i])-1):
            codes.append(Path.LINETO)
            codes.append(Path.CLOSEPOLY)
        for j in range(len(stored[i])):
            verts.append((stored[i][j][0][0],stored[i][j][0][1]))
        path = Path(verts, codes)
        patch = patches.PathPatch(path, facecolor=(p,0,p2), lw=2,alpha=1)
        ax.add_patch(patch)
    ax.set_xlim(0,800)
    ax.set_ylim(0,100)
    plt.show()

def Get_scen_Data(f_tp):
    f = csv.reader(open(f_tp, 'r'))
    f.next()
    for csv_value in f:
        scen_list = float(csv_value[0])
        date = float(csv_value[1])
    return(scen_list , date)

def main():
    start = time.time()
    d_price = {}  
    r_price = {}  
    d_quantity = {}  
    r_quantity = {}  
    demands = {}  
    reserves = {}  
    nodes = {}  
    gen_nodes = {}  
    dem_nodes = {}  
    d_tranche = {}  
    r_tranche = {}  
    d_offer_data = {}  
    r_offer_data = {}  
    d_M = {}  
    r_M = {}  
    d_y_inc_val= {}
d_pi_inc_val= {}
r_y_inc_val= {}
r_pi_inc_val= {}
b1_rhs= {}  
contracts = {}  
contract_quantity = {}  
N = {}  
I = {}  
beta = {}  
buses = {}  
arc_id = {}  
arc_from = {}  
arc_to = {}  
arc_cap = {}  
CET_RHS = 0  
rho= {}  

(strategic_nodes, d_gen_cap, gen_q, gen_c, r_gen_cap) = Strategic_Consumer_Data()  
folder = './data/'  
(scen_list, date) = Get_scen_Data('scen_data.csv')  
for sc in scen_list :  
  d_price[sc] = {}  
  r_price[sc] = {}  
  d_quantity[sc] = {}  
  r_quantity[sc] = {}  
  reserves[sc] = {}  
  node[node[sc]] = {}  
  gen_nodes[sc] = {}  
  dem_nodes[sc] = {}  
  d_tranche[sc] = {}  
  r_tranche[sc] = {}  
  d_offer_data[sc] = {}  
  r_offer_data[sc] = {}  
  d_M[sc] = {}  
  r_M[sc] = {}  
  d_y_inc_val[sc] = {}  
  d_pi_inc_val[sc] = {}  
  r_y_inc_val[sc] = {}  
  r_pi_inc_val[sc] = {}  
  bt1_rhs[sc] = {}  
  demands[sc] = {}  
  contracts[sc] = {}  
  contract_quantity[sc] = {}  
  N[sc] = {}  
  I[sc] = {}  
  beta[sc] = {}  
  buses[sc] = {}  
  arc_id[sc] = {}  
  arc_from[sc] = {}  
  arc_to[sc] = {}  
  arc_cap[sc] = {}  
  X[sc] = {}  
  d_gen_cap[sc] = {}  
  gen_q[sc] = {}  
  gen_c[sc] = {}  
  r_gen_cap[sc] = {}  
  r_gen_q[sc] = {}  
  r_gen_c[sc] = {}  
  rho[sc] = {}  

