The Dissertation Committee for Alexander Joseph Zolan
certifies that this is the approved version of the following dissertation:

Decomposition and Variance Reduction Techniques for Stochastic Mixed Integer Programs

Committee:

John Hasenbein, Supervisor

David Morton, Co-Supervisor

Jonathan Bard

Grani Hanasusanto

Alexandra Newman

Decomposition and Variance Reduction Techniques for Stochastic Mixed Integer Programs

by

Alexander Joseph Zolan

DISSERTATION

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY

THE UNIVERSITY OF TEXAS AT AUSTIN

December 2018

Dedicated to Eugene

Acknowledgments

I am deeply grateful for the large community of people who have helped me throughout my time at the University of Texas at Austin. I am indebted to my parents, Robert and Barbara Zolan, for instilling in me an appreciation of learning and effort, and to my brothers, Jesse and Matt Zolan, for their encouragement, and for setting good examples for returns on hard work. I thank the Operations Research and Industrial Engineering faculty at the University of Texas at Austin for teaching me the foundations of the research contained in this work, and for offering me the opportunity to teach other while I was here; special thanks go to my committee members, Dr. Grani Hanasusanto and Dr. Jonathan Bard, for their targeted feedback and support. Thanks to the other students in the ORIE program for their camaraderie and contribution to a great working environment. In particular, I want to thank Areesh Mittal for his review of drafts of this work, Josh Woodruff for his collaboration on a collection of projects both in and out of the classroom, and Murat Karatas for his friendship and perspective. I am grateful for the many collaborators with whom I worked closely on my assorted research projects during my time as a graduate student, listed in alphabetical order: Dr. Stephen Frank, Dr. Gavin Goodall, Dr. Chris Hadlock, Dr. Michael Helwig, Dr. Amanda Hering, Mark Husted, Ernie Kee, Luigi Gentile Polese, Dr. Bharat Suthar, Dr. Jeremy Tejada, and Dr. Michael Wagner. In particular, I want to thank Dr. Michael Scioletti for his friendship, support, and diligence as a collaborator. Finally, I owe thanks to those who have made outsize

contributions to the work in this document: to Dr. Alexandra Newman, for spending an excessive amount of her time connecting me to research and career opportunities, and for teaching me the foundations of scientific writing; to my supervisors, Dr. John Hasenbein and Dr. David Morton, for their patience, knowledge, direction, research opportunities, lessons in writing, and advice on a large collection of topics; and, to my wife, Erin Wedepohl, for her unwavering support, and for bearing the emotional and financial burdens that come with having a spouse in graduate school for the better part of a decade.

Dr. Scioletti developed the prototype GAMS model from Scioletti et al. (2017), and implemented the various linearization formulations from Gounaris et al. (2009) to select the formulation we use for the baseline partitioning scheme in Section 2.3. Alex Zolan developed the improved model in Section 2.3, decomposition procedure in Section 2.4, and the CPLEX/Python implementation of all models used to generate the results in Section 2.5, under the supervision of Dr. Morton and Dr. Newman.

Dr. Hering developed the methodology for the base-wide occupancy model in Section 3.2.2, with support from Dr. Nate Putnam and Dr. Scioletti. Dr. Scioletti developed the conditional probabilities in Section 3.2.2.3. Alex Zolan developed the broader framework in Section 3.2.4, the scripts to pull and edit TMY weather files and use them as input to PVWatts, the Ruby measures in OpenStudio to obtain building-level load realizations, and the implementation of the models in Section 3.3 in the Python/CPLEX API, under the supervision of Dr. Morton.

The contributions in Chapter 4 were developed by Alex Zolan, under the supervision of Dr. Hasenbein and Dr. Morton.

Decomposition and Variance Reduction Techniques for Stochastic Mixed Integer Programs

Publication No. _____

Alexander Joseph Zolan, Ph.D.
The University of Texas at Austin, 2018

Supervisor: John Hasenbein
Co-Supervisor: David Morton

Obtaining upper and lower bounds on the optimal value of a stochastic integer program can require solution of multiple-scenario problems, which are computationally expensive or intractable using off-the-shelf integer-programming software. Additionally, optimal solutions to a two-stage problem whose second stage spans long time horizons may be optimistic, due to the model's inappropriate ability to plan for future periods which are not known in practice. To that end, we present a framework for optimizing system design in the face of a restricted class of policies governing system operation, which aim to model realistic operation. This leads to a natural decomposition of the problem yielding upper and lower bounds which we can compute quickly. We illustrate these ideas using a model that seeks to design and operate a microgrid to support a forward operating base. Here, designing the microgrid includes specifying the number and type of diesel generators, PV systems, and batteries while operating the grid involves dispatching these assets to satisfy load at minimum

cost. We extend our approach to solve the same problem under load and photovoltaic uncertainty, and propose a method to generate appropriately correlated scenarios by simulating building occupancy via a bottom-up approach, then using the occupancy levels to inform environmental control unit loads on the base. Finally, in a separate line of work, we optimize the design of the strata for a stratified sampling estimator to reduce variance. We extend this method to the multivariate setting by optimizing the strata for a nonuniform Latin hypercube estimator. We then present empirical results that show that our method reduces the variance of the estimator, compared to one using equal-probability strata.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

We consider the design of a microgrid for a forward operating base (FOB) in which the design decision specifies the number and type of diesel generators, photovoltaic (PV) systems, and batteries with the goal of reducing the total cost of energy. A FOB is a military outpost that operates independently in a remote location. FOBs generally require significant logistical support, which often involves moving supplies by armored convoy through unprotected supply lines. Diesel generators are currently the primary source of providing electric power to the FOB. The larger the required resupply of diesel fuel via logistics convoys, the larger the associated security risk. The model we consider seeks to reduce this risk by reducing the volume of fuel through the use of PV systems and batteries in the microgrid design.

Figure 1.1 shows an example of the current method of powering a FOB. This is typically called "spot generation," in which a few disconnected diesel generators each power a few small loads. The diesel generators on FOBs are usually oversized for the peak load, and this peak load is relatively infrequent, which leads to the generator running well under capacity most of the time. Because generators are built to operate at or near full capacity, the system runs inefficiently, which leads to unnecessarily high fuel costs and potential maintenance issues.

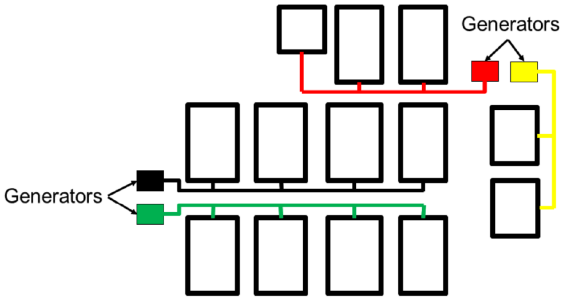

Figure 1.1: Example of current FOB power distribution under spot generation. Each generator is sized to exceed the peak load for the subset of the FOB it serves.

We consider the redesign of the electrical infrastructure of a FOB to match that of a microgrid, an example of which is shown in Figure 1.2. A microgrid is a modern, small-scale version of the centralized electricity system that is built for specific local goals, such as reliability, carbon emission reduction, diversification of energy sources, and/or cost reduction. Because the FOB is sufficiently small such that we can ignore power losses across distribution lines, we can treat all generation as occurring centrally and then being distributed to the loads throughout the FOB. A microgrid setup offers multiple advantages over spot generation, such as:

- a centralized power source allows some generators to be shut down during off-peak hours;
- the presence of PV systems and energy storage (e.g., batteries) can reduce the number of generators needed to meet load; and,
- energy storage can allow the generators to run at capacity anytime they are on, potentially increasing the efficiency of the system and avoiding maintenance issues due to wet-stacking.

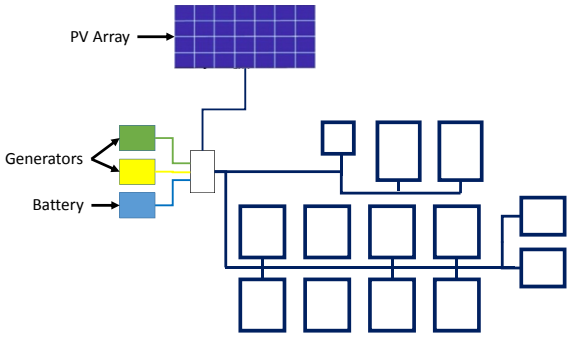

Figure 1.2: Example of microgrid FOB power distribution. Here, we have centralized generation which includes diesel generators, a PV array, and a battery. Generation capacity is sized to peak load across the entire FOB.

Scioletti et al. (2017) develop a mixed-integer nonlinear programm (MINLP) for establishing a design and dispatch strategy for a microgrid that supports a FOB, modeling the acquisition of different power technologies as integer variables and their operation using linear and nonlinear expressions. Specifically, the nonlinear relationships consist of a collection of bilinear terms which have no common components. The model aims to answer the following questions at minimum cost:

- How many, and what type, of diesel generators, PV power systems, and batteries should be purchased?
- For each timestep, how should these assets be operated to meet the load?

Scioletti et al. (2017) use convex and exact underestimators originally developed by McCormick (1976) to linearize the bilinear terms, resulting in a mixed integer linear program (MIP) that, unlike the MINLP, can solve realistic year-long instances with hourly fidelity to within 5% of optimality in a matter of hours. A second model developed by Scioletti et al. (2016a) uses a collection of convex sub-envelopes, exactly

one of which is selected for each bilinear term through the use of binary decision variables, which we term *subregion activation variables*. While these sub-envelopes reduce the error of the solution with respect to the original MINLP, the binary variables add a significant computational cost. Additionally, in practical situations, the load over the entire time horizon will likely not be known, which may cause dispatch solutions to the MIP model to be optimistic.

In Chapter 2, we make two novel contributions. First, we present an improvement of the fastest-performing linearization scheme from Scioletti et al. (2016a) for our application that allows for the model to be equivalent to the McCormick envelope when binary restrictions on the subregion activation variables are relaxed; this tighter formulation allows us to obtain approximately optimal solutions to the problem more quickly than with the partitioning scheme described by Scioletti et al., and sufficiently reduce the approximation error associated with the linearization enough to allow us to use a perturbation-based heuristic to obtain high-quality solutions to the MINLP. Second, we implement a policy that requires energy storage assets to revert to the same level of inventory (i.e., state-of-charge) at regular intervals; this subdivides the year-long operations problem into subproblems that we may solve in parallel, and reduces the timeframe for which we assume the load is known when solving the model. We present models that produce upper and lower bounds on the optimal value of the original problem, and we obtain nearly optimal solutions to a collection of instances in 6 or fewer minutes.

In Chapter 3, we leverage the decomposition technique to present a version of the microgrid optimization model that incorporates uncertainty in the photovoltaic

resources available and the FOB's hourly electricity demand. We present a bottom-up approach to simulating base-wide occupancy by starting with a predetermined schedule, then adding variability to mission times and including random special events to supplement normal FOB operations. We develop a collection of conditional probabilities that we use to simulate the location of each soldier on the base, which informs the hourly load for each building according to a physics-based building simulation model. The weather inputs used for the loads are common with those we use to generate hourly photovoltaic power output in a bivariate sample path; the weather inputs include historical observations for the location we use in our case study. We then solve an instance of the stochastic programming model using the sample average approximation technique developed by Mak et al. (1999), and compare the solution to our model under uncertainty to that obtained by using a point forecast as input.

The contributions in Chapter 4 compose a separate effort targeted toward the application of Latin hypercube sampling (LHS). LHS is commonly used to sample a multivariate function as an alternative to naïve Monte Carlo, because the LHS estimator is usually more efficient. Most applications of LHS use equal-probability strata. In this chapter, we present a model that optimizes the stratification of an LHS estimator. We start with the examples of a stratified sampling estimator and univariate LHS, and recast the nonconvex, nonlinear programming formulations as dynamic programs. In these example cases, we use an objective function that fully characterizes the variance of the LHS estimator. We also present the variance of an LHS estimator in the multivariate setting, which is computationally intractable as the number of components and/or strata grows large. We present two separate approxi-

mations of the objective functions; one allows us to use a dynamic program to obtain optimal strata, and the other is solved using a coordinate descent-based heuristic. Finally, we provide empirical results showing that our methods yield estimators that exhibit significant variance reduction compared to using equal-probability strata for a collection of sample functions.

Chapter 2

Decomposing Mixed-Integer Programs for Optimal Microgrid Design

2.1 Introduction and Literature Review

Spot generation is necessary for remote sites at which electric power must be produced without a connection to the grid, e.g., to administer disaster relief, to maintain quality of life on Native American reservations, to run mining operations, or to sustain combat operations. In many cases, each load is satisfied by an individual diesel generator, and requires a rated capacity slightly greater than the designed peak demand for that load source. As a result, total system efficiency is low in off-peak periods, and estimates of the fully burdened cost of fuel can reach $1,000/gallon when the fuel is transported by air and armored convoy (Erwin 2010). Microgrids that integrate renewable technologies, such as photovoltaic (PV) systems, energy storage technologies (e.g., batteries), and diesel generators, are emerging as a means to power remote sites. Like spot generation, microgrid technology can provide energy needs, but may include PV and battery modules to limit the use of diesel generators and increase efficiency. The fuel consumed by a remote site using these technologies, compared to that by a spot generation strategy, may reduce (i) the required flow of resupply fuel by ground convoys or helicopters, (ii) the environmental footprint, and (iii) wear and tear on the diesel generators, among other benefits.

The availability of portable generation technologies at a remote site limits the design to one or more diesel generators and batteries of various sizes, as well as a modular PV system technology in prespecified capacity increments. The costs of design and dispatch include (i) generator, PV system and battery procurement, (ii) diesel fuel consumption, and (iii) generator and battery "lifecycles," our measurement of the degradation associated with their use. The design decision specifies the number and types of generators, PV systems, and batteries in the microgrid. Dispatch decisions for each time period include which generators to turn on, the power output of all generators and PV systems, and power allocated to charging or discharging the batteries; for this small, non-market-driven environment, we do not make "commitment decisions" typical of large, grid-connected power plants.

In our model, decisions are subject to the following sets of constraints: (i) the design must meet load and spinning reserve requirements for each time period; (ii) power input and output from dispatched generators and batteries are limited by their minimum and maximum rated capacities; (iii) the fuel consumed is a known linear function of each generator's power output and state (on or off) in each time period; (iv) battery power input and output is the (bilinear) product of its voltage and the current into and out of the battery, respectively, where voltage is a function of the state-of-charge and direction of current; (v) a battery's state-of-charge is a function of that of the previous time period and the net current in the present time period, and is modeled as a percentage of total capacity; (vi) PV systems have a maximum power output per unit for each time period; and, (vii) spatial constraints limit the number of units in the design for each type of PV system. This paper presents a

model that seeks to design and dispatch such a microgrid to support a remote site at minimum cost, given known load and PV power output per array at hourly fidelity for an operating time horizon of one year. We refer to this model as the microgrid design and dispatch problem.

The literature includes work to assess microgrid operation policies in which batteries and photovoltaics are modeled as distributed energy resources (e.g., van der Kam and van Sark 2015, Zhang et al. 2016). However, we assume that the forward operating base is the size of a football field, obviating the need to enforce alternating current power flow requirements, nor do we account for line losses that would be present in a larger system. The selected set of generators, batteries and photovoltaics meets alternating current power demand, where (i) generators are connected directly to an alternating current bus; (ii) batteries are connected to the same bus via bi-directional converters that account for efficiency, and (iii) the photovoltaics are modeled using a PVWatts calculator (Dobos 2013) that includes seasonal variations and assumes a direct current-to-alternating current conversion and maximum power point tracking. Holding spinning reserves mitigates the intermittent nature of the photovoltaics; our model requires that a fraction of the photovoltaic power must be covered, if necessary, through battery or generator operation. We treat the load as deterministic though our techniques extend to a stochastic environment as we point out throughout the paper.

Most microgrid design and dispatch models in the literature enforce nonlinear relationships but restrict operation to a fixed number of potential dispatch strategies, and simulate operations for one or more load scenarios to estimate the total cost.

Because enumerating all designs is intractable for realistic instances, these models are solved by heuristic search methods, which use, e.g., a simulation procedure to determine whether a design can meet power demand under prespecified dispatch rules. Simulation, or a rule-based heuristic, is used to dispatch assets to satisfy demand, in part because nonlinear relationships associated with modeling the incumbent technologies can render problem instances intractable. These approaches set operating rules, pre-qualify technology procurements, or use site-specific data to obtain a microgrid solution that attempts to meet demand at minimum cost.

Green and Manwell (1995) solve a design and dispatch problem with PV, wind, battery, and diesel generators as candidate technologies by using a time series model to generate scenarios and running a collection of fixed dispatch schemes for potential designs. Barley and Winn (1996) evaluate a set of operations strategies for PV-diesel-battery systems that cycle the batteries to different depths of discharge. The Hybrid Optimization Model for Electric Renewables, or HOMER (2015), enumerates a series of dispatch strategies for a user-directed collection of designs. Dufo-López and Bernal-Agustín (2005) use a genetic algorithm to design a PV-diesel system by implementing a collection of operations policies borrowed from HOMER (2015) and from Barley and Winn (1996). Katsigiannis and Georgilakis (2008) employ tabu search to size small, isolated hybrid power systems using cycle-charging dispatch, which considers combinations of hybrid system technologies. Bala and Siddiqui (2009) propose to design a PV-diesel hybrid system using genetic algorithms that find a dispatch strategy to minimize net present cost. Gupta et al. (2011) develop a dispatch algorithm to maintain a constant power level for diesel generators in a hybrid system,

to maximize the use of renewable technologies. While these approaches yield practical, low-cost designs, they lack proof of (near-) optimality.

Most exact solution methods for optimizing microgrid design and dispatch arise via a MINLP or MIP model, and are limited in time horizon or other model fidelity. Morais et al. (2010) present a MIP for the optimal design of a hybrid system grid, which considers fuel cells and wind power technologies and solves 24-hour instances with hourly fidelity. Huneke et al. (2012) and Barbier et al. (2014) present solutions to year-long instances with hourly fidelity; the former use linear programming to design a microgrid, and the latter optimize the design and dispatch strategy of a system with wind turbines, batteries, and generators via a MIP. Khodaei et al. (2015) present a MIP that optimizes the design and dispatch of a microgrid with generic dispatchable and non-dispatchable assets and energy storage, at hourly fidelity, for at least a year of operation, with linear operating constraints; the authors assume the microgrid is disconnected from the grid for less than one day per year. Huneke et al. (2012), Barbier et al. (2014) and Khodaei et al. (2015) use battery operating constraints similar to those of Barley and Winn (1996), which do not address the variable capacity and degradation of the battery.

Optimization models that include both design and dispatch decisions and integrate hybrid technologies such as batteries, diesel generators, and PV systems are often presented as MINLPs, which are typically intractable for large instances. Pruitt et al. (2013) develop a non-convex MINLP to describe the design and dispatch of a combined heat and power system using solid oxide fuel cells for commercial buildings for a time horizon of one year.

MIPs that employ linearization and approximation techniques to bound and solve the nonlinear problem can serve as tractable alternatives to MINLPs and yield solutions which approximate realistic dispatch. The model of Scioletti et al. (2017) seeks an optimal hybrid system design with diesel generators, PV systems, and batteries as candidate technologies, subject to (i) a set of nonlinear battery power constraints that hold for each time period, and (ii) lifecycle degradation as a function of the battery's state-of-charge and current for a given time period. This predecessor paper approximates these nonlinear (specifically, bilinear) relationships in a MIP formulation for the design and dispatch model, which they solve for a collection of year-long instances at hourly fidelity.

Our first contribution lies in reducing the error associated with linearizing nonlinear battery power and lifecycle degradation constraints by replacing a single convex envelope with a series of sub-envelopes, which bound the feasible region of the approximation variable for the bilinear term and its components. This technique also allows for the construction of a MINLP-feasible solution to form a high-quality upper bound. Our second contribution is a decomposition method that divides the dispatch time horizon into smaller time periods, which yield subproblems that we solve in parallel to obtain upper and lower bounds on the model's optimal objective function value. While we illustrate the method using the microgrid design and dispatch problem from Scioletti et al. (2017), the approach extends to models that have a strategic decision that impacts operational decisions over a long time horizon, including airlift scheduling (Baker et al. 2002) and capital budgeting (Brown et al. 2004).

Section 2.2 describes the general model. Section 2.3 explains the approximation of bilinear terms associated with the product of battery state-of-charge and current. Section 2.4 details the decomposition method used to obtain upper and lower bounds on the model's optimal value. Section 2.5 assesses the performance of the proposed approximations and decomposition algorithm using a collection of instances from the literature. Section 2.6 concludes and presents possible extensions of this work.

2.2 Model Description

Our microgrid design and dispatch problem possesses the following properties: (i) the goal is to minimize the cost associated with both a time-invariant strategic decision and a series of time-varying operational (dispatch) decisions; (ii) some operational decisions include relationships modeled with bilinear terms; (iii) the model is loosely coupled with respect to time in that only a few constraints link consecutive time periods via inventory; and, (iv) after specified time intervals, the inventory (state-of-charge) variables must "reset" to the same value. In what follows, we use $t \in \mathcal{T}$ to index time periods, and the system must reset the inventory variables every υ time periods, where we assume $|\mathcal{T}|/\upsilon$ is an integer.

2.2.1 ($\mathcal{P}$) Formulation

We formulate our microgrid design and dispatch model using constructs that highlight temporal dependencies and yet are relatively general. This simplifies notation in our subsequent descriptions of both the linearization schemes for bilinear

terms and the decomposition methods with accompanying bounds.

Sets

$t \in \mathcal{T} = \{1, 2, \ldots, \lvert\mathcal{T}\rvert\}$	time periods
$\ell \in \mathcal{L} = \{1, 2, \ldots, \lvert\mathcal{L}\rvert\}$	time blocks indexing a partition of $\mathcal{T}$; i.e., $\cup_{\ell \in \mathcal{L}} \mathcal{T}_\ell = \mathcal{T}$ and $\mathcal{T}_\ell \cap \mathcal{T}_{\ell'} = \emptyset$, $\ell \neq \ell'$
$\mathcal{T}_\ell \subset \mathcal{T}$	time periods in block ℓ
$X \in \mathcal{X}$	strategic design decisions
$Y_t \in \mathcal{Y}_t(X)$	operational decisions made in time period t, given decision X

Functions

$f_0(\cdot)$	cost of a design decision
$f_t(\cdot)$	cost of an operational decision at time period t
$g_t(\cdot)$	net change in inventory associated with an operational decision at time period t

Parameters

υ	number of time periods per block

With these constructs, we have:

$$\mathcal{T}_\ell = \{(\ell - 1)\upsilon + 1, (\ell - 1)\upsilon + 2, \ldots, \ell\upsilon\}, \ \forall \ell \in \mathcal{L}.$$

Decision Variables

X	strategic design decision
R	strategic inventory reset value
Y_t	operational decision at time period t; $Y = (Y_t)_{t \in \mathcal{T}}$
$\underline{Y}_t$	inventory at start of time period t; $\underline{Y} = (\underline{Y}_t)_{t \in \mathcal{T}}$
$\bar{Y}_t$	inventory at end of time period t; $\bar{Y} = (\bar{Y}_t)_{t \in \mathcal{T}}$

Boundary Condition

Y_0	initial inventory

($\mathcal{P}$) Formulation

$$z^P = \min_{X,R,Y,\underline{Y},\bar{Y}} \quad f_0(X) + \sum_{t\in\mathcal{T}} f_t(Y_t) \tag{2.1a}$$

$$\text{s.t.} \quad X \in \mathcal{X} \tag{2.1b}$$

$$Y_t \in \mathcal{Y}_t(X), \quad \forall t \in \mathcal{T} \tag{2.1c}$$

$$\bar{Y}_t = \underline{Y}_t + g_t(Y_t), \quad \forall t \in \mathcal{T} \tag{2.1d}$$

$$\underline{Y}_t = \bar{Y}_{t-1}, \quad \forall t \in \mathcal{T} \setminus \{1\} \tag{2.1e}$$

$$\underline{Y}_{(\ell-1)v+1} = R, \quad \forall \ell \in \mathcal{L} \setminus \{1\} \tag{2.1f}$$

$$\underline{Y}_1 = Y_0 \tag{2.1g}$$

Through the objective in (2.1a) we seek a minimum-cost set of strategic and operational decisions. This objective includes the cost of battery degradation, which grows with deeper levels of discharge; therefore, we model lifecycles spent as a function of battery current and state-of-charge, which involves the product of the two. Constraint (2.1b) specifies feasible strategic designs, as defined by the set $\mathcal{X}$. Constraint (2.1c) restricts operational decisions Y_t to those allowable by the design decision X at each time period t; for example, only an asset purchased as part of a design decision may be operated at time period t. More generally, constraint (2.1c) includes nonlinear relationships as we detail in Section 2.2.2. Constraint (2.1d) captures changes in inventory from the start to the end of a given time period. Con-

straint (2.1e) reconciles the inventory between the end of one time period and the beginning of the next. Constraint (2.1f) enforces our reset policy by restricting the inventory at the boundaries of the time blocks to be the same. The reset policy limits the number of future time periods the model may use to inform dispatch decisions, and hence limits the model's ability to over-optimize dispatch to future variations in load. Constraint (2.1g) fixes the starting inventory to the boundary condition. The strategic decisions X and R, along with inventory constraints (2.1e), couple decisions across time. In Section 2.4, we develop a method that decomposes model (2.1) into $|\mathcal{L}|$ subproblems, in which the respective subproblems correspond to decisions within blocks $\mathcal{T}_\ell$, $\ell \in \mathcal{L}$.

2.2.2 ($\mathcal{M}$) Formulation: Linearization of ($\mathcal{P}$)

A subset of the relationships within constraint (2.1c), such as battery lifecycles spent as a function of current and state-of-charge, include bilinear terms. We can linearize these bilinear terms using ideas that begin with McCormick (1976) and include significant subsequent work such as Androulakis et al. (1995). Let Y_{1t} and Y_{2t} denote two components of operational decision Y_t that contribute a nonlinear term to constraint (2.1c), and let auxiliary decision variable Z_t represent the product $Y_{1t} \cdot Y_{2t}$. McCormick (1976) presents an approximation that replaces the bilinear relationship

$$Z_t = Y_{1t} \cdot Y_{2t}, \quad \forall t \in \mathcal{T}, \tag{2.2}$$

with a convex envelope to constrain Z_t, allowing for the reformulation of ($\mathcal{P}$) as an approximating MIP, i.e., as a linear mixed-integer program. The approximation used in McCormick (1976) follows.

Additional Decision Variables

Y_{1t}, Y_{2t}	two components of operational decision vector Y_t that form a bilinear relationship
Z_t	linear approximation variable representing the product $Y_{1t} \cdot Y_{2t}$

Additional Parameters

u_1	upper bound of operational component variable Y_{1t}
l_1	lower bound of operational component variable Y_{1t}
u_2	upper bound of operational component variable Y_{2t}
l_2	lower bound of operational component variable Y_{2t}

Formulation

$$Z_t \geq u_2 Y_{1t} + u_1 Y_{2t} - u_1 u_2, \quad \forall t \in \mathcal{T} \tag{2.3a}$$

$$Z_t \geq l_2 Y_{1t} + l_1 Y_{2t} - l_1 l_2, \quad \forall t \in \mathcal{T} \tag{2.3b}$$

$$Z_t \leq l_2 Y_{1t} + u_1 Y_{2t} - u_1 l_2, \quad \forall t \in \mathcal{T} \tag{2.3c}$$

$$Z_t \leq u_2 Y_{1t} + l_1 Y_{2t} - l_1 u_2, \quad \forall t \in \mathcal{T} \tag{2.3d}$$

Our application has time-invariant lower and upper bounds (l_1, u_1) and (l_2, u_2) for Y_{1t} and Y_{2t}, respectively, though this assumption may be relaxed in general. We refer to the formulation that starts with $(\mathcal{P})$ and replaces equation (2.2), which is a part of constraint (2.1c), with the relaxation described in constraints (2.3a)-(2.3d) as model $(\mathcal{M})$, or the *McCormick relaxation*.

2.2.3 Application to Microgrid Design and Dispatch Problem

Model ($\mathcal{M}$) specializes to the microgrid design and dispatch problem by mapping the strategic design vector X to the number and type of diesel generators, PV systems, and batteries in the design, and by mapping the operational decision vector Y_t to power input and output for each asset, and fuel consumption for each time period. Decision vectors $\underline{Y}$ and $\bar{Y}$ denote the available battery storage at the start and end of each time period, respectively. The variable R denotes the available battery storage at the beginning and end of each specified time block; this reset value is identical across all time blocks to simplify the dispatch policy and to limit appropriate adaptation of the dispatch policy to future load. In some applications, this target may be a prespecified level of inventory; however, in what follows we optimize this value. Because the load and PV availability exhibit diurnal patterns, we select the duration of a time block to be a day.

Our application assumes that a diesel generator consumes a fixed amount of fuel per time period when running, in addition to a variable amount that is a linear function of power output. Therefore, to maximize efficiency, it is preferable to run a diesel generator at its rated capacity. However, in response to rapid changes in load, one generator may run isochronous to the others. For this reason, in the MINLP in Scioletti et al. (2017), multiple copies of the same generators are modeled as "twins," each of which has an independent decision variable for power output at each time period. While the general problem may include twins of the same battery technology, the instances in Scioletti et al. (2017) restrict the solution to include at most one battery; this assumption is tantamount to honoring the policy in which batteries

operate in droop, rather than individually, to avoid a situation in which one battery is used to charge another. This high-level mapping suffices for our immediate purposes; however, the full microgrid design and dispatch model is detailed in Appendix A.1, largely following the model of Scioletti et al. (2017).

2.3 Reducing Linearization Error in ($\mathcal{M}$)

The envelope described in constraints (2.3a)-(2.3d) represents the tightest possible convex relaxation for a bilinear term (Al-Khayyal and Falk 1983); however, the relaxation in these models may neither yield sufficiently tight lower bounds on ($\mathcal{P}$)'s optimal value, nor produce implementable solutions to the nonlinear model. Further, constructing a feasible solution to ($\mathcal{P}$) by starting with a solution to ($\mathcal{M}$) may provide a low-quality upper bound on ($\mathcal{P}$). Bergamini et al. (2005) and Karuppiah and Grossman (2006) tighten this relaxation by subdividing the interval defined by the simple bounds of each component; we refer to this technique as *partitioning*. We introduce binary variables that determine which subregion defines the active constraints for Y_{1t}, Y_{2t}, and Z_t for each time period, and we refer to these as *subregion activation variables*. Figure 2.1 shows an example of such a subdivision when partitioning the domain of Y_{2t}. Both Wicaksono and Karimi (2008) and Gounaris et al. (2009) derive ways to partition McCormick's relaxation, noting that the computational performance of these formulations is impossible to predict; however, Gounaris et al. (2009) identify ten partitioning schemes that computationally outperform the other methods they present.

At a computational cost, partitioning reduces the approximation error, which

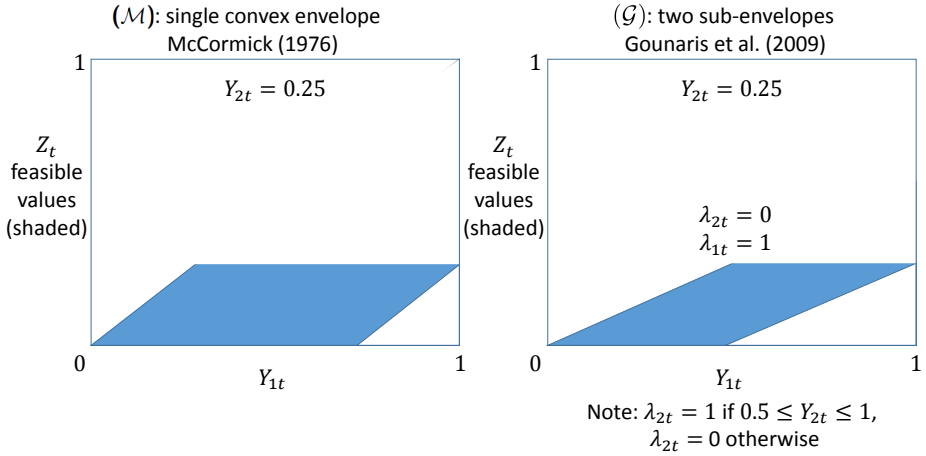

Figure 2.1: Example of subdividing the McCormick envelope given $Y_{2t} = 0.25$, shown on the left-hand side, by partitioning on one variable, in which we linearize $Y_{1t} \cdot Y_{2t}$. In this example, Y_{1t} and Y_{2t} have simple bounds $l_1 = l_2 = 0$ and $u_1 = u_2 = 1$. The shape on the right-hand side represents the sub-envelope that defines the feasible region for Y_{1t} and Z_t, with binary subregion activation variables; see subsequent constraints (2.8) and associated variables.

can improve lower bounds on, and allow for the construction of, a higher-quality solution to model ($\mathcal{P}$) when using the resulting linear MIP solution as a starting point. Partitioning requires binary restrictions to activate sub-envelopes, and introduces additional variables and constraints that grow linearly with the number of total subregions, both of which may compromise tractability. Vielma et al. (010a) and Vielma and Nemhauser (010b) present a model for piecewise-linear functions in which the number of binary variables scales logarithmically with the number of segments; Misener et al. (2011) use this model to implement a partitioning scheme that they then apply to the pooling problem, and show that the linear partitioning technique we adopt is preferable for situations in which the number of partitions is relatively low (eight or fewer subregions in their work).

If one component in the bilinear term can be discretized to a finite number of reference values in the place of a continuous domain, an exact linearization can be performed; Dvorkin et al. (2017) perform such a linearization, applied to a model for the use of energy storage for arbitrage in an electricity market. Similarly, Gupte et al. (2013) present an exact reformulation for terms that are the product of a nonnegative integer variable and a nonnegative continuous variable. The former is replaced by its binary expansion, and a McCormick envelope linearizes the resulting product. The authors develop the convex hull of the corresponding MIP set.

Castro (2015) presents a univariate partitioning scheme that specifies lower and upper bounds on both components of the bilinear term for each subregion, and applies the procedure to a subset of the cases in Gounaris et al. (2009). In Section 2.3.1, we reformulate a different case from Gounaris et al. (2009) that reduces to a single McCormick envelope when binary restrictions on the subregion activation variables are relaxed; our improvement applies to both univariate and bivariate partitioning schemes.

2.3.1 Partitioning Approach

For a given relaxation of $Z_t = Y_{1t} \cdot Y_{2t}$, we define the *approximation error* as a function of Y_{1t} as:

$$\mathcal{E}(\mathcal{Y}) = \max_{\mathcal{Y},\mathcal{Z}} \mathcal{Z} - \mathcal{Y} \cdot \mathcal{Y}, \tag{2.4}$$

in which the maximization over Y_{2t} and Z_t is constrained by the simple bounds on Y_{2t}, and the appropriate subregion depending on the approximation in use (e.g., constraints (2.3a)-(2.3d) if model ($\mathcal{M}$) is used). For the McCormick relaxation, $\mathcal{E}(\mathcal{Y})$

is maximized at the midpoint of the interval $[l_1, u_1]$, and this maximized value is one fourth of the area defined by the simple bounds on Y_{1t} and Y_{2t}, i.e., $(u_1 - l_1) \cdot (u_2 - l_2)/4$ (Androulakis et al. 1995).

Applying a partitioning scheme to the McCormick relaxation by creating m uniform subregions on the domain of one variable in the bilinear term (Bergamini et al. 2005, Karuppiah and Grossman 2006) decreases the worst-case approximation error by a factor of m to:

$$\frac{1}{m}\left(\frac{(u_1 - l_1)(u_2 - l_2)}{4}\right).$$

Hasan and Karimi (2010) present a bivariate partitioning scheme, which creates m uniform subregions on one variable and n uniform subregions on the other in the bilinear term. In this setting, the worst-case approximation error decreases by a factor of mn, to:

$$\frac{1}{mn}\left(\frac{(u_1 - l_1)(u_2 - l_2)}{4}\right).$$

Figure 2.2 depicts the maximum approximation error as a function of Y_{1t}, $\mathcal{E}(\mathcal{Y})$, when partitioning on one or both variables. Nonuniform partitions are possible, and even advisable, if partitioning is done dynamically in the course of solving the problem (Wicaksono and Karimi 2008). However, we restrict our attention to uniform partitions built a priori, which minimize the worst-case approximation error (Hasan and Karimi 2010). Dey and Gupte (2015) provide solution quality guarantees for applications of partitioning schemes to pooling problems, and show empirically that, for their application, a uniform partitioning scheme outperforms an asymmetric approach.

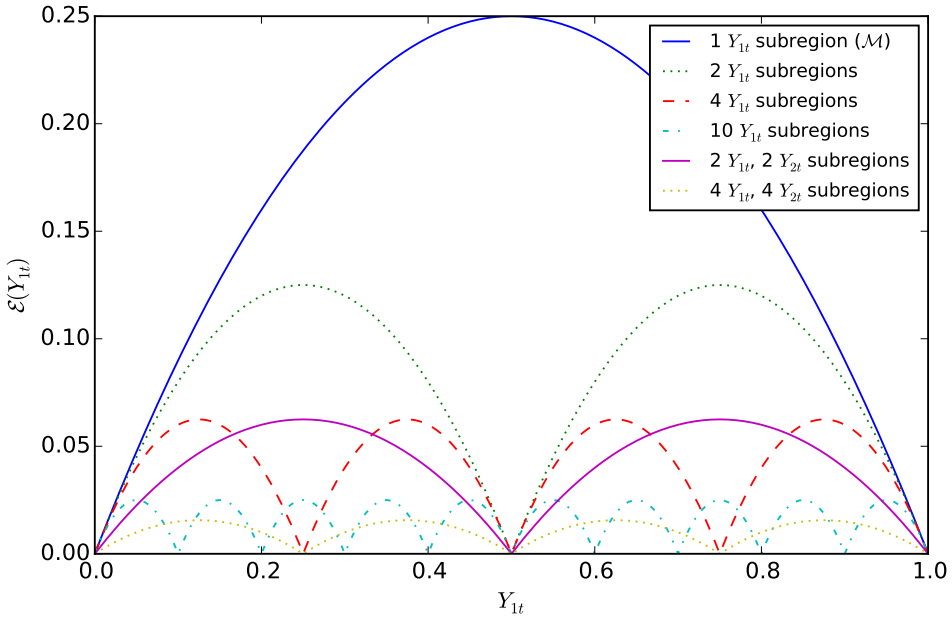

Figure 2.2: Approximation error per equation (2.4), as a function of Y_{1t} under partitioning schemes for different numbers of uniform subregions, using partitioning on Y_{1t} only, and on both Y_{1t} and Y_{2t}, respectively. This figure assumes that the range of both Y_{1t} and Y_{2t} is [0,1].

Section 2.3.2 presents the constraints we propose by partitioning on the variables Y_{1t} and Y_{2t}. We replace the McCormick relaxation in ($\mathcal{M}$) with new constraints from this approach to approximate the bilinear terms in ($\mathcal{P}$), and we refer to the resulting model as ($\mathcal{U}$) because it underestimates ($\mathcal{P}$).

2.3.2 ($\mathcal{U}$) Formulation

Our scheme partitions the feasible region into subregions according to intervals in the domain of Y_{1t}, which we index by $m \in \mathcal{M}$. So, we augment the simple bounds of u_1 and l_1 with boundaries for each active subregion u_{1m} and l_{1m}, $\forall m \in \mathcal{M}$, in which $l_{1,1} = l_1$, $u_{1,|\mathcal{M}|} = u_1$, and $l_{1,m} = u_{1,m-1}$, $\forall m \in \mathcal{M} \setminus \{1\}$; we perform an analogous

augmentation of the simple bounds of u_2 and l_2, which we index by $n \in \mathcal{N}$.