tp = 'TP' + str(sc+1)  
(nodes[sc], demands[sc]) = Get_Node_Data(tp, folder + date = 'TradePeriodNode.csv')  
(folder + date = 'TradePeriodNodeDemand.csv')  
(islands, reserves[sc]) = Get_Reserve_Data(tp, folder + date = 'TradePeriodIslandReserve.csv')  
(bt1_rhs[sc] = Get_BT1_RHS_Data(tp, folder + date = 'TradePeriodBT1RHS1.csv')  
(buses[sc], I[sc]) = Get_Bus_Data(tp, folder + date = 'TradePeriodBusIsland.csv')  
(beta[sc]) = Get_Bus_Allocation_Factor(tp, nodes[sc], buses[sc], folder + date = 'TradePeriodBusAllocationFactor.csv')
Appendix A: Gurobipy: Bi-parametric Sensitivity Analysis Region Construction

```python
(arc_id[sc], arc_from[sc], arc_to[sc], arc_cap[sc],
 fixed_loss, R[sc], X[sc], num_loss_tranches) = Get_Arc_Data(tp, folder + date + "TradePeriodBranchDefn.csv",
 folder + date + "TradePeriodBranchOpenStatus.csv",
 folder + date + "TradePeriodBranchCapacity.csv",
 folder + date + "TradePeriodBranchParameter.csv")

gen_nodes[sc] = Get_Gen_Nodes(tp , folder + date + "TradePeriodEnergyOffer.csv"
) dem_nodes[sc]= Get_Dem_Nodes(tp , folder + date + "TradePeriodNodeDemand.csv")
(d_offer_data[sc])= Get_Energy_Offer_Data(tp, folder + date + "TradePeriodEnergyOffer.csv", strategic_nodes)
d_tranche[sc] = d_offer_data[sc].keys()
(d_price[sc], d_quantity[sc], d_M[sc]) = splitDict(d_offer_data[sc])
(r_offer_data[sc])= Get_Reserve_Offer_Data(tp, folder + date + "TradePeriodSustainedILROffer.csv",folder + date + "TradePeriodSustainedPLSROffer.csv",folder + date + "TradePeriodSustainedTWDROffer.csv",strategic_nodes)
r_tranche[sc] = r_offer_data[sc].keys()
(r_price[sc], r_quantity[sc], r_M[sc])= splitDict(r_offer_data[sc])
(contracts[sc], contract_quantity[sc], H[sc]) = Contract_Data(nodes[sc], buses[sc], beta[sc], I[sc], demands[sc])

for scen in scen_list:
    m = Model("Dispatch")
    RHS = []
    createLP(scen)
    oldVBase = []
    oldCBase = []
    Bmatrix = []
    binv = None
    pos_b = []
    basisLB = []
    basisUB = []
    xb = []
    dvars = m.getVars()
    constrs = m.getConstrs()
    a = get_matrix_coo(m)
    print(len(a))
    print(len(a[0]))
    numV = len(dvars)
    rangeV = range(numV)
    rangeC = range(len(constrs))
    for i in range(len(m.getConstrs())):
        RHS.append(constrs[i].RHS)
    I2 = np.matrix(np.identity(len(a[0])))[np.ix_(range(len(a[0])), [0, 1])]
    stored = findRegions(0, 0)
    stored = reduceRegions(stored, 0)
    stored2 = findVertical(stored)
    print(stored)
    end = time.time()
    print(end - start)
    #plotRegions(stored, 10, 10)
    stored = stored + stored2
    #outputPoints(stored[1])
    #plotRegions(stored, 50, 50)

    region_count = -1
    with open('regions_%s.csv' % (day), 'ab') as fp0:
        all0 = csv.writer(fp0)
        for l in range(len(stored)):
            region_count = region_count + 1
            for j in range(len(stored[l])):
                all0.writerow([str(day), str(scen), str(region_count), str(stored[l][j][1][0]), str(stored[l][j][1][1]), stored[l][j][0][0], stored[l][j][0][1]])

if __name__ == '__main__':
    main()
```
Appendix B

Gurobipy: Construction of Optimal Monotone Bid and Offer Curves

```python
from pulp import *
from gurobipy import import random as random
import pickle as pickle
import csv as csv
from pprint import pprint
import numpy
from numpy import linspace
from numpy import median
ZERO = 1e-5
max_e_price = 2000
max_r_price = 2000

def Get_TP_Data(f_tp,f_day,sim_no):
    tp_type = ()
    season = ()
    scens_pool = ()
    days = ()
    f = csv.reader(open(f_tp, 'r'))
    f.next()
    for csv_value in f:
        tp_type[sim_no] = int(csv_value[0])
        season[sim_no] = int(csv_value[1])
        scens_pool[sim_no] = range(int(csv_value[2]),int(csv_value[3]))
    f = csv.reader(open(f_day, 'r'))
    f.next()
    for csv_value in f:
        days[sim_no] = int(csv_value[0])
    return(tp_type,season,scens_pool,days)

def Get_u_Values(f_ut):
    utilities = []
    f = csv.reader(open(f_ut, 'r'))
    f.next()
    for csv_value in f:
        utilities.append(float(csv_value[0]))
    return(utilities)

def Get_regions_data(day, scen, f_region):
    y_d = ()
    y_r = ()
```

187
pi_d = {}  
pi_r = {}  
for csv_offer in f1:  
    if (int(csv_offer[0]) == day):  
        if (int(csv_offer[1]) == scen):  
            pi_d[int(csv_offer[2])] = []  
            pi_r[int(csv_offer[2])] = []  
            y_d[int(csv_offer[2])] = []  
            y_r[int(csv_offer[2])] = []  
for csv_offer in f2:  
    if (int(csv_offer[0]) == day):  
        if (int(csv_offer[1]) == scen):  
            y_d[int(csv_offer[2])].append(round(float(csv_offer[5]),3))  
            y_r[int(csv_offer[2])].append(round(float(csv_offer[6]),3))  
            pi_d[int(csv_offer[2])].append(round(float(csv_offer[3]),3))  
            pi_r[int(csv_offer[2])].append(round(float(csv_offer[4]),3))  
address = csv_offer[7]  
return (y_d,y_r,pi_d,pi_r,address)
for $j$ in $n_d$:
    if ($i\neq 0$):
        $\gamma_P[i][j] = m.addVar(vtype=GRB.BINARY,name='gammaP'+str(i)+'_'+str(j))$
        $\gamma_Q[i][j] = m.addVar(vtype=GRB.BINARY,name='gammaQ'+str(i)+'_'+str(j))$
    $P_a[num_tranches+2] = m.addVar(vtype=GRB.BINARY,name='P_a'+str(i))$
    m.update()
# Tranche Constraints
for $j$ in $n_d$:
    m.addConstr(quicksum($\gamma_P[i][j]$ for $i$ in range(1,num_tranches+1))
                + quicksum($\gamma_Q[i][j]$ for $i$ in range(0,num_tranches+1)) == 1)
    temp=[]
    for $i$ in range(0,num_tranches+1):
        temp.append($\gamma_Q[i][j]$)
    for $i$ in range(1,num_tranches+1):
        temp.append($\gamma_P[i][j]$)
    m.addSOS(GRB.SOS_TYPE1, temp)
for $i$ in range(0,num_tranches+1):
    for $j$ in $n_d$:
        m.addConstr($p[j] \geq P_a[i] - 1000*(1-\gamma_Q[i][j])$
                    + $P_a[i+1] + 1000*(1-\gamma_Q[i][j])$
                    + $Q_a[i] - 800*(1-\gamma_Q[i][j])$
                    + $Q_a[i] + 800*(1-\gamma_Q[i][j])$
        for $i$ in range(1,num_tranches):
            m.addConstr($P_a[i] \leq P_a[i+1]$
                        + $Q_a[i] \leq Q_a[i+1]$
                        + $Q_a[0] = 0$
        m.update()