We introduce binary subregion activation variables to indicate which constraints define the active part of the feasible region for Y_{1t}, Y_{2t}, and Z_t when assigned a value of one; the constraints relax to nominal McCormick bounds otherwise.

Additional Sets

$m \in \mathcal{M} = \{1, 2, \ldots, \lvert\mathcal{M}\rvert\}$	set of subregions obtained by partitioning the domain of Y_{1t}
$n \in \mathcal{N} = \{1, 2, \ldots, \lvert\mathcal{N}\rvert\}$	set of subregions obtained by partitioning the domain of Y_{2t}

Additional Parameters

l_{1m}	lower bound for Y_{1t} within subregion m
u_{1m}	upper bound for Y_{1t} within subregion m
l_{2n}	lower bound for Y_{2t} within subregion n
u_{2n}	upper bound for Y_{2t} within subregion n

Additional Decision Variables

λ_{mnt}	1 if subregion (m, n) defines the active part of the feasible region of Y_{1t}, Y_{2t}, and Z_t in period t, 0 otherwise [binary]

Formulation

The following set of constraints represents partitioning on both Y_{1t} and Y_{2t}, which we tailor from formulation `NF2g`, the fastest-performing partitioning scheme of those described in Gounaris et al. (2009) when implemented for our application. Scioletti et al. (2016a) details the performance of alternative partitioning schemes from Gounaris

et al. when applied to the microgrid design and dispatch problem.

$$\sum_{m\in\mathcal{M}}\sum_{n\in\mathcal{N}}\lambda_{mnt}=1, \qquad \forall t\in\mathcal{T} \tag{2.5a}$$

$$Y_{1t}\geq l_1+(l_{1m}-l_1)\lambda_{mnt}, \qquad \forall m\in\mathcal{M}, n\in\mathcal{N}, t\in\mathcal{T} \tag{2.5b}$$

$$Y_{1t}\leq u_1-(u_1-u_{1m})\lambda_{mnt}, \qquad \forall m\in\mathcal{M}, n\in\mathcal{N}, t\in\mathcal{T} \tag{2.5c}$$

$$Y_{2t}\geq l_2+(l_{2n}-l_2)\lambda_{mnt}, \qquad \forall m\in\mathcal{M}, n\in\mathcal{N}, t\in\mathcal{T} \tag{2.5d}$$

$$Y_{2t}\leq u_2-(u_2-u_{2n})\lambda_{mnt}, \qquad \forall m\in\mathcal{M}, n\in\mathcal{N}, t\in\mathcal{T} \tag{2.5e}$$

$$\begin{aligned} Z_t \geq\ & u_{2n}Y_{1t}+u_{1m}Y_{2t}-u_{1m}u_{2n} \\ & -(u_1u_2-u_{1m}u_{2n}-(u_2-u_{2n})l_1-(u_1-u_{1m})l_2)(1-\lambda_{mnt}), \\ & \qquad \forall m\in\mathcal{M}, n\in\mathcal{N}, t\in\mathcal{T} \end{aligned} \tag{2.5f}$$

$$\begin{aligned} Z_t \geq\ & l_{2n}Y_{1t}+l_{1m}Y_{2t}-l_{1m}l_{2n} \\ & -(l_1l_2-l_{1m}l_{2n}-(l_2-l_{2n})u_1-(l_1-l_{1m})u_2)(1-\lambda_{mnt}), \\ & \qquad \forall m\in\mathcal{M}, n\in\mathcal{N}, t\in\mathcal{T} \end{aligned} \tag{2.5g}$$

$$\begin{aligned} Z_t \leq\ & l_{2n}Y_{1t}+u_{1m}Y_{2t}-u_{1m}l_{2n} \\ & +(u_{1m}l_{2n}-u_1l_2+(l_2-l_{2n})l_1+(u_1-u_{1m})u_2)(1-\lambda_{mnt}), \\ & \qquad \forall m\in\mathcal{M}, n\in\mathcal{N}, t\in\mathcal{T} \end{aligned} \tag{2.5h}$$

$$\begin{aligned} Z_t \leq\ & u_{2n}Y_{1t}+l_{1m}Y_{2t}-l_{1m}u_{2n} \\ & +(l_{1m}u_{2n}-l_1u_2+(u_2-u_{2n})u_1+(l_1-l_{1m})l_2)(1-\lambda_{mnt}), \\ & \qquad \forall m\in\mathcal{M}, n\in\mathcal{N}, t\in\mathcal{T} \end{aligned} \tag{2.5i}$$

$$\lambda_{mnt}\in\{0,1\}, \quad \forall m\in\mathcal{M}, n\in\mathcal{N}, t\in\mathcal{T}. \tag{2.5j}$$

Constraint (2.5a) requires exactly one of the subregion activation variables λ_{mnt} to

assume a value of one for each time period t. Constraints (2.5b)-(2.5e) restrict Y_{1t} and Y_{2t} to each variable's active subregion. If λ_{mnt}=1, then subregion (m, n) is active, and Y_{1t} and Y_{2t} are restricted to the intervals $[l_{1m}, u_{1m}]$ and $[l_{2n}, u_{2n}]$, respectively. If λ_{mnt}=0, the constraints reduce to simple bounds on Y_{1t} and Y_{2t}.

The McCormick relaxation uses four constraints to restrict Z_t as a function of the components Y_{1t} and Y_{2t} and their simple bounds. The partitioning scheme reduces the size of the active subregion, which allows for a smaller convex envelope, i.e., sub-envelope, which, in turn, yields a tighter relaxation. Constraints (2.5f) through (2.5i) bound Z_t according to the sub-envelope chosen by the subregion activation variables, λ_{mnt}. If λ_{mnt}=1, then Z_t is constrained by the convex envelope of the active subregion. If a subregion is not active, the associated constraints are dominated by McCormick bounds.

As we show in Proposition 2.3.1, if $\lambda_{mnt} = 1$, then the partitioning scheme of Gounaris et al. (2009) generates constraints (2.5), except that the last term in their analog of constraints (2.5f)-(2.5i) is $(u_1 - l_1)(u_2 - l_2)(1 - \lambda_{mnt})$ whose coefficient is the smallest value that reduces constraints (2.5f)-(2.5i) to simple bounds on Y_{1t} and Y_{2t} when $\lambda_{mnt} = 0$. Our approach differs from Gounaris et al. in that we find the smallest such value for each individual constraint, rather than a single value to apply to all constraints. When we use the approach in Gounaris et al. (2009) rather than constraints (2.5) to approximate the bilinear terms in ($\mathcal{P}$), we call the resulting model ($\mathcal{G}$). Proposition 2.3.1 characterizes the relative tightness of formulations ($\mathcal{U}$), ($\mathcal{G}$), and ($\mathcal{M}$). Our formulation is tighter than ($\mathcal{G}$) because when $\lambda_{mnt} = 0$, our constraints (2.5) revert to McCormick bounds, while the analogous constraints in ($\mathcal{G}$)

relax to simple bounds. To simplify the notation, we drop the t index, and treat Y_1, Y_2, and Z as scalar decision variables in the proposition.

Proposition 2.3.1. *Let* $m \in \mathcal{M} = \{1, 2, \ldots, |\mathcal{M}|\}$, *and* l_{1m}, u_{1m} *satisfy* $l_{1,1} = l_1$, $u_{1,|\mathcal{M}|} = u_1$, *and* $l_{1m} = u_{1,m-1}$, $l_{1,m-1} < l_{1m}$, $u_{1,m-1} < u_{1m}$, $\forall m \in \mathcal{M} \setminus \{1\}$. *Likewise, let* $n \in \mathcal{N} = \{1, 2, \ldots, |\mathcal{N}|\}$, *and* l_{2n}, u_{2n} *satisfy* $l_{2,1} = l_2$, $u_{2,|\mathcal{N}|} = u_2$, *and* $l_{2,n} = u_{2,n-1}$, $l_{2,n-1} < l_{2n}$, $u_{2,n-1} < u_{2n}$, $\forall n \in \mathcal{N} \setminus \{1\}$. *Let*

$$S = \{(Y_1, Y_2, Z) : Z = Y_1 \cdot Y_2,\ l_1 \leq Y_1 \leq u_1,\ l_2 \leq Y_2 \leq u_2\},$$

and

$$\begin{aligned} S^M = \{(Y_1, Y_2, Z) :\ & Z \geq u_2 Y_1 + u_1 Y_2 - u_1 u_2, \\ & Z \geq l_2 Y_1 + l_1 Y_2 - l_1 l_2, \\ & Z \leq l_2 Y_1 + u_1 Y_2 - u_1 l_2, \\ & Z \leq u_2 Y_1 + l_1 Y_2 - l_1 u_2, \\ & l_1 \leq Y_1 \leq u_1,\ l_2 \leq Y_2 \leq u_2\}. \end{aligned}$$

Define

$$Y_1 \geq l_1 + (l_{1m} - l_1)\lambda_{mn},\ \forall m \in \mathcal{M}, n \in \mathcal{N} \tag{2.6a}$$

$$Y_1 \leq u_1 - (u_1 - u_{1m})\lambda_{mn},\ \forall m \in \mathcal{M}, n \in \mathcal{N} \tag{2.6b}$$

$$Y_2 \geq l_2 + (l_{2n} - l_2)\lambda_{mn},\ \forall m \in \mathcal{M}, n \in \mathcal{N} \tag{2.6c}$$

$$Y_2 \leq u_2 - (u_2 - u_{2n})\lambda_{mn},\ \forall m \in \mathcal{M}, n \in \mathcal{N} \tag{2.6d}$$

$$Z \geq u_{2n} Y_1 + u_{1m} Y_2 - u_{1m} u_{2n} - a_{mn}(1 - \lambda_{mn}),\ \forall m \in \mathcal{M}, \forall n \in \mathcal{N} \tag{2.6e}$$

$$Z \geq l_{2n}Y_1 + l_{1m}Y_2 - l_{1m}l_{2n} - b_{mn}(1-\lambda_{mn}),\ \forall m \in \mathcal{M}, \forall n \in \mathcal{N} \tag{2.6f}$$

$$Z \leq l_{2n}Y_1 + u_{1m}Y_2 - u_{1m}l_{2n} + c_{mn}(1-\lambda_{mn}),\ \forall m \in \mathcal{M}, \forall n \in \mathcal{N} \tag{2.6g}$$

$$Z \leq u_{2n}Y_1 + l_{1m}Y_2 - l_{1m}u_{2n} + d_{mn}(1-\lambda_{mn}),\ \forall m \in \mathcal{M}, \forall n \in \mathcal{N}. \tag{2.6h}$$

Let

$$\begin{aligned} S^G = \{(Y_1, Y_2, Z) : (2.6)\ \textit{with}\ & a_{mn} = b_{mn} = c_{mn} = d_{mn} = (u_1 - l_1)(u_2 - l_2), \\ & l_1 \leq Y_1 \leq u_1,\ l_2 \leq Y_2 \leq u_2, \\ & \sum_{m\in\mathcal{M}} \sum_{n\in\mathcal{N}} \lambda_{mn} = 1,\ \lambda_{mn} \in \{0,1\},\ \forall m \in \mathcal{M}, n \in \mathcal{N}\} \end{aligned}$$

and

$$\begin{aligned} S^U = \{(Y_1, Y_2, Z) : (2.6)\ \textit{with}\ & a_{mn} = u_1u_2 - u_{1m}u_{2n} - (u_2 - u_{2n})l_1 - (u_1 - u_{1m})l_2, \\ & b_{mn} = l_1l_2 - l_{1m}l_{2n} - (l_2 - l_{2n})u_1 - (l_1 - l_{1m})u_2, \\ & c_{mn} = u_{1m}l_{2n} - u_1l_2 + (l_2 - l_{2n})l_1 + (u_1 - u_{1m})u_2, \\ & d_{mn} = l_{1m}u_{2n} - l_1u_2 + (u_2 - u_{2n})u_1 + (l_1 - l_{1m})l_2, \\ & l_1 \leq Y_1 \leq u_1,\ l_2 \leq Y_2 \leq u_2, \\ & \sum_{m\in\mathcal{M}} \sum_{n\in\mathcal{N}} \lambda_{mn} = 1,\ \lambda_{mn} \in \{0,1\},\ \forall m \in \mathcal{M}, n \in \mathcal{N}\}. \end{aligned}$$

Then, $S \subseteq S^U \subseteq S^G \subseteq S^M = S^U_{LP} \subseteq S^G_{LP}$, where S^U_{LP} and S^G_{LP} represent the linear programming relaxation of S^U and S^G, respectively; i.e., with $\lambda_{mn} \in \{0,1\}$ replaced by $0 \leq \lambda_{mn} \leq 1$, $\forall m \in \mathcal{M}, n \in \mathcal{N}$. Moreover, $S \subseteq S^U$, $S^G \subseteq S^M$, and $S^U_{LP} \subseteq S^G_{LP}$ can be strict.

Proof. Let $(Y_1, Y_2, Z) \in S$. Let $\lambda_{\bar{m}\bar{n}} = 1$ for an interval satisfying $Y_1 \in [l_{1\bar{m}}, u_{1\bar{m}}]$ and $Y_2 \in [l_{2\bar{n}}, u_{2\bar{n}}]$, breaking ties arbitrarily, and let $\lambda_{mn} = 0,\ \forall m \in \mathcal{M}, n \in$

$\mathcal{N} : (m,n) \neq (\bar{m},\bar{n})$. For $(m,n) \neq (\bar{m},\bar{n})$, constraints (2.6a)-(2.6d) yield $l_1 \leq Y_1 \leq u_1$, $l_2 \leq Y_2 \leq u_2$, and for $(m,n) = (\bar{m},\bar{n})$, these constraints yield $l_{1m} \leq Y_1 \leq u_{1m}$, $l_{2n} \leq Y_2 \leq u_{2n}$. Consider constraints (2.6e)-(2.6h). For $(m,n) = (\bar{m},\bar{n})$, these constraints yield

$$\begin{aligned} Z &\geq u_{2n}Y_1 + u_{1m}Y_2 - u_{1m}u_{2n} && \text{(2.7a)} \\ Z &\geq l_{2n}Y_1 + l_{1m}Y_2 - l_{1m}l_{2n} && \text{(2.7b)} \\ Z &\leq l_{2n}Y_1 + u_{1m}Y_2 - u_{1m}l_{2n} && \text{(2.7c)} \\ Z &\leq u_{2n}Y_1 + l_{1m}Y_2 - l_{1m}u_{2n}. && \text{(2.7d)} \end{aligned}$$

The following shows that constraint (2.7a) holds for $(Y_1, Y_2, Z) \in S$ with $Y_1 \in [l_{1\bar{m}}, u_{1\bar{m}}]$ and
$Y_2 \in [l_{2\bar{n}}, u_{2\bar{n}}]$:

$$\begin{aligned} 0 &\leq (u_{1m} - Y_1)(u_{2n} - Y_2) \\ 0 &\leq Y_1Y_2 - u_{2n}Y_1 - u_{1m}Y_2 + u_{1m}u_{2n} \\ Z &\geq u_{2n}Y_1 + u_{1m}Y_2 - u_{1m}u_{2n}. \end{aligned}$$

Similarly, constraints (2.7b)-(2.7d) hold via:

$$\begin{aligned} 0 \leq &(Y_1 - l_{1m})(Y_2 - l_{2n}) \Rightarrow Z \geq l_{2n}Y_1 + l_{1m}Y_2 - l_{1m}l_{2n} \\ 0 \leq &(u_{1m} - Y_1)(Y_2 - l_{2n}) \Rightarrow Z \leq l_{2n}Y_1 + u_{1m}Y_2 - u_{1m}l_{2n} \\ 0 \leq &(Y_1 - l_{1m})(u_{2n} - Y_2) \Rightarrow Z \leq u_{2n}Y_1 + l_{1m}Y_2 - l_{1m}u_{2n}. \end{aligned}$$

Therefore, (Y_1, Y_2, Z) satisfies constraints (2.6e)-(2.6h) for $(m,n) = (\bar{m},\bar{n})$. If $(m,n) \neq (\bar{m},\bar{n})$, the right-hand sides of these constraints for S^U are dominated by those of the

McCormick relaxation of S^M; i.e.,

$$\begin{aligned}
\text{(2.6e)}: \quad & u_{2n}Y_1 + u_{1m}Y_2 - u_{1m}u_{2n} - (u_1u_2 - u_{1m}u_{2n} - (u_2 - u_{2n})l_1 - (u_1 - u_{1m})l_2) \\
&= u_{2n}Y_1 + u_{1m}Y_2 - u_1u_2 + (u_2 - u_{2n})l_1 + (u_1 - u_{1m})l_2 \\
&\leq u_{2n}Y_1 + u_{1m}Y_2 - u_1u_2 + (u_2 - u_{2n})Y_1 + (u_1 - u_{1m})Y_2 \\
&= u_2Y_1 + u_1Y_2 - u_1u_2 \quad \forall (m,n) \in \mathcal{M} \times \mathcal{N} \setminus \{(\bar{m}, \bar{n})\} \\
\text{(2.6f)}: \quad & l_{2n}Y_1 + l_{1m}Y_2 - l_{1m}l_{2n} - (l_1l_2 - l_{1m}l_{2n} - (l_2 - l_{2n})u_1 - (l_1 - l_{1m})u_2) \\
&= l_{2n}Y_1 + l_{1m}Y_2 - l_1l_2 - (l_{2n} - l_2)u_1 - (l_{1m} - l_1)u_2 \\
&\leq l_{2n}Y_1 + l_{1m}Y_2 - l_1l_2 - (l_{2n} - l_2)Y_1 - (l_{1m} - l_1)Y_2 \\
&= l_2Y_1 + l_1Y_2 - l_1l_2 \quad \forall (m,n) \in \mathcal{M} \times \mathcal{N} \setminus \{(\bar{m}, \bar{n})\} \\
\text{(2.6g)}: \quad & l_{2n}Y_1 + u_{1m}Y_2 - u_{1m}l_{2n} + (u_{1m}l_{2n} - u_1l_2 + (l_2 - l_{2n})l_1 + (u_1 - u_{1m})u_2) \\
&= l_{2n}Y_1 + u_{1m}Y_2 - u_1l_2 + (l_2 - l_{2n})l_1 + (u_1 - u_{1m})u_2 \\
&\geq l_{2n}Y_1 + u_{1m}Y_2 - u_1l_2 + (l_2 - l_{2n})Y_1 + (u_1 - u_{1m})Y_2 \\
&= l_2Y_1 + u_1Y_2 - u_1l_2 \quad \forall (m,n) \in \mathcal{M} \times \mathcal{N} \setminus \{(\bar{m}, \bar{n})\} \\
\text{(2.6h)}: \quad & u_{2n}Y_1 + l_{1m}Y_2 - l_{1m}u_{2n} + (l_{1m}u_{2n} - l_1u_2 + (u_2 - u_{2n})u_1 + (l_1 - l_{1m})l_2) \\
&= u_{2n}Y_1 + l_{1m}Y_2 - l_1u_2 + (u_2 - u_{2n})u_1 + (l_1 - l_{1m})l_2 \\
&\geq u_{2n}Y_1 + l_{1m}Y_2 - l_1u_2 + (u_2 - u_{2n})Y_1 + (l_1 - l_{1m})Y_2 \\
&= u_2Y_1 + l_1Y_2 - l_1u_2 \quad \forall (m,n) \in \mathcal{M} \times \mathcal{N} \setminus \{(\bar{m}, \bar{n})\}.
\end{aligned}$$

Therefore, (Y_1, Y_2, Z) satisfies constraints (2.6e)-(2.6h) when $(m,n) \neq (\bar{m}, \bar{n})$, and so $(Y_1, Y_2, Z) \in S \Rightarrow (Y_1, Y_2, Z) \in S^U$.