def d_Tranche_Monoronicity(m, tranche_no,y,pi,n_d):
    # Tranche constraints formulated as an assignment problem to limit the number of tranches
    Qa={}  
    Pa={}   
    gammaP={} 
    gammaQ={} 
    Qa[0] = 0 
    Pa[0] = 2000 
    Pa[tranche_no + 1]=0 
    for $i$ in range(1,tranche_no + 1):
        Qa[i] = m.addVar(name='Demand_Level_%s' % (i))
        Pa[i] = m.addVar(name='Price_Level_%s' % (i))
        gammaP[i]={}
        gammaQ[i]={}
    for $j$ in $n_d$:
        gammaP[i][j] = m.addVar(vtype=GRB.BINARY,name='gammaP'+str(i))
        gammaQ[i][j] = m.addVar(vtype=GRB.BINARY,name='gammaQ'+str(i))
    m.update()
# Tranche Constraints
for $j$ in $n_d$:
    m.addConstr(quicksum(gammaP[i][j] for $i$ in range(1,tranche_no+1))
                + quicksum(gammaQ[i][j] for $i$ in range(1,tranche_no+1)) == 1)
    temp=[]
    for $i$ in range(1,tranche_no+1):
        temp.append(gammaQ[i][j])
    for $i$ in range(1,tranche_no+1):
        temp.append(gammaP[i][j])
    m.addSOS(GRB.SOS_TYPE1, temp)
for $i$ in range(1,tranche_no):
    for $j$ in $n_d$:
        m.addConstr($p[j] \geq P_a[i] - max_a_price*(1-gammaQ[i][j])$
                    + $P_a[i] + max_a_price*(1-gammaQ[i][j])$
                    + $Q_a[i-1] - 800*(1-gammaP[i][j])$
                    + $Q_a[i] + 800*(1-gammaP[i][j])$
        for $i$ in range(1,tranche_no):
            m.addConstr($p[j] \geq P_a[i] - max_a_price*(1-gammaP[i][j])$
                        + $P_a[i] + max_a_price*(1-gammaP[i][j])$
                        + $Q_a[i] - 800*(1-gammaP[i][j])$
                        + $Q_a[i] + 800*(1-gammaP[i][j])$
m.addConstr(y[j]==Q[i]==800*(1-gammaP[i][j]));
for i in range(0, tranche_no):
    m.addConstr(Pa[i]==Pa[i+1]);
    m.addConstr(Qa[i]==Qa[i+1]);
m.update()
def Monotonicity(m, y_d, y_r, pi_d, pi_r, count):
d_zed = {}  
r_zed = {}  
for i in range(count):
    for j in range(count):
        if i != j:
            d_zed[i,j] = m.addVar(vtype = GRB.BINARY, name = "d_Monotonicity_%s_%s" % (i,j))
            r_zed[i,j] = m.addVar(vtype = GRB.BINARY, name = "r_Monotonicity_%s_%s" % (i,j))
m.update()  
# demand monotonicity  
for i in range(count):
    for j in range(count):
        if i != j:
            m.addConstr(y_d[i] <= y_d[j] + 800*d_zed[i,j], name = "d_Monotonicity_1_%s_%s" %(i,j))
            m.addConstr(pi_d[j] <= pi_d[i]+ max_e_price*d_zed[i,j], name = "d_Monotonicity_2_%s_%s" %(i,j))
for i in range(count):
    for j in range(count):
        if i != j:
            m.addConstr(d_zed[i,j] + d_zed[j,i] == 1, name = "d_Monotonicity_3_%s_%s" %(i,j))
# reserve monotonicity  
for i in range(count):
    for j in range(count):
        if i != j:
            m.addConstr(y_r[i] <= y_r[j] + 800*r_zed[i,j], name = "r_Monotonicity_1_%s_%s" %(i,j))
            m.addConstr(pi_r[i] <= pi_r[j]+ max_e_price*r_zed[i,j], name = "r_Monotonicity_2_%s_%s" %(i,j))
for i in range(count):
    for j in range(count):
        if i != j:
            m.addConstr(r_zed[i,j] + r_zed[j,i] == 1, name = "r_Monotonicity_3_%s_%s" %(i,j))
m.update()
return(d_zed, r_zed)
def solve(u, n_simulation, scens, days, season, tp_type):
m = Model("Monotones")
tranche_no = 4
m_gap = 0.004
utility = u
y_d = {}  
y_r = {}  
pi_d = {}  
pi_r = {}  
y_d_var = {}  
y_r_var = {}  
pi_d_var = {}  
pi_r_var = {}  
ypi_d_var = {}  
ypi_r_var = {}  
y_d_list = []  
y_r_list = []  
pi_d_list = []  
pi_r_list = []  
count = 0
sum_y_r = 0
sum_y_d = 0
sum_pi_d = 0
sum_pi_r = 0
ave_y_r = 0
ave_y_d = 0
ave_pi_d = 0
ave_pi_r = 0
med_y_r = 0
med_y_d = 0
med_pi_d = 0
med_pi_r = 0
downright = []
upleft = []
downleft = []
upright = []
d_zed = {}
r_zed = {}
mon_d ={}
mon_r = {}
max_profit = 0
max_ilr = 0
max_profit_price = 0
max_ilr_price = 0
day_address ={}
for d in days:
    for s in scens:
        (y_d[count],y_r[count],pi_d[count],pi_r[count],address) = Get_regions_data(d, s, 'regions_%s.csv' %(d) )