Suppose $(Y_1, Y_2, Z) \in S^U$. The constraints of S^G are identical to those of S^U for $(m,n) = (\bar{m}, \bar{n})$, and S^U has tighter constraints than S^G when $(m,n) \neq (\bar{m}, \bar{n})$,

because the values for $a_{mn},\ b_{mn},\ c_{mn}$, and d_{mn} in S^U are smaller than the analogous terms specified in the definition of S^G; i.e.,

$$\begin{aligned}
\text{(2.6e)}: \quad & (u_1 - l_1)(u_2 - l_2) - (u_1u_2 - u_{1m}u_{2n} - (u_2 - u_{2n})l_1 - (u_1 - u_{1m})l_2) \\
& = (u_{1m} - l_1)(u_{2n} - l_2) \geq 0 \\
\text{(2.6f)}: \quad & (u_1 - l_1)(u_2 - l_2) - (l_1l_2 - l_{1m}l_{2n} - (l_2 - l_{2n})u_1 - (l_1 - l_{1m})u_2) \\
& = (u_1 - l_{1m})(u_2 - l_{2n}) \geq 0 \\
\text{(2.6g)}: \quad & (u_1 - l_1)(u_2 - l_2) - (u_{1m}l_{2n} - u_1l_2 + (l_2 - l_{2n})l_1 + (u_1 - u_{1m})u_2) \\
& = (u_{1m} - l_1)(u_2 - l_{2n}) \geq 0 \\
\text{(2.6h)}: \quad & (u_1 - l_1)(u_2 - l_2) - (l_{1m}u_{2n} - l_1u_2 + (u_2 - u_{2n})u_1 + (l_1 - l_{1m})l_2) \\
& = (u_1 - l_{1m})(u_{2n} - l_2) \geq 0.
\end{aligned}$$

Therefore, $(Y_1, Y_2, Z) \in S^U \Rightarrow (Y_1, Y_2, Z) \in S^G$ and
$(Y_1, Y_2, Z) \in S^U_{LP} \Rightarrow (Y_1, Y_2, Z) \in S^G_{LP}$.

Suppose $(Y_1, Y_2, Z) \in S^G$, $Y_1 \in [l_{1\bar{m}}, u_{1\bar{m}}]$, and $Y_2 \in [l_{2\bar{n}}, u_{2\bar{n}}]$. We restrict our attention to the constraints for $(m, n) = (\bar{m}, \bar{n})$, which dominate those for $(m, n) \neq (\bar{m}, \bar{n})$. Using a similar argument to that for $S \subseteq S^U$ above, we can show that the first constraint of S^M holds:

$$\begin{aligned}
& 0 \leq (u_{1m} - Y_1)(u_{2n} - Y_2) \\
& 0 \leq Y_1Y_2 - u_{1m}Y_2 + u_{2n}(u_{1m} - Y_1) \leq Y_1Y_2 - u_{1m}Y_2 + u_2(u_{1m} - Y_1) \\
& 0 \leq Y_1Y_2 - u_2Y_1 + u_{1m}(u_2 - Y_2) \leq Y_1Y_2 - u_2Y_1 + u_1(u_2 - Y_2) \\
& Z \geq u_2Y_1 + u_1Y_2 - u_1u_2.
\end{aligned}$$

Analogous arguments show that the next three constraints of S^M also hold; hence, $(Y_1, Y_2, Z) \in S^G \Rightarrow (Y_1, Y_2, Z) \in S^M$.

For $(m, n) = (|\mathcal{M}|, |\mathcal{N}|)$, $u_{1m} = u_1$ and $u_{2n} = u_2$, so $a_{mn} = 0$, and constraint (2.6e) replicates the first constraint in S^M. Likewise, the pairs $(1, 1), (|\mathcal{M}|, 1)$, and $(1, |\mathcal{N}|)$ yield instances of constraints (2.6f), (2.6g), and (2.6h) that are equivalent to the second, third, and fourth constraints in S^M, respectively. This shows that $S^U_{LP} \subseteq S^M$. However, the extreme points of S^M are:

$$S^M_e = \{(u_1, u_2, u_1u_2), (l_1, l_2, l_1l_2), (u_1, l_2, u_1l_2), (l_1, u_2, l_1u_2)\},$$

all of which are points in S and, by extension, S^U_{LP}. Further, because S^U_{LP} is convex, $\text{conv}(S^M_e) = S^M \subseteq S^U_{LP}$. Therefore, $S^M = S^U_{LP}$.

For examples showing that the relationships $S \subseteq S^U$ and $S^G \subseteq S^M$ can be strict, let $l_1 = l_2 = 0$, $u_1 = u_2 = 1$, $|\mathcal{M}| = 1$, and $|\mathcal{N}| = 2$ with intervals of equal length. Then, $(Y_1, Y_2, Z) = (0.6, 0.6, 0.4) \in S^U$ but $(0.6, 0.6, 0.4) \notin S$, and $(0.6, 0.6, 0.6) \in S^M$ but $(0.6, 0.6, 0.6) \notin S^G$. To show $S^U_{LP} \subseteq S^G_{LP}$ can be strict, the solution $(0.5, 0.5, 0.75) \in S^G_{LP}$ for $\lambda_{1,1} = \lambda_{1,2} = 0.5$, but $(0.5, 0.5, 0.75) \notin S^U_{LP}$. ■

Proposition 2.3.1 shows that the set we propose, S^U, is a tighter relaxation of S than those available in the literature (i.e., $S \subset S^U \subset S^G$ and $S^U_{LP} \subset S^G_{LP}$) and, further, that its linear programming relaxation is as tight as possible (i.e., $\text{conv}(S) = S^M = S^U_{LP}$), where $\text{conv}(S) = S^M$ is established in Al-Khayyal and Falk (1983). In our context, the value of Proposition 2.3.1 is that the set S captures how bilinear terms are treated in model ($\mathcal{P}$), and sets S^U, S^G, and S^M capture how we relax the

bilinear terms in models ($\mathcal{U}$), ($\mathcal{G}$), and ($\mathcal{M}$), respectively. Section 2.4 investigates the computational advantage offered by the tighter model ($\mathcal{U}$).

2.3.3 Application to Microgrid Design and Dispatch Problem

We adopt the most precise battery modeling paradigm of which the authors are aware at the time of this writing (Scioletti et al. 2016b), in which two sets of physical constraints in the microgrid design and dispatch problem include bilinear terms: (i) the battery power input and output at each time period is the product of incoming or outgoing current, respectively, and voltage, which, in turn, is a linear function of battery state-of-charge; and, (ii) our application models battery degradation as a function of current and state-of-charge, which includes the product of the two as a bilinear term. Hence, the product of battery state-of-charge and incoming or outgoing current at each time period compose all the bilinear terms in the formulation of ($\mathcal{P}$) because this accounts for both the battery's power and the manner in which battery lifecycle degradation occurs. We map Y_{1t} to the starting state-of-charge and Y_{2t} to the current discharged from the battery in each time period, and we have an analogous pairing for the state-of-charge and current used to charge the battery, respectively.

Partitioning on the domain of both Y_{1t} and Y_{2t} causes the model to grow quickly in size, which can lead to intractable instances. Thus, without loss of generality, we partition only on Y_{2t} to obtain the following special case of model ($\mathcal{U}$):

$$\sum_{n \in \mathcal{N}} \lambda_{nt} = 1, \qquad \forall t \in \mathcal{T} \tag{2.8a}$$

$$Y_{2t} \geq l_2 + (l_{2n} - l_2)\lambda_{nt}, \qquad \forall n \in \mathcal{N}, t \in \mathcal{T} \tag{2.8b}$$

$$Y_{2t} \leq u_2 - (u_2 - u_{2n})\lambda_{nt}, \qquad \forall n \in \mathcal{N}, t \in \mathcal{T} \tag{2.8c}$$

$$Z_t \geq u_{2n}Y_{1t} + u_1 Y_{2t} - u_1 u_{2n} - (u_1 - l_1)(u_2 - u_{2n})(1 - \lambda_{nt}), \qquad \forall n \in \mathcal{N}, t \in \mathcal{T} \tag{2.8d}$$

$$Z_t \geq l_{2n}Y_{1t} + l_1 Y_{2t} - l_1 l_{2n} - (u_1 - l_1)(l_{2n} - l_2)(1 - \lambda_{nt}), \qquad \forall n \in \mathcal{N}, t \in \mathcal{T} \tag{2.8e}$$

$$Z_t \leq l_{2n}Y_{1t} + u_1 Y_{2t} - u_1 l_{2n} + (u_1 - l_1)(l_{2n} - l_2)(1 - \lambda_{nt}), \quad \forall n \in \mathcal{N}, t \in \mathcal{T} \tag{2.8f}$$

$$Z_t \leq u_{2n}Y_{1t} + l_1 Y_{2t} - l_1 u_{2n} + (u_1 - l_1)(u_2 - u_{2n})(1 - \lambda_{nt}), \qquad \forall n \in \mathcal{N}, t \in \mathcal{T} \tag{2.8g}$$

$$\lambda_{nt} \in \{0, 1\}, \quad \forall n \in \mathcal{N}, t \in \mathcal{T}. \tag{2.8h}$$

Constraints (2.8) are the special case of constraints (2.5) in which $|\mathcal{M}| = 1$; hence, we have removed index m and only use simple bounds l_1 and u_1 in our application. The formulation in Appendix A.1 details the mappings of Y_{1t}, Y_{2t}, and Z_t to the state-of-charge and current decision variables in the microgrid design and dispatch problem.

2.4 Decomposition of MIP Formulation

While the approach we describe in Section 2.3 reduces approximation error compared to a McCormick linearization, the additional variables and constraints may compromise tractability. Because models ($\mathcal{P}$), ($\mathcal{U}$), ($\mathcal{G}$), and ($\mathcal{M}$) are loosely coupled with respect to time, a temporal decomposition can expedite computation. Rockafellar and Wets (1991) present the progressive hedging algorithm, which couples a Lagrangian relaxation of nonanticipativity constraints with a proximal term. Gade et al. (2014) and Escudero et al. (2016) both form lower bounds on the optimal value of multi-stage stochastic programs through Lagrangian relaxations of (subsets of) nonanticipativity constraints. This section introduces our approach to decomposing

a larger problem into subproblems that can be solved in parallel to obtain upper and lower bounds on each model. While our approach can be applied to any of ($\mathcal{P}$), ($\mathcal{U}$), ($\mathcal{G}$), and ($\mathcal{M}$), for simplicity, we frame the discussion in terms of problem ($\mathcal{P}$), and we refer to the modified models that yield upper and lower bounds as ($\bar{\mathcal{P}}$) and ($\underline{\mathcal{P}}$), respectively.

The decomposition procedure begins by exploiting the time blocks indexed by $\ell \in \mathcal{L}$ in model ($\mathcal{P}$) to create a subproblem of the model for each interval; we create a clone of each strategic decision variable in each subproblem, and we introduce nonanticipativity constraints that force all subproblem strategic decisions to be equal. Next, we obtain a lower bound by relaxing the nonanticipativity constraints. We then obtain an upper bound by selecting and fixing a specific strategic decision, including fixing the inventory level on the boundaries of each block. Updating the Lagrange multipliers associated with the relaxed nonanticipativity constraints tightens the lower bound, and we propose an update scheme that performs well when applied to the microgrid design and dispatch problem.

2.4.1 ($\underline{\mathcal{P}}$) Formulation: Lower Bounds

To decompose model ($\mathcal{P}$) into $|\mathcal{L}|$ subproblems that together provide a lower bound on model ($\mathcal{P}$), we start by allowing for separate strategic decisions, X and R, per subproblem, and rewriting model ($\mathcal{P}$) as follows, in which we define μ^ℓ and θ^ℓ as dual variables for the nonanticipativity constraints on X and R, respectively. (Typically, nonanticipativity constraints prevent decision variables from adapting to individual scenarios in a stochastic program. Here, they prevent strategic decisions

from adapting to individual time blocks, ℓ.)

Additional Sets

$\mathcal{T}_\ell^-$	time periods in block ℓ, excluding the first period; i.e.,

$$\mathcal{T}_\ell^- = \mathcal{T}_\ell \setminus \{(\ell-1)\upsilon + 1\},\ \forall \ell \in \mathcal{L}$$

Additional Decision Variables

X^ℓ	strategic design decision for subproblem ℓ
R^ℓ	strategic inventory reset level for subproblem ℓ

Formulation

$$z^P = \min_{X, X^\ell, R, R^\ell, Y, \underline{Y}, \bar{Y}} \ \frac{1}{|\mathcal{L}|} \sum_{\ell \in \mathcal{L}} f_0(X^\ell) + \sum_{\ell \in \mathcal{L}} \sum_{t \in \mathcal{T}_\ell} f_t(Y_t) \tag{2.9a}$$

$$\text{s.t.} \quad X^\ell \in \mathcal{X},\ \forall \ell \in \mathcal{L} \tag{2.9b}$$

$$Y_t \in \mathcal{Y}_t(X^\ell), \quad \forall t \in \mathcal{T}_\ell, \ell \in \mathcal{L} \tag{2.9c}$$

$$\bar{Y}_t = \underline{Y}_t + g_t(Y_t), \quad \forall t \in \mathcal{T} \tag{2.9d}$$

$$\underline{Y}_t = \bar{Y}_{t-1}, \quad \forall t \in \mathcal{T}_\ell^-, \ell \in \mathcal{L} \tag{2.9e}$$

$$\underline{Y}_{(\ell-1)\upsilon+1} = R^\ell, \quad \forall \ell \in \mathcal{L} \setminus \{1\} \tag{2.9f}$$

$$\bar{Y}_{\ell\upsilon} = R^\ell, \quad \forall \ell \in \mathcal{L} \tag{2.9g}$$

$$X^\ell = X, \quad \forall \ell \in \mathcal{L} : \mu^\ell \tag{2.9h}$$

$$R^\ell = R, \quad \forall \ell \in \mathcal{L} : \theta^\ell \tag{2.9i}$$

$$\underline{Y}_1 = Y_0 \tag{2.9j}$$

Model (2.9) clones the strategic decisions X and R via X^ℓ and R^ℓ, respectively, but enforces nonanticipativity constraints (2.9h) and (2.9i), and hence is equivalent to ($\mathcal{P}$)

as specified in model (2.1). The objective function in (2.9a) is modified to incorporate the new X^ℓ variables. Constraints (2.9b)-(2.9d) replicate those of (2.1b)-(2.1d), when the nonanticipativity constraint (2.9h) is included. Constraint (2.9e) captures the inventory constraint (2.1e) for time periods within a block, i.e., for $t \in \mathcal{T}_\ell^-$, $\ell \in \mathcal{L}$. Constraints (2.9f) and (2.9g), along with constraint (2.9i), maintain inventory across time blocks. Below, we use the dual variables indicated on constraints (2.9h) and (2.9i) to develop a Lagrangian relaxation of model (2.9). We refer to the following model as ($\underline{\mathcal{P}}$) because it provides a lower bound on model ($\mathcal{P}$)'s optimal value.

($\underline{\mathcal{P}}$) Formulation

$$\underline{z}^P = \min_{X^\ell, R^\ell, Y, \underline{Y}, \bar{Y}} \quad \frac{1}{|\mathcal{L}|} \sum_{\ell \in \mathcal{L}} f_0(X^\ell) + \sum_{\ell \in \mathcal{L}} \sum_{t \in \mathcal{T}_\ell} f_t(Y_t) + \sum_{\ell \in \mathcal{L}} (\mu^\ell X^\ell + \theta^\ell R^\ell) \tag{2.10a}$$

$$\text{s.t.} \quad X^\ell \in \mathcal{X}, \quad \forall \ell \in \mathcal{L} \tag{2.10b}$$

$$Y_t \in \mathcal{Y}_t(X^\ell), \quad \forall t \in \mathcal{T}_\ell, \ell \in \mathcal{L} \tag{2.10c}$$

$$\bar{Y}_t = \underline{Y}_t + g_t(Y_t), \quad \forall t \in \mathcal{T} \tag{2.10d}$$

$$\underline{Y}_t = \bar{Y}_{t-1}, \quad \forall t \in \mathcal{T}_\ell^-, \ell \in \mathcal{L} \tag{2.10e}$$

$$\underline{Y}_{(\ell-1)\upsilon+1} = R^\ell, \quad \forall \ell \in \mathcal{L} \setminus \{1\} \tag{2.10f}$$

$$\bar{Y}_{\ell\upsilon} = R^\ell, \quad \forall \ell \in \mathcal{L} \tag{2.10g}$$

$$\underline{Y}_1 = Y_0 \tag{2.10h}$$

We require $\sum_{\ell \in \mathcal{L}} \mu^\ell = 0$ and $\sum_{\ell \in \mathcal{L}} \theta^\ell = 0$ to avoid an unbounded Lagrangian relaxation, which eliminates the free variables X and R. We can compute the lower bound of model (2.10), $\underline{z}^P$, by separating model ($\underline{\mathcal{P}}$) into $|\mathcal{L}|$ subproblems, one for each time block indexed by $\ell \in \mathcal{L}$. Proposition 2.4.1 shows that the optimal value of ($\underline{\mathcal{P}}$)

provides a lower bound on model ($\mathcal{P}$), and extends the reformulation to create models ($\underline{\mathcal{U}}$), ($\underline{\mathcal{G}}$), and ($\underline{\mathcal{M}}$) to form lower bounds on models ($\mathcal{U}$), ($\mathcal{G}$), and ($\mathcal{M}$), respectively. The proposition further specifies relationships among these models.

Proposition 2.4.1. *Let z^P, z^U, z^G, and z^M be the optimal values of ($\mathcal{P}$), ($\mathcal{U}$), ($\mathcal{G}$), and ($\mathcal{M}$), respectively. Similarly, let $\underline{z}^P$, $\underline{z}^U$, $\underline{z}^G$, and $\underline{z}^M$ be the optimal values of models ($\underline{\mathcal{P}}$), ($\underline{\mathcal{U}}$), ($\underline{\mathcal{G}}$), and ($\underline{\mathcal{M}}$), i.e., of model* (2.10) *specialized to the bilinear representation, or linear approximation, under ($\mathcal{P}$), ($\mathcal{U}$), ($\mathcal{G}$), and ($\mathcal{M}$), respectively. Then, $z^P \geq \underline{z}^P$, $z^U \geq \underline{z}^U$, $z^G \geq \underline{z}^G$, $z^M \geq \underline{z}^M$, and $z^P \geq z^U \geq z^G \geq z^M$. Further, assume that the same multipliers μ^ℓ, θ^ℓ, $\forall \ell \in \mathcal{L}$, are used in models ($\underline{\mathcal{P}}$), ($\underline{\mathcal{U}}$), ($\underline{\mathcal{G}}$), and ($\underline{\mathcal{M}}$). Then, $\underline{z}^P \geq \underline{z}^U \geq \underline{z}^G \geq \underline{z}^M$.*

Proof. Model ($\mathcal{P}$) as defined in (2.1) is equivalent to model (2.9), and model ($\underline{\mathcal{P}}$) as defined in (2.10) is a Lagrangian relaxation of (2.9) under the restriction on the multipliers of $\sum_{\ell \in \mathcal{L}} \mu^\ell = 0$ and $\sum_{\ell \in \mathcal{L}} \theta^\ell = 0$. Thus, $z^P \geq \underline{z}^P$. An analogous argument shows that $z^U \geq \underline{z}^U$, $z^G \geq \underline{z}^G$, and $z^M \geq \underline{z}^M$. Using Proposition 2.3.1, we have $z^M \leq z^G \leq z^U \leq z^P$, and $\underline{z}^M \leq \underline{z}^G \leq \underline{z}^U \leq \underline{z}^P$. ■

2.4.2 ($\bar{\mathcal{P}}$) Formulation: Upper Bounds

We make two modifications to model (2.9) to obtain an upper bound on ($\mathcal{P}$)'s optimal value, and call the resulting model ($\bar{\mathcal{P}}$). First, we fix a design decision $\hat{X} \in \mathcal{X}$, and second, we fix a "reset point" for inventory levels at boundaries of the partition on $\mathcal{T}$, which we call $\hat{R}$. We obtain as many as $|\mathcal{L}|$ candidate solutions by solving ($\underline{\mathcal{P}}$), and we may select one of these; our implementation retrieves candidates from peak

demand intervals and retains that which provides the tightest upper bound upon solving $(\bar{\mathcal{P}})$.