        (y_d_var[count], y_r_var[count],pi_d_var[count],pi_r_var[count]), = Define_ypi(m,y_d[count],y_r[count],pi_d[count],pi_r[count],count)
        day_address[d]= address
        count = count + 1
m.setObjective(sum((utility*y_d_var[c] - ypi_d_var[c] + ypi_r_var[c])
        for c in range(count)), GRB.MAXIMIZE)
m.setParam('MIPGap', 0.00000001)
m.update()
m.optimize()
if m.status == GRB.status.OPTIMAL:
    with open('monotonicity_solution_monon_%s_%s_%s_%s.csv' %(
        season, tp_type_string,u,n_simulation), 'ab') as fp0:
        all0 = csv.writer(fp0)
        all0.writerow([days,scens])
        for c in range(count):
            all0.writerow([str(c),str(y_d_var[c].getAttr('x') ),str(y_r_var[c].getAttr('x') ),
            str(pi_d_var[c].getAttr('x') ),str(pi_r_var[c].getAttr('x') ),
            str(utility*y_d_var[c].getAttr('x') - ypi_d_var[c].getAttr('x')
            + ypi_r_var[c].getAttr('x'))])
y_d_list.append(y_d_var[c].getAttr('x'))
sum_y_d = sum_y_d + y_d_var[c] .getAttr('x')
y_r_list.append(y_r_var[c] .getAttr('x'))
sum_y_r = sum_y_r + y_r_var[c] .getAttr('x')
pi_d_list.append(pi_d_var[c] .getAttr('x'))
sum_pi_d = sum_pi_d + pi_d_var[c] .getAttr('x')
pi_r_list.append(pi_r_var[c] .getAttr('x'))
sum_pi_r = sum_pi_r + pi_r_var[c] .getAttr('x')
all0.writerow([str(sum((utility*y_d_var[c].getAttr('x') - ypi_d_var[c].getAttr('x')
            + ypi_r_var[c].getAttr('x'))) for c in range(count))/count])
ave_y_d = sum_y_d/count
ave_y_r = sum_y_r/count
ave_pi_d = sum_pi_d/count
ave_pi_r = sum_pi_r/count
med_y_d = median(y_d_list)
med_y_r = median(y_r_list)
med_pi_d = median(pi_d_list)
med_pi_r = median(pi_r_list)
for c in range(count):
    if (y_d_var[c] .getAttr('x')*(90 - pi_d_var[c] .getAttr('x') ) > max_profit):
        max_profit = y_d_var[c] .getAttr('x')*(90 - pi_d_var[c] .getAttr('x') )
        max_profit_price = pi_d_var[c] .getAttr('x')
    if (y_r_var[c] .getAttr('x')*pi_r_var[c] .getAttr('x') > max_ilr):
        max_ilr = (y_r_var[c] .getAttr('x')*pi_r_var[c] .getAttr('x')
        max_ilr_price = pi_r_var[c] .getAttr('x')
m1 = Model('Monontone')
for c in range(count):
    (y_d_var[c], y_r_var[c],pi_d_var[c],pi_r_var[c],ypi_d_var[c],ypi_r_var[c]) = Define_ypi(m1,y_d[count],y_r[count],pi_d[count],pi_r[count],c)
m1.update()
    (d_zed,r_zed)= Monotonicity(m1,y_d_var,y_r_var,pi_d_var,pi_r_var,count)
m1.setObjective(sum((utility*y_d_var[c] - ypi_d_var[c] + ypi_r_var[c])
    for c in range(count)), GRB.MAXIMIZE)
m1.setParam('MIPGap', m_gap)
m1.setParam('TimeLimit', 5000)
m1.update()
m1.optimize()
if m1.status == GRB.Status.OPTIMAL or m1.status == GRB.Status.TIME_LIMIT:
    m.write(output_file)
for c in range(count):
yd_list.append(round(y_d_var[c].getAttr('x'),2))
yr_list.append(round(y_r_var[c].getAttr('x'),2))
pid_list.append(round(pi_d_var[c].getAttr('x'),2))
pir_list.append(round(pi_r_var[c].getAttr('x'),2))
d_sort = zip(yd_list, d_count_list)
list.sort(d_sort)
(yd_list, d_count_list) = zip(*d_sort)
for c in d_count_list:
yd_list.append(yd_list[c])
pid_list.append(pid_list[c])
for c in range(count-1):
    if pid_list[c] - pid_list[c+1] < 0.01:
        if c not in d_remove_list:
            d_remove_list.append(c)
            d_count = d_count - 1
for c in range(count):
temp_dy.append(yd_list[c])
temp_dpi.append(pid_list[c])
for c in d_remove_list:
yd_list.remove(temp_dy[c])
pid_list.remove(temp_dpi[c])
r_sort = zip(yr_list, r_count_list)
list.sort(r_sort)
(yr_list, r_count_list) = zip(*r_sort)
for c in r_count_list:
yr_list.append(yr_list[c])
pir_list.append(pir_list[c])
for c in range(count-1):
    if pir_list[c] - pir_list[c+1] < 0.01:
        if c not in r_remove_list:
            r_remove_list.append(c)
            r_count = r_count - 1
for c in range(count):
temp_ry.append(yr_list[c])
temp_rpi.append(pir_list[c])
for c in r_remove_list:
yr_list.remove(temp_ry[c])
pir_list.remove(temp_rpi[c])
y_r_tranches = []
pi_d_list_2 = []
pi_r_list_2 = []
y_d_tranches = []
y_r_tranches = []
pi_d_lst2 = []
pi_r_lst2 = []
addresses = []