$(\bar{\mathcal{P}})$ Formulation

$$\bar{z}^P = \min_{Y,\underline{Y},\bar{Y}} \quad f_0(\hat{X}) + \sum_{\ell\in\mathcal{L}}\sum_{t\in\mathcal{T}_\ell} f_t(Y_t) \tag{2.11a}$$

$$\text{s.t.} \quad Y_t \in \mathcal{Y}_t(\hat{X}), \quad \forall t \in \mathcal{T}_\ell, \ell \in \mathcal{L} \tag{2.11b}$$

$$\bar{Y}_t = \underline{Y}_t + g_t(Y_t), \quad \forall t \in \mathcal{T} \tag{2.11c}$$

$$\underline{Y}_t = \bar{Y}_{t-1}, \quad \forall t \in \mathcal{T}_\ell^-, \ell \in \mathcal{L} \tag{2.11d}$$

$$\underline{Y}_{(\ell-1)\upsilon+1} = \hat{R}, \quad \forall \ell \in \mathcal{L} \setminus \{1\} \tag{2.11e}$$

$$\bar{Y}_{\ell\upsilon} = \hat{R}, \quad \forall \ell \in \mathcal{L} \tag{2.11f}$$

$$\underline{Y}_1 = Y_0 \tag{2.11g}$$

Fixing the design decision at $\hat{X}$ and inventory reset value at $\hat{R}$ allows for the removal of constraints (2.9b), (2.9h) and (2.9i). The boundary conditions (2.11e) and (2.11f) link the pairs of decision variables $\underline{Y}$ and $\bar{Y}$ that span multiple intervals within the partition on $\mathcal{T}$ by fixing their values to the reset point $\hat{R}$; these replace constraint (2.9i), and allow model (2.11) to separate by $\ell \in \mathcal{L}$. The optimal value of $(\bar{\mathcal{P}})$ provides an upper bound on the optimal value of $(\mathcal{P})$, per Proposition 2.4.2.

Proposition 2.4.2. *Let $\hat{X} \in \mathcal{X}$ and $\hat{R}$ be given, and assume that for those $\hat{X}$ and $\hat{R}$, model $(\bar{\mathcal{P}})$ is feasible. Let z^P, z^U, z^G, and z^M be the optimal values of $(\mathcal{P})$, $(\mathcal{U})$, $(\mathcal{G})$, and $(\mathcal{M})$, respectively. Similarly, let $\bar{z}^P$, $\bar{z}^U$, $\bar{z}^G$, and $\bar{z}^M$ be the optimal values of models $(\bar{\mathcal{P}})$, $(\bar{\mathcal{U}})$, $(\bar{\mathcal{G}})$, and $(\bar{\mathcal{M}})$, i.e., of model* (2.11) *specialized to the bilinear*

representation, or linear approximation, under ($\mathcal{P}$), ($\mathcal{U}$), ($\mathcal{G}$), and ($\mathcal{M}$), respectively. Then, $\bar{z}^P \geq z^P$, $\bar{z}^U \geq z^U$, $\bar{z}^G \geq z^G$, and $\bar{z}^M \geq z^M$. Further, $\bar{z}^P \geq \bar{z}^U \geq \bar{z}^G \geq \bar{z}^M$.

Proof. By Proposition 2.3.1, feasibility of ($\bar{\mathcal{P}}$) ensures feasibility of ($\bar{\mathcal{U}}$), ($\bar{\mathcal{G}}$), and ($\bar{\mathcal{M}}$) under $\hat{X}$ and $\hat{R}$. Consider models ($\bar{\mathcal{P}}$) as defined in (2.11) and ($\mathcal{P}$) as defined in (2.9). Boundary conditions (2.11e) and (2.11f) are equivalent to constraints (2.9f) and (2.9g), except that the inventory variable $R = \hat{R}$ has been fixed. Additionally, the design decision $X = \hat{X} \in \mathcal{X}$ is fixed. Therefore, ($\bar{\mathcal{P}}$) is a restriction of ($\mathcal{P}$), and hence its optimal value yields an upper bound for ($\mathcal{P}$); i.e., $\bar{z}^P \geq z^P$. Analogous arguments show $\bar{z}^U \geq z^U$, $\bar{z}^G \geq z^G$, and $\bar{z}^M \geq z^M$. Using Proposition 2.3.1, we have $\bar{z}^P \geq \bar{z}^U \geq \bar{z}^G \geq \bar{z}^M$. ■

2.4.3 Decomposition Algorithm

Algorithm 1 iteratively improves the solutions to the lower and upper bounding formulations in Sections 2.4.1 and 2.4.2, respectively, tightening the bounds as the algorithm proceeds. For simplicity, we describe the algorithm as applied to a generic model, ($\mathcal{A}$), but it may be applied to any of ($\mathcal{P}$), ($\mathcal{U}$), ($\mathcal{G}$), and ($\mathcal{M}$). Additionally, valid mixtures according to Propositions 2.4.1 and 2.4.2 are possible; for example, we may use ($\bar{\mathcal{P}}$)=($\bar{\mathcal{A}}$) and ($\underline{\mathcal{U}}$)=($\underline{\mathcal{A}}$) to obtain upper and lower bounds on model ($\mathcal{P}$)=($\mathcal{A}$).

To obtain ε-optimal solutions to model ($\mathcal{A}$), we iteratively solve models ($\underline{\mathcal{A}}$) and ($\bar{\mathcal{A}}$) using a variation of the progressive hedging method developed by Rockafellar and Wets (1991). Let X_0 and R_0 denote a design decision and an inventory level, respectively, obtained by selecting a candidate from solutions to the subproblems of model ($\underline{\mathcal{A}}$). Then, Algorithm 1 aims to iteratively improve upper and lower bounds

on the optimal value of the model, which we call z^A, until we obtain a solution whose optimality gap is within the desired tolerance.

Each iteration in Algorithm 1 updates Lagrange multipliers θ^ℓ according to input parameters ρ_θ, the distance between each reset inventory level R^ℓ, and its time-weighted mean value; we update multipliers μ^ℓ using ρ_μ and X^ℓ under an analogous method. Algorithm 1 is similar to progressive hedging, but in solving $(\underline{\mathcal{A}})$ we do not include a proximal term in the objective function; therefore, solving $(\underline{\mathcal{A}})$ provides a valid lower bound for z^A at every iteration. Figure 2.3 displays the key outputs of each model under our approach.

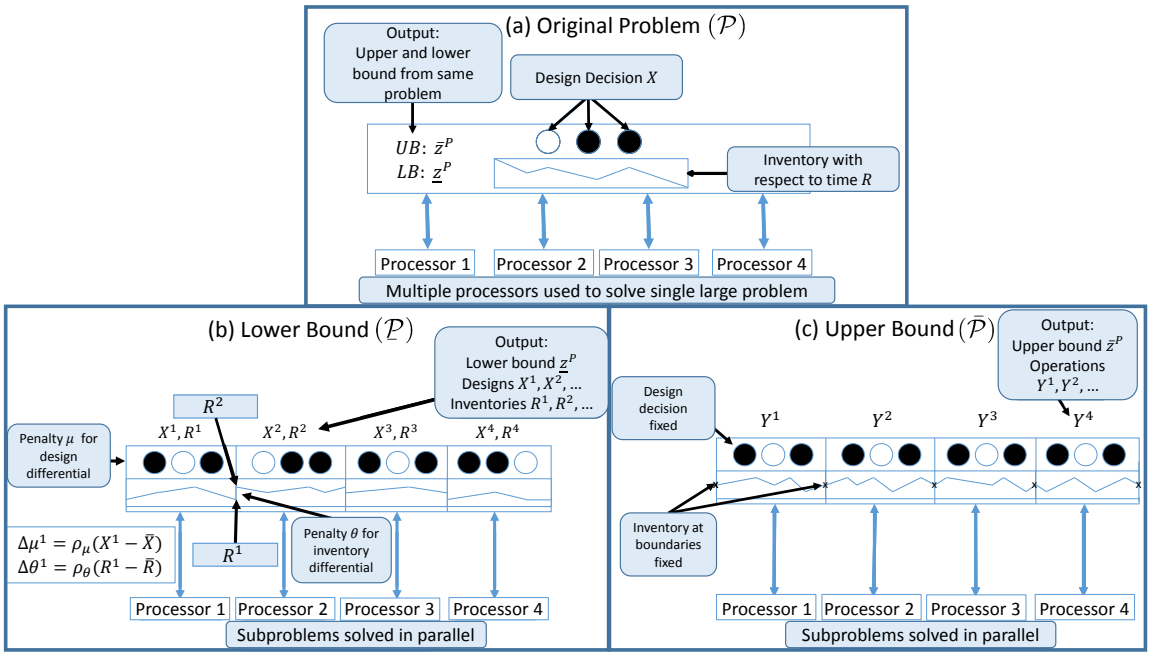

Figure 2.3: Illustration of direct solution of $(\mathcal{P})$ and Algorithm 1 applied to $(\mathcal{A})$=$(\mathcal{P})$. Part (a) of the figure depicts how a general purpose MINLP solver would solve model $(\mathcal{P})$. The bounds $\underline{z}$ and $\bar{z}$ are from a branch-and-bound algorithm, and that algorithm could use multiple processors. Based on Algorithm 1, part (b) shows the reconciliation of the subproblems of $(\underline{\mathcal{P}})$, while part (c) shows that of the subproblems of $(\bar{\mathcal{P}})$.

Algorithm 1 Decomposition procedure to approximately solve model $(\mathcal{A})$

procedure DECOMPOSITION

Inputs: $\varepsilon > 0, \rho_\mu > 0, \rho_\theta > 0, k \in \mathbb{Z}_+, \kappa \in \mathbb{Z}_+, (\underline{\mathcal{A}}), (\bar{\mathcal{A}})$ ▷ stopping criterion, ▷ proximal weights, upper bound search frequency, upper bound search depth, ▷ models to obtain lower and upper bounds

Outputs: $X^*, R^*, \bar{z}^A, \underline{z}^A$ ▷ ε-optimal design and inventory, ▷ upper and lower bounds on z^A

$i \leftarrow 0;\ \underline{z}^A \leftarrow -\infty;\ \bar{z}^A \leftarrow \infty$ ▷ iteration and initial lower and upper bounds

$\mu^\ell \leftarrow 0,\ \theta^\ell \leftarrow 0,\ \forall \ell \in \mathcal{L}$ ▷ initial Lagrange multipliers for model $(\underline{\mathcal{A}})$

while $\bar{z}^A - \underline{z}^A > \varepsilon \bar{z}^A$ **do**

Solve $(\underline{\mathcal{A}})$ with μ^ℓ, θ^ℓ to obtain $\underline{z}_i, X^\ell, R^\ell,\ \ell \in \mathcal{L}$ ▷ $\underline{z}_i$ is a lower bound for z^A ▷ $|\mathcal{L}|$ subproblems in $(\underline{\mathcal{A}})$ can be solved in parallel

$\bar{X} \leftarrow \frac{1}{|\mathcal{L}|} \sum_{\ell \in \mathcal{L}} X^\ell;\ \bar{R} \leftarrow \frac{1}{|\mathcal{L}|} \sum_{\ell \in \mathcal{L}} R^\ell$ ▷ mean of subproblem designs, ▷ reset inventories

for $\ell = 1, \ldots, |\mathcal{L}|$ **do** ▷ update Lagrange multipliers

$\mu^\ell \leftarrow \mu^\ell + \rho_\mu (X^\ell - \bar{X});\ \theta^\ell \leftarrow \theta^\ell + \rho_\theta (R^\ell - \bar{R})$

if $\underline{z}_i > \underline{z}^A$ **then** $\underline{z}^A \leftarrow \underline{z}_i$ ▷ update lower bound

if $i \bmod k = 0$ **then** ▷ attempt to find a new upper bound every k iterations

$j \leftarrow 0$

for $\ell \in \mathcal{L}$ **do**

if X^ℓ not previously used to solve $(\bar{\mathcal{A}})$ **then**

$j \leftarrow j + 1$

Solve $(\bar{\mathcal{A}})$ with $\hat{X} = X^\ell,\ \hat{R} = r(X^\ell)$ to obtain $\bar{z}_i^\ell$ ▷ $\bar{z}_i^\ell$ is an upper bound for z^A ▷ $r(X^\ell)$ is midpoint of bounds on R, given X^ℓ ▷ $|\mathcal{L}|$ subproblems in $(\bar{\mathcal{A}})$ can be solved in parallel

if $\bar{z}_i^\ell < \bar{z}^A$ **then** ▷ update upper bound and incumbent

Solve $(\bar{\mathcal{A}})$ with $\hat{X} = X^\ell$, optimizing R via bisection search to obtain R^ℓ, $\bar{z}_i^\ell$

$X^* \leftarrow X^\ell,\ R^* \leftarrow R^\ell,\ \bar{z}^A \leftarrow \bar{z}_i^\ell$

if $\bar{z}^A - \underline{z}^A \leq \varepsilon \bar{z}^A$ **or** $j = \kappa$ **then end-for**

$i \leftarrow i + 1$

return $(X^*, R^*, \bar{z}^A, \underline{z}^A)$

2.4.4 Application to a Microgrid Design and Dispatch Problem

In what follows, we detail our methodology to formulate, and update, models that bound ($\mathcal{P}$) specific to the microgrid design and dispatch problem developed by Scioletti et al. (2017). We use multiple models from this paper to approximate ($\mathcal{P}$), and we solve these via Algorithm 1. Specifically, we use a mixture of ($\bar{\mathcal{P}}$) and ($\underline{\mathcal{U}}$) to obtain upper and lower bounds, respectively.

2.4.4.1 Tightening ($\underline{\mathcal{U}}$): Lower Bound on Generator Capacity

Design decisions chosen for subproblems in off-peak time periods may be infeasible to the year-long problem due to insufficient generator capacity. To address this and to tighten the lower bound obtained by solving ($\underline{\mathcal{U}}$), we develop and solve a variant of model ($\underline{\mathcal{U}}$) in which in each subproblem we minimize the sum of the capacities of all diesel generators selected in the optimal design. That is, we seek a feasible solution to each subproblem that uses the smallest possible total generator capacity. The maximum of the optimal objective function values, i.e., the sum of the generator capacities, from these subproblems is then used to set a lower bound on diesel generator capacity when iteratively solving model ($\underline{\mathcal{U}}$) in Algorithm 1. Enforcing this minimum generator capacity strengthens constraints (2.10b)-(2.10c) for periods in which average and peak loads are low relative to those over the time horizon.

2.4.4.2 Obtaining Feasible Solutions to Model ($\mathcal{P}$)

We exploit the limited approximation error associated with model ($\mathcal{U}$) to develop solutions feasible to the nonlinear model ($\mathcal{P}$) by using a solution to model ($\mathcal{U}$) as

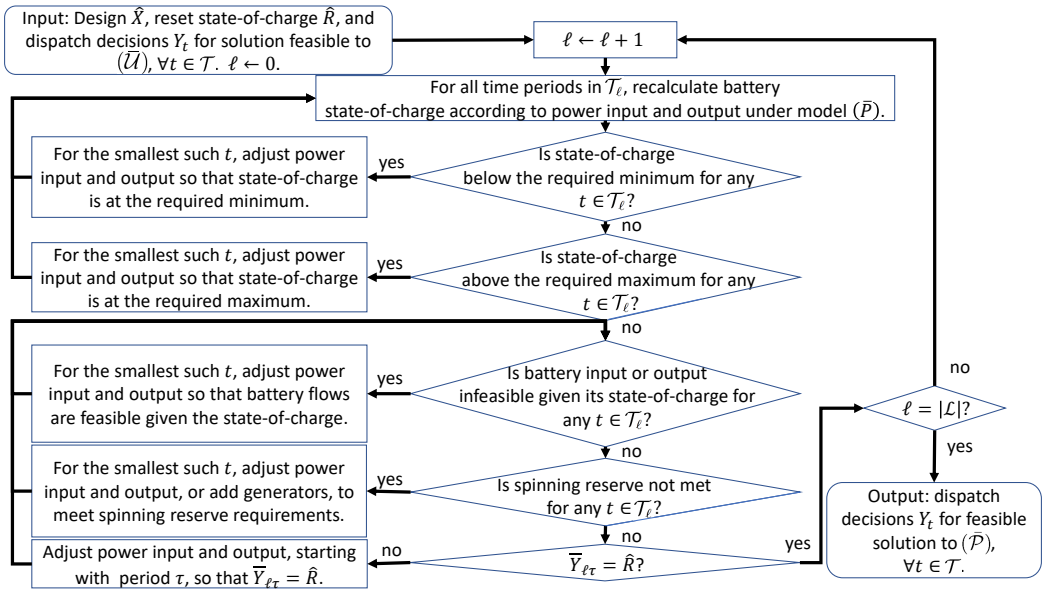

Figure 2.4: Flowchart describing procedure to find a feasible solution to $(\mathcal{P})$, using a solution to $(\mathcal{U})$ as a starting point.

input. The flowchart in Figure 2.4 details the heuristic for obtaining such a solution. For each subproblem ℓ, the heuristic enforces the nonlinear constraints of model $(\mathcal{P})$ and recalculates battery state-of-charge for each time period in the subproblem, using the battery power from the optimized dispatch decisions from model $(\mathcal{U})$. Then, the heuristic turns diesel generators on or off and adjusts their power output until the battery power input and output are feasible to $(\mathcal{P})$ for each period $t \in \mathcal{T}_\ell$.

2.5 Computational Results

In this section, we first briefly describe the collection of 14 problem instances from Scioletti et al. (2017). Then, we report computational results for these instances under the partitioning scheme for bilinear terms from Section 2.3 and the

decomposition algorithm of Section 2.4. The results: (i) demonstrate the benefit of Algorithm 1 by comparing the time to solve the McCormick relaxation model ($\mathcal{M}$) using our method, to that of solving model ($\mathcal{M}$) directly; (ii) exhibit the improvements due to our tightened linearization of bilinear terms by comparing results of our model ($\mathcal{U}$) and that of Gounaris et al. ($\mathcal{G}$); (iii) validate the use of a partitioning scheme by using solutions to ($\mathcal{U}$) as a starting point to obtain an upper bound on the nonlinear model ($\mathcal{P}$); and, (iv) detail the impact of the problem-specific methodology in Section 2.4.4 on solution quality by comparing the bounds on ($\mathcal{U}$) achieved under subsets of these ideas. For all MIP instances, we used CPLEX v. 12.6.2.0 (IBM 2017) to either solve the model directly or to solve subproblems associated with our decomposition algorithm. Our implementation of the decomposition algorithm is in Python 2.7.9 (Rossum 1995), and all results were obtained on a Cray XC40 compute node with two Intel E5-2690 v3 12-core (Haswell) processors and 64 GB of DDR4 memory.

2.5.1 Load and Candidate Design Technologies

Forward Operating Bases (FOBs) serve as protected locations from which to project and sustain combat operations, require significant logistical support, and must produce electrical power without a connection to the grid. While FOBs, or, in general, the population and land mass of any remote site, vary greatly in size and location, we obtained data for remote locales with no more than 150 soldiers on a 100m by 100m area, with peak power load of at most 300kW over an operating time horizon of one year. This section outlines the source of load and candidate storage and generation

assets for our model instances.

The FOB demand profiles were simulated in EnergyPlus (2001), based on an experiment conducted at the Basecamp Integration Lab at Fort Devens, MA. The four diesel generator technologies we consider are currently available in the military inventory, and have power ratings from 15kW to 100kW, with purchase costs between \$25K and \$38K. The lone type of PV system we consider is assumed to have fixed-tilt-and-angle panels with a nameplate rating of 1kW; we assume that spatial restrictions impose a limit of 75 such systems, at a purchase cost of \$2/W. The battery technologies that we consider are sourced from A123 (2018) and have a purchase cost of \$500/kWh, with a minimum of one hour for a full discharge and three hours for a full charge.

2.5.2 Solution of Model ($\mathcal{M}$) via Decomposition

We demonstrate the benefit of using Algorithm 1 to solve model ($\mathcal{M}$), over solving ($\mathcal{M}$) directly. Table 2.1 displays solution times and optimality gaps for our 14 test problems; the decomposition procedure outperforms direct solution in each instance, decreasing the solution time from a minimum of 56 minutes to a maximum of six minutes.

2.5.3 Solution of Model ($\mathcal{U}$) vs. Model ($\mathcal{G}$)

We illustrate the impact of our tighter mixed integer linear relaxation, ($\mathcal{U}$), compared to that of Gounaris et al. (2009), ($\mathcal{G}$), by using Algorithm 1 to solve each formulation over the 14 instances. Proposition 2.3.1 shows that these problems share

	Algorithm 1, $\lvert\mathcal{L}\rvert$=365				Direct Solution	
	$\underline{z}^M$	$\bar{z}^M$	Gap	Solve Time	Gap	Solve Time
Instance	**($MM)**	**($MM)**	**(%)**	**(seconds)**	**(%)**	**(seconds)**
Bagram	1.961	2.031	3.53	285	4.96	15333
Bamako	1.016	1.050	3.41	52	4.40	8684
Brazzaville	1.241	1.249	0.70	44	4.06	7155
Buenos Aires	1.591	1.631	2.50	94	6.67	18000
Dili	1.438	1.448	0.71	66	4.17	9273
Dushanbe	2.113	2.166	2.52	328	8.08	18000
Boston	3.401	3.466	1.92	222	7.43	18000
Gengneung	2.520	2.575	2.15	195	6.81	18000
Istanbul	2.157	2.200	2.01	154	8.39	18000
Kuwait City	1.627	1.702	4.59	146	10.13	18000
Mexico City	1.127	1.157	2.66	119	9.61	18000
San Salvador	0.967	0.974	0.76	52	4.53	3333
Tallinn	2.567	2.620	2.08	190	9.95	18000
Springfield	3.885	3.978	2.38	245	4.97	11746

Table 2.1: Computational results for a collection of year-long ($|\mathcal{T}| = 8760$) instances of model ($\mathcal{M}$). Models are solved using CPLEX v. 12.6.2.0, via Python 2.7.9.
• Termination criterion for "Algorithm 1" and "Direct Solution": min{time limit $\leq$ 5 hours, optimality gap $\leq$ 5%}.
• Termination criterion per subproblem: min{time limit $\leq$ 60 seconds, optimality gap $\leq$ 0.5%}; the former time limit was reached in fewer than 1% of cases.

the same set of feasible solutions when the binary restrictions in constraint (2.8h) hold; however, when this constraint is relaxed, the tighter model ($\mathcal{U}$) allows us to obtain high-quality lower bounds more quickly. Table 2.2 compares the solution times from applying Algorithm 1 to models ($\mathcal{U}$) and ($\mathcal{G}$), using the same tolerance of a 5% optimality gap. These results use $|\mathcal{N}| = 4$ uniform subregions, and we discuss

this choice of $|\mathcal{N}|$ in Section 2.5.4. Using our tighter relaxation reduces solution times by about 20% on average.

	Time to reach 5% tolerance (seconds)	
Instance	$(\mathcal{U})$	$(\mathcal{G})$
Bagram	671	882
Bamako	438	451
Brazzaville	235	260
Buenos Aires	318	339
Dili	358	365
Dushanbe	827	1120
Boston	453	532
Gengneung	871	1375
Istanbul	257	463
Kuwait City	510	520
Mexico City	318	387
San Salvador	221	224
Tallinn	261	305
Springfield	585	652

Table 2.2: Computational results of using Algorithm 1 to approximately solve instances of models $(\mathcal{U})$ and $(\mathcal{G})$ ($|\mathcal{T}|$ = 8,760 hrs, $|\mathcal{L}| = 365$), using the partitioning approach in constraints (2.8) and the one developed by Gounaris et al. (2009), with $|\mathcal{N}| = 4$ uniform subregions in both approaches.
Models are solved using CPLEX v. 12.6.2.0, via Python 2.7.9.
• Termination criterion for Algorithm 1: optimality gap $\leq 5\%$.
• Termination criterion per subproblem: min{time limit $\leq$ 60 seconds, optimality gap $\leq 0.5\%$}; the former time limit was reached in fewer than 5% of cases.

2.5.4 Solution of Model $(\mathcal{P})$

To obtain a feasible solution to the MINLP model $(\mathcal{P})$, we iteratively solve models $(\underline{\mathcal{U}})$ and $(\bar{\mathcal{U}})$, and then use the procedure in Figure 2.4 to obtain solutions

to model ($\bar{\mathcal{P}}$). Table 2.3 displays the solution times and optimality gaps achieved on model ($\mathcal{P}$).

We obtain solutions within 5% of optimality within 20 minutes for all 14 MINLP instances, a significant result for the nonlinear microgrid design and dispatch problem using this set of technologies on year-long instances. For comparison, Scioletti et al. (2017) were unable to achieve 5% optimality gaps for day-long instances ($|\mathcal{T}| = 24$ hours) in the 14 locations within three hours, using any of the general-purpose MINLP solvers BARON (Sahinidis 1996), Bonmin (Bonami and Lee 2009), or Couenne (Belotti 2009), and for nine of the instances, they fail to obtain a feasible solution within three hours.

We compare the performance related to different values for $|\mathcal{N}|$ in the partitioning scheme we use to create model ($\mathcal{U}$) by adopting two variants of the performance profile developed by Dolan and Moré (2002). In the first variant, we compare the optimality gaps achieved for a one-hour time limit as the number of subregions in our partitioning scheme varies. Let g_{ps} be the optimality gap achieved in one hour of runtime using p subregions for instance s. Then, our performance ratio, r_{ps}, is defined as

$$r_{ps} = \frac{g_{ps}}{\min\limits_{p \in \{1,\ldots,6\}} \{g_{ps}\}}. \tag{2.12}$$

Figure 2.5 displays the performance profile of one to six subregions as an estimate of the cumulative distribution function of the performance ratio, using the 14 instances from our application as samples. The x-axis in Figure 2.5 specifies a multiplicative factor on the best optimality gap. So, with $|\mathcal{N}| = 4$ subregions, we can solve 13 out

Instance	$\underline{z}^*$ ($MM)	$\bar{z}^*$ ($MM)	Gap (%)	Solve Time (seconds)
Bagram	1.963	2.062	5.00	877
Bamako	1.017	1.061	4.29	883
Brazzaville	1.244	1.268	1.92	443
Buenos Aires	1.594	1.651	3.63	522
Dili	1.442	1.466	1.69	563
Dushanbe	2.114	2.190	3.58	1172
Boston	3.401	3.486	2.50	651
Gengneung	2.521	2.585	2.53	1078
Istanbul	2.157	2.219	2.87	266
Kuwait City	1.629	1.698	4.29	837
Mexico City	1.129	1.176	4.20	525
San Salvador	0.969	0.992	2.31	426
Tallinn	2.568	2.635	2.62	476
Springfield	3.886	3.995	2.80	791

Table 2.3: Computational results of approximately solving MINLP model ($\mathcal{P}$) ($|\mathcal{T}|$ = 8,760 hrs, $|\mathcal{L}| = 365$).
Models ($\bar{\mathcal{U}}$) and ($\mathcal{U}$) are solved with $|\mathcal{N}| = 4$ uniform subregions using CPLEX v. 12.6.2.0, via Python 2.7.9. The second column ($\underline{z}^*$) reports the optimal value of ($\mathcal{U}$). The third column ($\bar{z}^*$) reports the objective function value of the feasible solution to model ($\bar{\mathcal{P}}$) obtained by adjusting solutions to model ($\bar{\mathcal{U}}$) via the procedure in Figure 2.4.
• Termination criterion for Algorithm 1: ($\mathcal{U}$) optimality gap $\leq 5\%$.
• Termination criterion per subproblem: min{time limit $\leq$ 60 seconds, optimality gap $\leq 0.5\%$}; the former time limit was reached in fewer than 5% of cases.

of 14 instances within a factor of 1.05 of the best gap obtained by all six procedures.

In our second variant of the performance profile from Dolan and Moré (2002), we report the optimality gaps achieved by each partitioning scheme as a function of time in the following manner. Let g_{pst} be the optimality gap achieved in t seconds of runtime using p subregions to solve instance s. Then, our performance ratio at time

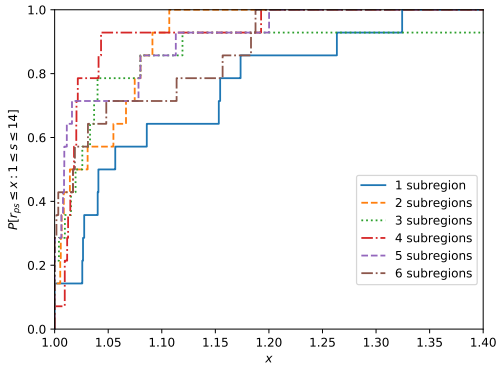

Figure 2.5: Performance profile for the partitioning scheme of model ($\mathcal{U}$) in which the number of subregions, $|\mathcal{N}|$, varies between one and six, for our 14 instances. Here, we solve models ($\mathcal{U}$) and ($\bar{\mathcal{U}}$) via Algorithm 1, and then use the procedure of Figure 2.4 to obtain solutions to ($\bar{\mathcal{P}}$). The value r_{ps} is calculated using equation (2.12). For $|\mathcal{N}| = 4$, the performance profile reaches the value of 1.0 at $x = 1.193$, meaning that the procedure with $|\mathcal{N}| = 4$ achieves a gap within 1.193 times the best gap for all six procedures (i.e., using $|\mathcal{N}| = 1, 2, \ldots, 6$) across all 14 instances.

t, r_{pst}, is defined as

$$r_{pst} = \frac{\min\limits_{p \in \{1,\ldots,6\}} \{g_{ps\tau}\}}{g_{pst}}, \tag{2.13}$$

in which $\tau = 3,600$ seconds, the time limit for each instance.

Unlike the performance measure of equation (2.12), the measure in equation (2.13) tracks the optimality gap as a function of time. Both measures are constructed so that values "higher and to the left" indicate superior performance. Figure 2.6 displays the geometric and arithmetic means of r_{pst} across the 14 instances. While the partitioning scheme exhibits the tightest optimality gaps using $|\mathcal{N}| = 4$ subregions for our collection of instances with a one-hour time limit, the results suggest that $|\mathcal{N}| = 2$ subregions may provide tighter gaps under a stricter computational budget.

Table 2.4 details the impact of the problem-specific methodology we describe in

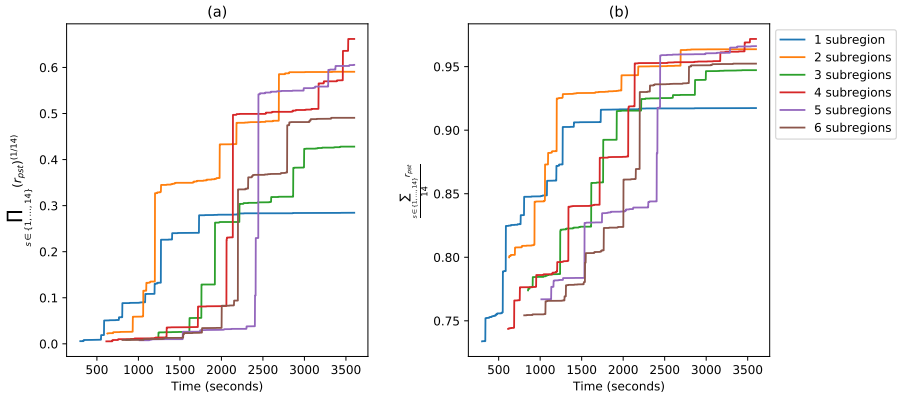

Figure 2.6: Performance profile of the metric r_{pst} over time for our partitioning scheme in which the number of subregions, $|\mathcal{N}|$, varies between one and six, for the 14 instances in our application. The value r_{pst} is calculated using equation (2.13). Part (a) and part (b) display the geometric and arithmetic means of r_{pst}, respectively. Here, we solve models ($\underline{\mathcal{U}}$) and ($\bar{\mathcal{U}}$) via Algorithm 1, and then use the procedure of Figure 2.4 to obtain solutions to ($\bar{\mathcal{P}}$).

Section 2.4.4, applied to our 14 instances. The addition of a constraint that provides a lower bound on diesel generator capacity significantly reduces solution times when using Algorithm 1 to solve the nonlinear microgrid design and dispatch problem.

2.6 Conclusions

We present a methodology that takes advantage of problems that are loosely coupled with respect to time and contain some bilinear terms. We present a partitioning scheme that approximates constraints containing bilinear terms with a tighter relaxation than similar approaches in the literature. We present a decomposition method that separates the problem into smaller subproblems, using Lagrangian relaxation to obtain a lower bound and fixing inventory levels at regular intervals to obtain an upper bound. We apply the methodology to a microgrid design and dispatch problem, solve instances to within 5% of MIP optimality in at most six minutes,

	$(\bar{\mathcal{P}})$ and $(\underline{\mathcal{U}})$ Only		Add Minimum Generator Capacity	
Instance	**Gap (%)**	**Solve Time (seconds)**	**Gap (%)**	**Solve Time (seconds)**
Bagram	17.77	7200	5.00	877
Bamako	5.51	7200	4.29	883
Brazzaville	2.25	325	1.92	443
Buenos Aires	5.74	7200	3.63	522
Dili	1.75	524	1.69	563
Dushanbe	5.36	7200	3.58	1172
Boston	3.98	7184	2.50	651
Gangneung	3.73	620	2.53	1078
Istanbul	4.26	2824	2.87	266
Kuwait City	5.28	7200	4.29	837
Mexico City	4.16	2021	4.20	525
San Salvador	2.61	469	2.31	426
Tallinn	4.34	3570	2.62	476
Springfield	5.57	7200	2.80	791

Table 2.4: Computational results of approximately solving instances of MINLP model $(\mathcal{P})$ ($|\mathcal{T}| = 8{,}760$ hrs, $|\mathcal{L}| = 365$). Models $(\underline{\mathcal{U}})$ and $(\underline{\mathcal{U}})$ are solved with $|\mathcal{N}| = 4$ uniform subregions using CPLEX v. 12.6.2.0, via Python 2.7.9. Solutions to model $(\bar{\mathcal{P}})$ were obtained by adjusting solutions to model $(\underline{\mathcal{U}})$ via the procedure in Figure 2.4.
• Termination criterion for Algorithm 1: min{time limit $\leq$ 2 hours, $(\mathcal{P})$ optimality gap $\leq$ 5%}.
• Termination criterion per subproblem: min{time limit $\leq$ 60 seconds, optimality gap $\leq$ 0.5%}; the former time limit was reached in fewer than 5% of cases.

and achieve a 5% MINLP gap for all instances within 20 minutes, which significantly improves on alternative approaches.

Chapter 3

Remote Microgrid Design Optimization Under Photovoltaic And Load Uncertainty

3.1 Introduction

The United States military constructs forward operating bases (FOBs) to serve as protected locations from which to project combat power and sustain combat operations in remote regions. While most bases employ spot generation, in which a diesel generator is attached directly to a load, microgrids that combine these individual loads and augment generators with photovoltaic (PV) cells and batteries as a distributed energy system are emerging as a safer and more efficient alternative to provide the energy needs of an entire FOB. Scioletti et al. (2017) present a model that seeks the minimum-cost microgrid design and ideal dispatched power to support a FOB for one year with hourly fidelity under a detailed battery model; this mixed-integer nonlinear program (MINLP) is intractable with commercial solvers but loosely coupled with respect to time. A mixed-integer linear program (MIP) approximates the model, and McCormick envelopes (McCormick 1976) linearize the bilinear terms. In Section 2.4, we introduce a policy for loosely coupled MIPs in which the system reverts to equivalent conditions at regular time intervals; this separates the problem into subproblems that we solve in parallel to obtain upper and lower bounds on the model's optimal objective function value.

We extend the model from Scioletti et al. (2017) to account for uncertainty in the FOB's load and available PV resources. We develop an approach to simulate the occupancy of each building on the FOB, which we then use to obtain sample paths of environmental control unit (ECU) loads; these realizations share common, observed weather inputs with the model we use to generate realizations of PV power output by technology. We develop models that provide lower and upper bounds on the new model's optimal value using the method in Section 2.4. Finally, we solve these instances using the sample average approximation technique from Mak et al. (1999), and compare the quality of the solutions to those obtained by optimizing under point forecasts. Section 3.2 describes the FOB we use for our case study, and the method we use to generate realizations. Section 3.3 details a compact formulation of our optimization model and decomposition scheme. Section 3.4 assesses the quality of our solutions and compares the designs to those obtained by point forecasts of load and PV.

3.2 Load and PV Model

This section describes the method we use to generate realizations of the sources of uncertainty in our optimization model, specifically, the hourly FOB load and the hourly power output of our candidate solar PV technology. Section 3.2.1 describes the layout of the FOB we use for our case study and the buildings we model to obtain baseline data. Section 3.2.2 describes the procedure for simulating base-wide, and then building-level, hourly occupancy by personnel. Section 3.2.3 describes our sources for weather data, and our method for producing realizations of PV power

output by hour for the solar technology we consider. Section 3.2.4 summarizes how we use these data to obtain bivariate sample paths of FOB load and PV power output.

3.2.1 FOB Buildings

Gildea et al. (2017) provide simulation-based analyses with respect to fuel and water consumption, and waste reduction, for Technology Enabled Capability Demonstration (TECD) camps with varying personnel levels; we use the equipment and building lists from the 312-personnel camp for our case study. Table 3.1 summarizes the number and structure of each building type on the base for which an environmental control unit (ECU) provides heating and cooling to each building. The two structure types used in the 312-troop camp are (i) a $20' \times 8'$ military van (MILVAN) shelter, which has a capacity of eight personnel and a rigid structure, and (ii) a $20' \times 32'$ AirBeam shelter (HDT Global 2016b), which is soft-walled and has a maximum capacity of 24 personnel. We create renderings of these two building types via OpenStudio, a free-to-use modeling software suite developed by the National Renewable Energy Laboratory. Figure 3.1 compares these renderings to images of each building.

The collection of buildings in Table 3.1 differs from that described in Gildea et al. (2017) in two ways. First, we assume that the billeting structures are assigned an equal number of soldiers to that building's full capacity, and so we only use 20 such buildings instead of the 29 implemented in Gildea et al. (2017). Second, we restrict our analysis to the two building types above, and replace other structure types in Gildea et al. (2017) with the one closest in size and capacity of the two we implement. We assume that each building is equipped with a F100-60K 5-ton ECU

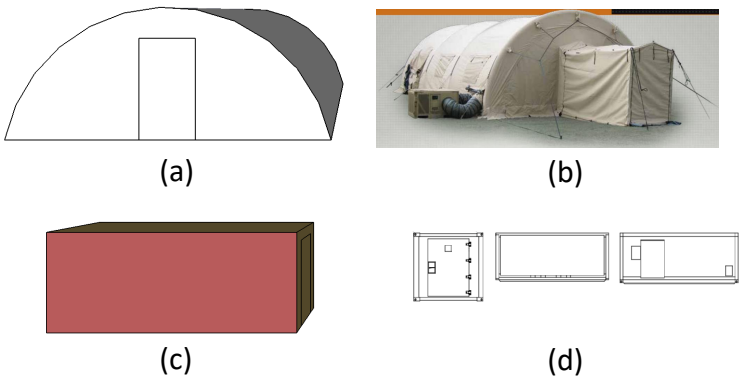

Figure 3.1: OpenStudio renderings of the MILVAN shelter and AirBeam tent that we assume compose all buildings on the FOB with ECUs. Images (a) and (b) display an OpenStudio rendering and picture (source: HDT Global 2016b) of an AirBeam tent, respectively. Images (c) and (d) display an OpenStudio rendering and the front, left and right side specifications (source: Department of Defense 2002) for a MILVAN shelter, respectively.

(HDT Global 2016a).