#For energy
for c in range(d_count):
    if last_c_d:
        y_d_tranches.append(yd_list[c])
        pi_d_list_2.append(pid_list[c])
        last_c_d = 0
    else:
        y_d_tranches.append(yd_list[c] - yd_list[c - 1])
        pi_d_list_2.append(pid_list[c])
for c in range(d_count):
    if y_d_tranches[c] != 0:
        y_d_tranches.append(y_d_tranches[c])
        pi_d_lst2.append(pi_d_list_2[c])
#For reserve
for c in range(r_count):
    if first_c_r:
        y_r_tranches.append(yr_list[c])
        pi_r_list_2.append(pir_list[c])
        first_c_r = 0
    else:
        y_r_tranches.append(yr_list[c] - yr_list[c - 1])
        pi_r_list_2.append(pir_list[c])
for c in range(r_count):
    if y_r_tranches[c] != 0:
        y_r_tranches.append(y_r_tranches[c])
        pi_r_lst2.append(pi_r_list_2[c])
for d in days:
    addresses.append(day_address[d])
with open('monotonicity_solution_mon_%s_%s_%s_%s.csv' % (season, tp_type, u, n_simulation), 'ab') as fp1:
    all1 = csv.writer(fp1)
    all1.writerow([days, scens, m_gap])
    for c in range(count):
        all1.writerow([str(c), str(y_d_var[c].getAttr('x')), str(y_r_var[c].getAttr('x'))
                        , str(pi_d_var[c].getAttr('x')), str(pi_r_var[c].getAttr('x'))
                        , str(utility * y_d_var[c].getAttr('x') - ypi_d_var[c].getAttr('x')
                        + ypi_r_var[c].getAttr('x'))])
with open('monotone_d_stacks_%s_%s_%s_%s.csv' % (season, tp_type, u, n_simulation), 'ab') as fp10:
    all0 = csv.writer(fp10)
    for c in range(d_count):
        all0.writerow([str(y_d_tranches[c]), str(pi_d_list_2[c])])
with open('monotone_r_stacks_%s_%s_%s_%s.csv' % (season, tp_type, u, n_simulation), 'ab') as fp2:
    all2 = csv.writer(fp2)
    for c in range(r_count):
        all2.writerow([str(y_r_tranches[c]), str(pir_list[c])])

def main():
    (tp_type, season, p_scens_pool, days) = Get_TP_Data("TP.csv", "Days.csv")
    ut_values = Get_u_Values("utility_values.csv")
    for i in range(200):
        for ut in ut_values:
            solve(ut, i, p_scens[i], days[i], season[i], tp_type[i])
if __name__ == "__main__":
    main()
Bibliography


[121] Gurobi Optimization, “Gurobi.”

[122] IBM, “Cplex.”