3.2.2 Occupancy Model

The occupancy model's starting point is the hourly schedule for each of the 312 soldiers on our FOB, which is used as input to the analyses developed by Gildea et al. (2017). Each soldier in the camp is assigned a predetermined hourly schedule for 30 days, which repeats over the course of our year-long operating time horizon. We assume: (i) groups of soldiers move together in units of 19-38 people; (ii) units leave the base for missions at the same time of day as given in the original schedule; (iii) the proportion of time that soldiers are off-base is the same as in the original schedule; and, (iv) the activity of every soldier on the base is known throughout the day. However, we allow the length of time that the soldiers are off-base performing missions to vary, and we also assume that different events and activities taking place

Building Type	**Number**	**Structure**
AAFES[a]	1	MILVAN[b]
Aid Station	1	MILVAN
Battalion TAC[c]	1	MILVAN
Billet	13	AirBeam
Changing Tent	1	AirBeam
Shower	2	AirBeam
Dining	2	AirBeam
Kitchen	1	MILVAN
Fires CP[d]	1	MILVAN
Maintenance MILVAN	1	MILVAN
Rifles CP	1	MILVAN
Supply Office	1	MILVAN
Transient (Airbeam)	3	AirBeam
Transient (MILVAN)	5	MILVAN
Recreation	1	MILVAN
VIP[e]	1	MILVAN

Table 3.1: Summary of the subset of buildings on the TECD 312-soldier camp that contain ECUs.
[a] Army and Air Force exchange service
[b] Military van
[c] Tactical action center
[d] Command post
[e] Very important person

on the base temporarily increase or decrease the FOB's occupancy. These assumptions allow us to preserve the mean number of soldiers on the base over time from the analysis in Gildea et al. (2017), but allow the variability in hourly occupancy to more closely mimic real-world operations than a rigid schedule does.

3.2.2.1 Schedule Variability

The hourly occupancy model is based on a 30-day schedule, which is repeated to construct an 8,760-hour itinerary for each individual soldier. In this nominal schedule, while a subset of the personnel remain on the base at all hours, most are members of 19-38 soldier units that leave the base simultaneously for exactly 11 hours, and then return to the base together. Additionally, two civil affairs soldiers and four medics are off-campus for half of all hours, and these personnel operate independently of any one unit. In the randomized schedule, the time that each of these six personnel leaves matches the original schedule, but the number of hours spent off-base is a randomly selected integer between 7 and 17, instead of a fixed 12-hour excursion. Similarly, each time a unit leaves for a mission, its duration outside the FOB varies between 4 and 18 hours with equal probability. These random variables introduce uncertainty into the base population at any point in time, and hence ECU loads, while preserving the proportion of time each soldier spends on the FOB in expectation.

3.2.2.2 Special Events

We further incorporate variability into the hourly base population by modeling several events that commonly occur on FOBs, but outside of the schedule described in Section 3.2.2.1, and with uncertain timing. These events allow for groups of soldiers to enter from outside the base as visitors, or for large numbers of personnel from the base population to leave the base together. While some of these events occur year-round, others only occur during periods of high operational tempo, or *op-tempo*. We

assume the duration of high op-tempo may be (i) the entire year, (ii) from February 15th through November 15th, or (iii) from April 1st through September 30th, each of which takes place with equal probability. Table 3.2 describes each event and the range of its frequency, duration, and change in FOB population. Appendix B.1 provides further details on the assumptions associated with each event listed in Table 3.2.

Increase or Decrease	Event	Timing	Frequency	Length	Range of Soldiers
Increase	Holiday	Thanksgiving, Christmas, Super Bowl	Fixed date	24 hours	{0,15,30,full population}
Increase	Rebuild	High op-tempo	0-2/year	5-9 days	[20,35]
Increase	Resupply	Year-round	2-5/month	5-48 hours	[50,100]
Increase	Troop turnover	Year-round	1-2/year	36-96 hours	[100,156]
Increase	Task force mission	High op-tempo	1 per 1-2 months	6-24 hours	[60,100]
Increase	Training	Year-round	2-3/year	3-7 days	[15,35]
Decrease	Fighting mission	High op-tempo	1/month	20-96 hours	[90,180]
Decrease	Leave and other	Year-round	Ongoing	Varies	{1}

Table 3.2: Summary of special events that impact the FOB population beyond the typical schedule.

3.2.2.3 Building Occupancy

After accounting for variability in the unit departure schedules in Section 3.2.2.1 and special events in Section 3.2.2.2, we can obtain realizations of hourly FOB occupancy for each of the 312 soldiers, as well as for the number of visitors from outside the base. Next, we determine the building-specific location of soldiers on-base for each hour. We assume a soldier may be in one of four states: (i) conducting support operations; (ii) conducting camp security detail; (iii) off-duty; or, (iv) in transition, which takes place during the first hour upon a soldier's return from a mission outside

the base. Support operations vary by job description, and so we specify the likelihood of occupying each building type on the FOB according to each soldier's job description while in state (i). We do not model building occupancy for this camp security detail, which takes place outdoors. We assume that soldiers either off-duty or in transition take part in similar activities, regardless of job description. Table 3.3 details the conditional probability of each soldier's building-specific location, given that soldier's state and job. If someone occupies any building on the FOB without an ECU, we treat this event the same as if the soldier were outside. All soldiers visiting from outside the base are outside with probability 0.5 and in a transient building with probability 0.5.

	Support Operations											
	Battalion TAC	Platoons	Fires and Battery	Engineers	Medical	Maintenance	Feeding	Company Support	Battery Support	Field Maintenance	Off-Duty	First Hour After Mission
AAFES											0.05	
Aid Station					0.8						0.05	0.05
Battalion TAC	0.8											
Billet		0.3		0.3							0.5	0.2
Shower											0.1	
Dining	0.1						0.4				0.1	
Kitchen							0.4					
Fires CP			0.4									0.05
Maintenance MILVAN						0.2				0.4		0.05
Rifles CP		0.1						0.5	0.4			0.1
Supply Office		0.05						0.2				0.05
Recreation											0.1	
VIP												
Outside	0.1	0.55	0.6	0.7	0.2	0.8	0.2	0.3	0.6	0.6	0.1	0.5

Table 3.3: Summary of conditional probabilities of an individual soldier's building occupancy, given their state and job description.

Figure 3.2 compares the original schedule from Gildea et al. (2017) to one with variability in the schedule and special events added, and shows that adding these elements affects both the variability and maximum population of the FOB.

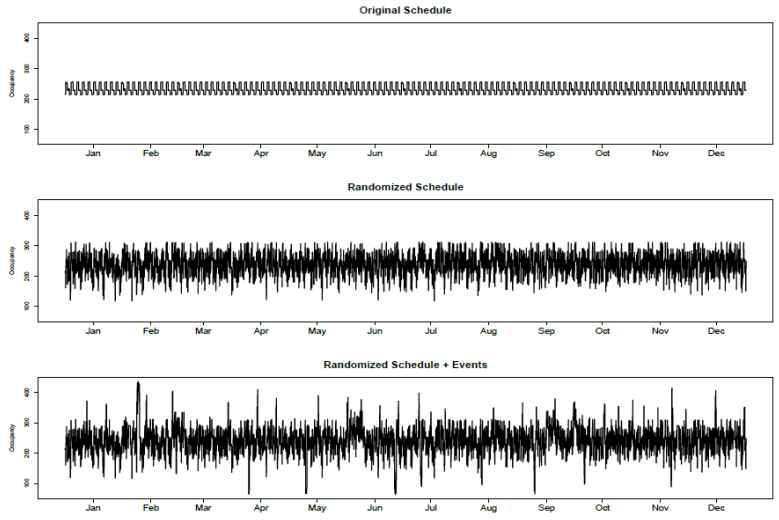

Figure 3.2: Comparison of the baseline occupancy model to a sample path of a model that incorporates a randomized schedule, and separate sample path that includes both a randomized schedule and special events.

3.2.3 PV Model and Weather Data

While several different methods exist for forecasting PV power output (see, e.g., Dolara et al. 2015, Antonanzas et al. 2016), we develop a collection of year-long weather datasets using historical sources, and select one at random to serve as input to our PV and load models. We obtain estimates using PVWatts, a physics-based software tool that provides the PV power output for a system at hourly fidelity, using the technology's characteristics (e.g., nameplate DC rating, tilt, azimuth, and cell material) and a weather dataset in typical meteorological year (TMY) format as input; see, e.g., Blair et al. (2014) for more details.

We use a TMY file for our chosen location developed by ASHRAE (2002) as a starting point. While these files contain hourly observations for all the weather characteristics required by PVWatts, the files available to the public consist of a single

year of data in aggregate. We use two separate sources of hourly weather data to supplant key characteristics in the TMY files. NASA's modern-era retrospective analysis for research and applications (MERRA) provides hourly historical data for temperature, wind speed and direction, humidity, and atmospheric pressure (Rienecker et al. 2011, Gelaro et al. 2017), while Copernicus Atmosphere Monitoring Service (CAMS) provides historical data for global, direct normal, and diffuse solar radiation by hour for analogous timeframes and locations (Schroedter-Homscheidt et al. 2016). CAMS and MERRA observations are available from 2004 forward, so we use the hourly observations from both sources to generate 14 unique weather datasets by editing the subset of data fields of the TMY file that we can source from CAMS or MERRA with the appropriate data source for each year. We note that some characteristics, such as infrared radiation, are only available in the TMY file, and so these data are identical for each of the 14 year-long observations.

To obtain a PV power output sample path, we use one of the modified TMY files that include observational data from CAMS and MERRA as our weather source, and enter the PV output characteristics from our candidate technology as input to PVWatts, which yields a realization of power output for the year-long weather file at hourly fidelity.

3.2.4 Scenario Generation

Figure 3.3 displays an overview of the procedure used to obtain bivariate load and PV power output sample paths. As a preprocessing step, a building-level ECU load file is created for each building type in Table 3.1, for each (constant) occupancy

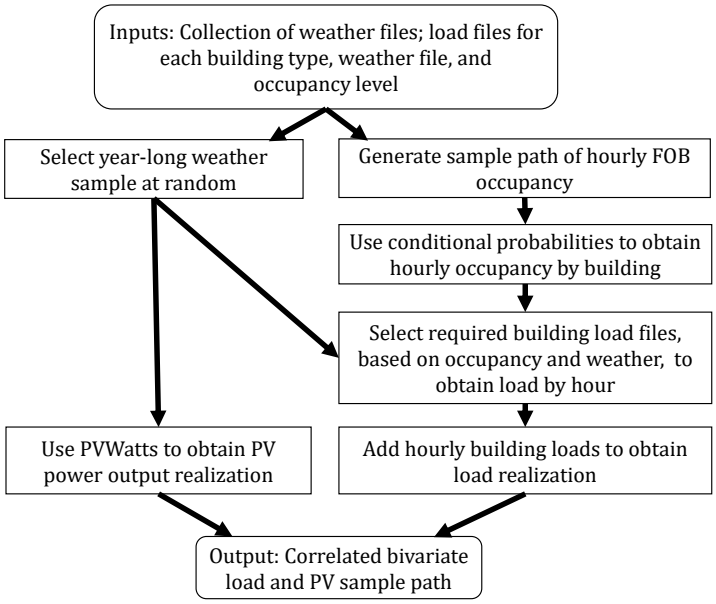

Figure 3.3: Overview of the procedure used to obtain bivariate load and PV power output sample paths.

level, and for each weather file generated by the procedure in Section 3.2.3. Then, for each sample path, we: (i) select a weather file at random; (ii) use the procedure in Section 3.2.2 to obtain building-specific occupancy on the FOB for each hour; (iii) look up the appropriate ECU load for each building's occupancy level and type at each hour; (iv) aggregate the ECU and other loads into a single quantity for each hour; and, (v) use PVWatts to obtain PV power output from the same weather file.

3.3 Optimization Model

The section details our adaptation of the optimization model from Scioletti et al. (2017) for our application. Similar to the model we describe in Section 2.2.1, our baseline formulation is written in a compact form with constructs that highlight

the temporal dependencies, but here we also focus on the stochastic elements of the model. We refer to this stochastic programming model with recourse as ($\mathcal{SP}$).

3.3.1 ($\mathcal{SP}$) Formulation

Sets

$\omega \in \Omega$	scenarios
$t \in \mathcal{T} = \{1, \ldots, \lvert\mathcal{T}\rvert\}$	time periods
$\ell \in \mathcal{L} = \{1, \ldots, \lvert\mathcal{L}\rvert\}$	time blocks indexing a partition of $\mathcal{T}$; i.e., $\cup_{\ell \in \mathcal{L}} \mathcal{T}_\ell = \mathcal{T}$ and $\mathcal{T}_\ell \cap \mathcal{T}_{\ell'} = \emptyset$, $\ell \neq \ell'$
$\mathcal{T}_\ell \subset \mathcal{T}$	time periods in block ℓ
$\mathcal{T}_\ell^-$	set of time periods in block ℓ, excluding the first period
$X \in \mathcal{X}$	feasible design decisions
$Y_t^\omega \in \mathcal{Y}_t^\omega(X, S_t^\omega)$	feasible dispatch decisions made in scenario ω, time period t, given decision X and shortfall level S_t^ω

Functions

$f_0(\cdot)$	cost of a design decision
$f_t(\cdot)$	cost of a dispatch decision in time period t
$g_t(\cdot)$	net change in energy storage (inventory) associated with a dispatch decision in time period t
$h_t(\cdot)$	cost of shortfall in time period t

Parameters

υ	number of time periods per block
p^ω	probability associated with scenario ω

With these constructs, we have:

$$\mathcal{T}_\ell = \{(\ell - 1)\upsilon + 1, (\ell - 1)\upsilon + 2, \ldots, \ell\upsilon\}, \ \forall \ell \in \mathcal{L}.$$

Decision Variables

X	strategic design decision
R	strategic inventory reset value
X_ℓ^ω	strategic design decision specific to scenario ω, block ℓ
R_ℓ^ω	strategic inventory reset value specific to scenario ω, block ℓ
Y_t^ω	operational dispatch decision at time period t in scenario ω; $Y = (Y_t^\omega)_{\omega\in\Omega, t\in\mathcal{T}}$
$\underline{Y}_t^\omega$	inventory at start of time period t in scenario ω; $\underline{Y} = (\underline{Y}_t^\omega)_{\omega\in\Omega, t\in\mathcal{T}}$
$\bar{Y}_t^\omega$	inventory at end of time period t in scenario ω; $\bar{Y} = (\bar{Y}_t^\omega)_{\omega\in\Omega, t\in\mathcal{T}}$
S_t^ω	shortfall in scenario ω, time period t

Boundary Condition

Y_0	initial inventory

($\mathcal{SP}$) Formulation

$$z^{SP} = \min_{X, X_\ell^\omega, R, R_\ell^\omega, Y, \underline{Y}, \bar{Y}, S} \quad f_0(X) + \sum_{\omega\in\Omega} p^\omega \left(\sum_{\ell\in\mathcal{L}} \sum_{t\in\mathcal{T}_\ell} (f_t(Y_t^\omega) + h_t(S_t^\omega)) \right) \tag{3.1a}$$

$$\text{s.t.} \quad X_\ell^\omega \in \mathcal{X}, \quad \forall \ell \in \mathcal{L}, \omega \in \Omega \tag{3.1b}$$

$$Y_t^\omega \in \mathcal{Y}_t^\omega(X_\ell^\omega, S_t^\omega), \quad \forall t \in \mathcal{T}, \omega \in \Omega \tag{3.1c}$$

$$\bar{Y}_t^\omega = \underline{Y}_t^\omega + g_t(Y_t^\omega), \quad \forall t \in \mathcal{T}, \omega \in \Omega \tag{3.1d}$$

$$\underline{Y}_t^\omega = \bar{Y}_{t-1}^\omega, \quad \forall t \in \mathcal{T}_\ell^-, \ell \in \mathcal{L}, \omega \in \Omega \tag{3.1e}$$

$$\underline{Y}_t^\omega = R_\ell^\omega, \quad \forall t = (\ell-1)\upsilon + 1, \ell \in \mathcal{L} \setminus \{1\}, \omega \in \Omega \tag{3.1f}$$

$$X_\ell^\omega = X, \quad \forall \ell \in \mathcal{L}, \omega \in \Omega : \mu_\ell^\omega \tag{3.1g}$$

$$R_\ell^\omega = R, \quad \forall \ell \in \mathcal{L}, \omega \in \Omega : \theta_\ell^\omega \tag{3.1h}$$

$$\underline{Y}_1^{\omega} = Y_0, \quad \forall \omega \in \Omega. \tag{3.1i}$$

We seek, via the objective function in (3.1a), a minimum-cost set of design and dispatch decisions. This objective includes the cost of procuring technologies, using fuel, degrading generators and batteries, and shortfall, i.e., unmet load. Constraint (3.1b) specifies feasible microgrid designs, as defined by the set $\mathcal{X}$. Constraint (3.1c) restricts dispatch decisions Y_t^{ω} to those allowable by the design decision X and shortfall level S_t^{ω} at each time period t and scenario ω. For example, only an asset purchased as part of a design decision may be operated at time period t, and any unmet load must be accounted for as shortfall. The latter condition yields a model with relatively complete recourse; that is, designs that are unable to meet demand in every time period are allowable. Without detailing the reformulation here, we address all nonlinear constraints within (3.1c) by using the McCormick underestimators from McCormick (1976) and from model ($\mathcal{M}$) in Section 2.2.2. Constraints (3.1d)-(3.1i) are analogous to constraints (2.9e)-(2.9j), but are now indexed on scenario ω in addition to time period t.

With the exception of constraints (3.1g) and (3.1h), model ($\mathcal{SP}$) would decompose into $|\mathcal{L}| \cdot |\Omega|$ separate problems that we could solve in parallel. In Sections 3.3.2 and 3.3.3, we develop models that address these constraints to obtain lower and upper bounds on model ($\mathcal{SP}$)'s optimal value.

3.3.2 ($\underline{\mathcal{SP}}$) Formulation: Lower Bounds

To obtain lower bounds on the optimal value of model ($\mathcal{SP}$), we develop a Lagrangian relaxation of the constraints (3.1g)-(3.1h) using the dual variables for the nonanticipativity constraints, μ_ℓ^ω and θ_ℓ^ω.

($\underline{\mathcal{SP}}$) Formulation

$$
\begin{aligned}
\underline{z}^{SP} = \min_{X_\ell^\omega, R_\ell^\omega, Y, \underline{Y}, \bar{Y}, S} \quad & \sum_{\omega\in\Omega} p^\omega \left(\frac{1}{|\mathcal{L}|} \sum_{\ell\in\mathcal{L}} f_0(X_\ell^\omega) + \sum_{\ell\in\mathcal{L}} \sum_{t\in\mathcal{T}_\ell} (f_t(Y_t^\omega) + h_t(S_t^\omega)) \right) \\
& + \sum_{\omega\in\Omega} \sum_{\ell\in\mathcal{L}} (\mu_\ell^\omega X_\ell^\omega + \theta_\ell^\omega R_\ell^\omega) && \text{(3.2a)} \\
\text{s.t.} \quad & X_\ell^\omega \in \mathcal{X}, \quad \forall \ell \in \mathcal{L}, \omega \in \Omega && \text{(3.2b)} \\
& Y_t^\omega \in \mathcal{Y}_t^\omega(X_\ell^\omega, S_t^\omega), \quad \forall t \in \mathcal{T}, \omega \in \Omega && \text{(3.2c)} \\
& \bar{Y}_t^\omega = \underline{Y}_t^\omega + g_t(Y_t^\omega), \quad \forall t \in \mathcal{T}, \omega \in \Omega && \text{(3.2d)} \\
& \underline{Y}_t^\omega = \bar{Y}_{t-1}^\omega, \quad \forall t \in \mathcal{T}_\ell^-, \ell \in \mathcal{L}, \omega \in \Omega && \text{(3.2e)} \\
& \underline{Y}_t^\omega = R_\ell^\omega, \quad \forall t = (\ell-1)\upsilon + 1, \ell \in \mathcal{L} \setminus \{1\}, \omega \in \Omega && \text{(3.2f)} \\
& \underline{Y}_1^\omega = Y_0, \quad \forall \omega \in \Omega. && \text{(3.2g)}
\end{aligned}
$$

Similar to model (2.10), we require $\sum_{\ell\in\mathcal{L}} \sum_{\omega\in\Omega} \mu_\ell^\omega = 0$ and $\sum_{\ell\in\mathcal{L}} \sum_{\omega\in\Omega} \theta_\ell^\omega = 0$ to avoid an unbounded Lagrangian relaxation, which eliminates the free variables X and R. The boundary conditions (3.2f) connecting time periods in multiple blocks now have separate storage reset levels in the relaxation, which allows model ($\underline{\mathcal{SP}}$) to separate into $|\mathcal{L}| \cdot |\Omega|$ subproblems, one for each time block, for each scenario.

3.3.3 ($\overline{\mathcal{SP}}$) Formulation: Upper Bounds

Similar to the procedure in Section 2.4.2, we develop a restriction of model ($\mathcal{SP}$) in which we fix a design decision, $\hat{X} \in \mathcal{X}$, and a common energy storage level to which the system must revert at the boundaries of each time block in each scenario; we refer to this storage level as $\hat{R}$.

($\overline{\mathcal{SP}}$) Formulation

$$\bar{z}^{SP} = \min_{Y,\underline{Y},\bar{Y},S} \quad f_0(\hat{X}) + \sum_{\omega\in\Omega} p^\omega \left(\sum_{\ell\in\mathcal{L}} \sum_{t\in\mathcal{T}_\ell} (f_t(Y_t^\omega) + h_t(S_t^\omega)) \right) \tag{3.3a}$$

$$Y_t^\omega \in \mathcal{Y}_t^\omega(\hat{X}, S_t^\omega), \quad \forall t \in \mathcal{T}, \omega \in \Omega \tag{3.3b}$$

$$\bar{Y}_t^\omega = \underline{Y}_t^\omega + g_t(Y_t^\omega), \quad \forall t \in \mathcal{T}, \omega \in \Omega \tag{3.3c}$$

$$\underline{Y}_t^\omega = \bar{Y}_{t-1}^\omega, \quad \forall t \in \mathcal{T}_\ell^-, \ell \in \mathcal{L}, \omega \in \Omega \tag{3.3d}$$

$$\underline{Y}_t^\omega = \hat{R}, \quad \forall t = (\ell-1)\upsilon + 1, \ell \in \mathcal{L} \setminus \{1\}, \omega \in \Omega \tag{3.3e}$$

$$\underline{Y}_1^\omega = Y_0, \quad \forall \omega \in \Omega. \tag{3.3f}$$

Constraints (3.3b) and (3.3e) are analogous to constraints (3.1c) and (3.1f), but use the fixed design, $\hat{X}$, and the fixed energy storage reset level, $\hat{R}$. These fixed variables allow for the removal of constraints (3.1b), (3.1g), and (3.1h), and they allow model ($\overline{\mathcal{SP}}$) to separate by $\ell \in \mathcal{L}$, and by $\omega \in \Omega$, into $|\mathcal{L}| \cdot |\Omega|$ subproblems that we may solve in parallel.

3.3.4 Decomposition Algorithm

Algorithm 2 provides an ε-optimal solution to model ($\mathcal{SP}$) by iteratively solving model ($\underline{\mathcal{SP}}$) and model ($\overline{\mathcal{SP}}$) to obtain lower and upper bounds, respectively, and

tightening these bounds as the algorithm proceeds. Similar to Algorithm 1, the procedure is a variant of the progressive hedging algorithm by Rockafellar and Wets (1991).

3.3.5 Scenario Pairing

The problem-specific adjustments in Section 2.4.4 include a setting a minimum generator capacity that tightens the lower bound. We do not include this adjustment, because model (3.2) allows relatively complete recourse. Instead, to tighten the lower bound provided by model ($\underline{\mathcal{SP}}$), we select the time block and scenario pairing with the highest peak demand, which we call $\hat{\ell}$ and $\hat{\omega}$, respectively, and we create a subproblem for all (ℓ, ω)-$(\hat{\ell}, \hat{\omega})$ pairs, $\ell \in \mathcal{L} \setminus \{\hat{\ell}\}$, $\omega \in \Omega \setminus \{\hat{\omega}\}$. Then, we include a copy of constraints from block $\hat{\ell}$ in scenario $\hat{\omega}$ to each of the $(|\mathcal{L}| \cdot |\Omega| - 1)$ subproblems. We add the terms in objective function (3.2a) specific to block $\hat{\ell}$ in scenario $\hat{\omega}$ to each subproblem's objective function, and we multiply each variable's coefficient by $1/(|\mathcal{L}| \cdot |\Omega| - 1)$. In Algorithm 2, we solve ($\underline{\mathcal{SP}}$) by solving the $(|\mathcal{L}| \cdot |\Omega| - 1)$ subproblems, which may be done in parallel. Model (3.4) details the formulation for a general block ℓ and scenario ω.

Additional Parameters

$(\hat{\ell}, \hat{\omega})$: time block-scenario pair with maximum peak demand

Subproblem Formulation

$$z_\ell^\omega = \min_{X_\ell^\omega, R_\ell^\omega, Y, \underline{Y}, \bar{Y}, S} \; p^\omega \left(\frac{1}{|\mathcal{L}|} f_0(X_\ell^\omega) + \sum_{t \in \mathcal{T}_\ell} (f_t(Y_t^\omega) + h_t(S_t^\omega)) \right) + \mu_\ell^\omega X_\ell^\omega + \theta_\ell^\omega R_\ell^\omega$$
$$+ \frac{p^{\hat{\omega}}}{|\mathcal{L}| \cdot |\Omega| - 1} \left(\frac{1}{|\mathcal{L}|} f_0(X_\ell^\omega) + \sum_{t \in \mathcal{T}_{\hat{\ell}}} (f_t(Y_t^{\hat{\omega}}) + h_t(S_t^{\hat{\omega}})) \right) \quad (3.4a)$$

Algorithm 2 Decomposition procedure to approximately solve model $(\mathcal{SP})$

procedure DECOMPOSITION

Inputs: $\varepsilon > 0, \rho_\mu > 0, \rho_\theta > 0, k \in \mathbb{Z}_+, \kappa \in \mathbb{Z}_+$ ▷ stopping criterion, ▷ proximal weights, upper bound search frequency, upper bound search depth,

Outputs: $X^*, R^*, \bar{z}, \underline{z}$ ▷ ε-optimal design and inventory, ▷ upper and lower bounds on z^{SP}

$i \leftarrow 0;\ \underline{z} \leftarrow -\infty;\ \bar{z} \leftarrow \infty$ ▷ iteration and initial lower and upper bounds

$\mu_\ell^\omega \leftarrow 0,\ \theta_\ell^\omega \leftarrow 0,\ \forall \ell \in \mathcal{L},\ \omega \in \Omega$ ▷ initial Lagrange multipliers ▷ for model $(\underline{\mathcal{SP}})$

while $\bar{z} - \underline{z} > \varepsilon \bar{z}$ **do**

Solve $(\underline{\mathcal{SP}})$ with $\mu_\ell^\omega, \theta_\ell^\omega$ to obtain $\underline{z}_i, X_\ell^\omega, R_\ell^\omega,\ \ell \in \mathcal{L}, \omega \in \Omega$

▷ $\underline{z}_i$ is a lower bound for z^{SP}

▷ $(|\mathcal{L}| \cdot |\Omega|)$ subproblems in $(\underline{\mathcal{SP}})$ can be solved in parallel

$\bar{X} \leftarrow \sum_{\omega \in \Omega} p^\omega \frac{1}{|\mathcal{L}|} \sum_{\ell \in \mathcal{L}} X_\ell^\omega;\ \bar{R} \leftarrow \sum_{\omega \in \Omega} p^\omega \frac{1}{|\mathcal{L}|} \sum_{\ell \in \mathcal{L}} R_\ell^\omega$ ▷ mean of subproblem designs, reset inventories

for $\ell \in \mathcal{L},\ \omega \in \Omega$ **do** ▷ update Lagrange multipliers

$\mu_\ell^\omega \leftarrow \mu_\ell^\omega + \rho_\mu (X_\ell^\omega - \bar{X});\ \theta_\ell^\omega \leftarrow \theta_\ell^\omega + \rho_\theta (R_\ell^\omega - \bar{R})$

if $\underline{z}_i > \underline{z}$ **then** $\underline{z} \leftarrow \underline{z}_i$ ▷ update lower bound

if $i \bmod k = 0$ **then** ▷ search for new upper bound every k iterations

$j \leftarrow 0$

for $\ell \in \mathcal{L},\ \omega \in \Omega$ **do**

if X_ℓ^ω not previously used to solve $(\overline{\mathcal{SP}})$ **then**

$j \leftarrow j + 1$

Solve $(\overline{\mathcal{SP}})$ with $\hat{X} = X_\ell^\omega,\ \hat{R} = r(X_\ell^\omega)$ to obtain $\bar{z}_{i\ell}^\omega$

▷ $\bar{z}_{i\ell}^\omega$ is an upper bound for z^{SP}

▷ $r(X_\ell^\omega)$ is midpoint of bounds on R, given X_ℓ^ω

▷ $|\mathcal{L}| \cdot |\Omega|$ subproblems in $(\overline{\mathcal{SP}})$ can be solved in parallel

if $\bar{z}_{i\ell}^\omega < \bar{z}$ **then** ▷ update upper bound and incumbent

Solve $(\overline{\mathcal{SP}})$ with $\hat{X} = X_\ell^\omega$, optimizing R via bisection search to obtain R_ℓ^ω, $\bar{z}_{i\ell}^\omega$

$X^* \leftarrow X_\ell^\omega,\ R^* \leftarrow R_\ell^\omega,\ \bar{z} \leftarrow \bar{z}_{i\ell}^\omega$

if $\bar{z} - \underline{z} \leq \varepsilon \bar{z}$ **or** $j = \kappa$ **then end-for**

$i \leftarrow i + 1$

return $(X^*, R^*, \bar{z}, \underline{z})$

$$\text{s.t.} \qquad X_\ell^\omega \in \mathcal{X} \tag{3.4b}$$

$$Y_t^\omega \in \mathcal{Y}_t^\omega(X_\ell^\omega, S_t^\omega), \quad \forall t \in \mathcal{T}_\ell \tag{3.4c}$$

$$Y_t^{\hat{\omega}} \in \mathcal{Y}_t^{\hat{\omega}}(X_\ell^\omega, S_t^{\hat{\omega}}), \quad \forall t \in \mathcal{T}_{\hat{\ell}} \tag{3.4d}$$

$$\bar{Y}_t^\omega = \underline{Y}_t^\omega + g_t(Y_t^\omega), \quad \forall t \in \mathcal{T}_\ell \tag{3.4e}$$

$$\bar{Y}_t^{\hat{\omega}} = \underline{Y}_t^{\hat{\omega}} + g_t(Y_t^{\hat{\omega}}), \quad \forall t \in \mathcal{T}_{\hat{\ell}} \tag{3.4f}$$

$$\underline{Y}_t^\omega = \bar{Y}_{t-1}^\omega, \quad \forall t \in \mathcal{T}_\ell^- \tag{3.4g}$$

$$\underline{Y}_t^{\hat{\omega}} = \bar{Y}_t^{\hat{\omega}}, \quad \forall t \in \mathcal{T}_{\hat{\ell}}^- \tag{3.4h}$$

$$\underline{Y}_t^\omega = R_\ell^\omega, \quad t = (\ell - 1)\upsilon + 1 \tag{3.4i}$$

$$\underline{Y}_t^{\hat{\omega}} = R_\ell^\omega, \quad t = (\hat{\ell} - 1)\upsilon + 1 \tag{3.4j}$$

$$\underline{Y}_t^{\hat{\omega}} = \bar{Y}_t^{\hat{\omega}}, \quad \forall t \in \mathcal{T}_{\hat{\ell}}^- \tag{3.4k}$$

$$\bar{Y}_t^\omega = R_\ell^\omega, \quad t = \ell\upsilon \tag{3.4l}$$

$$\bar{Y}_t^{\hat{\omega}} = R_\ell^\omega, \quad t = \hat{\ell}\upsilon \tag{3.4m}$$

$$\underline{Y}_1^\omega = Y_0, \tag{3.4n}$$

$$\underline{Y}_1^{\hat{\omega}} = Y_0. \tag{3.4o}$$

The objective function in (3.4) is equal to the sum of $1/(|\mathcal{L}| \cdot |\Omega| - 1)$ times the objective function for the decision variables in block $\hat{\ell}$ and scenario $\hat{\omega}$, and the entire objective function for the decision variables specific to block ℓ and scenario ω. With the exception of design constraints (3.4b), Model (3.4) contains exactly two copies of each of the constraints from model (3.2), one specific to block-scenario pair (ℓ, ω) and one for the pair $(\hat{\ell}, \hat{\omega})$. Solving the $(|\mathcal{L}| \cdot |\Omega| - 1)$ subproblems of the form in model (3.4) is equivalent to solving model ($\underline{\mathcal{SP}}$).

3.4 Preliminary Results

In this section, we report computational results for the case study of Kharga, Egypt under a deterministic instance, denoted ($\mathcal{M}$), using a rigid schedule from Gildea et al. (2017) and under model ($\mathcal{SP}$). The results: (i) demonstrate the relative robustness of design decisions obtained by solving models that incorporate uncertainty in load and PV resources, compared to those obtained by solving a deterministic model with a rigid schedule; and, (ii) provide insight on the impact that fuel and shortfall costs have on the optimal design for our case study. Our case study uses the same technology set and computational resources as those described in Section 2.5; the recourse problem consists of $|\Omega| = 5$ scenarios. Given a proposed design decision, whether by solving a deterministic instance with a rigid schedule or solving ($\mathcal{SP}$), we estimate expected fuel consumed and unmet load using 50 out-of-sample scenarios.

Table 3.4 compares the optimal design and solution time of the deterministic model with rigid scheduling to those of model ($\mathcal{SP}$). The results show that the while both solutions offer similar battery and PV capacity, solving model ($\mathcal{SP}$) yields a design that is more likely to meet the load under realistic assumptions for FOB occupancy than those provided by a rigid schedule, at a computational cost.

Table 3.5 displays a summary of the optimal design as the costs of fuel and shortfall vary. The results show that batteries are added to the design when both costs are high, while PV is added for high fuel costs.

	Deterministic model ($\mathcal{M}$)	**Stochastic model** ($\mathcal{SP}$), $\lvert\Omega\rvert = 5$
Design decision:		
Diesel generator capacity (kW)	200	260
Battery capacity (kW/kWh)	100	100
PV capacity (kW)	75	75
Performance measures:		
Expected fuel consumed (gal)	35,700	35,800
Expected unmet load (%)	2.552	0.006
Solution time (minutes):	2	27

Table 3.4: Comparison of optimal designs, solution times, and performance measures obtained by solving the deterministic model under a rigid schedule to those obtained by solving model ($\mathcal{SP}$). Algorithm 1 is used to obtain solutions to the deterministic model, with model ($\bar{\mathcal{A}}$)=($\bar{\mathcal{M}}$) and ($\underline{\mathcal{A}}$)=($\underline{\mathcal{M}}$) as described in Section 2.4. The model instances allow at most 75 kW of PV solar capacity, and allows battery capacity to be installed in 50 kW increments, up to 200kW. Algorithm 2 is implemented to obtain solutions to model ($\mathcal{SP}$). Models are solved using CPLEX v. 12.6.2.0, via Python 2.7.9. The performance measures are estimated using 50 out-of-sample scenarios.
Penalty for shortfall: \$100/kWh unmet load.
Termination criteria for each subproblem: min{60 seconds, 0.5% optimality gap.}
Termination criteria for each instance: 5% optimality gap.

Optimal Design's Diesel/Battery/Solar Capacity, in kW						
	Fuel cost ($/gal)					
Shortfall cost ($/kWh)	5	20	50	100	250	1000
0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0
0.1	300/0/0	275/0/75	275/0/75	275/0/75	275/0/75	275/50/75
0.25	330/0/0	275/0/75	275/0/75	275/0/75	275/50/75	260/100/75
0.5	330/0/0	275/0/75	275/50/75	275/50/75	260/100/75	260/100/75
1	330/0/0	275/50/75	275/50/75	275/50/75	260/100/75	260/100/75
2.5	330/0/0	275/50/75	260/100/75	260/100/75	260/100/75	260/100/75
5	330/0/0	275/50/75	260/100/75	260/100/75	260/100/75	260/100/75
10	330/0/0	260/100/75	260/100/75	260/100/75	260/100/75	260/100/75
100	330/0/0	260/100/75	260/100/75	260/100/75	260/100/75	260/100/75

Table 3.5: Comparison of optimal designs for model ($\mathcal{SP}$) ($|\Omega| = 5$) as shortfall and fuel costs vary for the Kharga, Egypt case study. Each entry shoes, in order, the capacity of all diesel generators, batteries, and PV systems in the optimal design. The model instances allow at most 75 kW of PV solar capacity, and allows battery capacity to be installed in 50 kW increments, up to 200kW. Algorithm 2 is used to obtain solutions to model ($\mathcal{SP}$). Models are solved using CPLEX v. 12.6.2.0, via Python 2.7.9.
Termination criteria for each subproblem: min{60 seconds, 0.5% optimality gap.}
Termination criteria for each instance: 5% optimality gap.

3.5 Conclusion

We present a method to generate scenarios of PV power output and load that in a correlated manner by using common wather files as input. We present an extension of the model in Section 2.2.3 that allows for uncertainty in the load and PV resources, and we leverage the decomposition method from Section 2.4 that separates the problem into smaller subproblems, using Lagrangian relaxation to obtain a lower bound and fixing inventory levels at regular intervals to obtain an upper bound. We apply the methodology to a case study using weather data from Kharga, Egypt, and solve instances to within 5% of MIP optimality. We show that solutions to models that account for variability in FOB occupancy in addition to weather are more likely to meet load than solutions to a deterministic model with a rigid schedule. Finally, we

provide insight on how shortfall and fuel costs influence the introduction of renewable and storage resources into the design.

Chapter 4

Optimizing the Design of a Latin Hypercube Sampling Estimator

4.1 Introduction

Variance reduction techniques are commonly used in Monte Carlo simulation to reduce the number of samples required to achieve a confidence interval of desired width when estimating the expectation of a univariate or multivariate function for cases in which analytical calculations are intractable. In the simpler univariate case, Neyman (1934) develops a stratified sampling estimator, wherein the support of the random variable is partitioned into strata. Straightforward extension of this idea to higher dimensions does not scale well because the number of strata grows exponentially in the dimension, but Latin hypercube sampling (LHS), introduced by McKay et al. (1979), provides a type of multivariate stratification.

When the underlying d-dimensional random vector has independent components, an LHS estimator partitions the support of each component into equal-probability strata, and exploits independence by randomly ordering samples from each component to form d-tuples. Iman and Conover (1980) generalize LHS to allow for cells of unequal probability, and characterize the sampling variance of an LHS estimator under specific conditions. Stein (1987) describes the asymptotic variance

of an LHS estimator, relative to a naïve Monte Carlo estimator, and Owen (1992) establishes a central limit theorem for LHS. Drew and Homem-de-Mello (2012) show that the upper bound on the probability of a large deviation under LHS is no higher than that of naïve Monte Carlo sampling.

LHS and its extensions are seen in a variety of applications. Olsson et al. (2003) implement LHS in the estimation of structural reliability. Freimer et al. (2012), Stockbridge and Bayraksan (2016), and Bayraksan (2018) assess the impact of LHS on reducing the variance and bias when estimating optimal values in stochastic optimization problems. Packham and Schmidt (2010) establish a central limit theorem for LHS with dependent variables, and apply this to the valuation of first-to-default credit baskets and Asian basket options. Helton and Davis (2003) use the sample reweighting technique for nonuniform LHS developed by Iman and Conover (1980) to conduct uncertainty and sensitivity analysis for a two-phase fluid flow model. Morton et al. (2014) implement nonuniform LHS to ensure the sampling of rare events to assess risk in nuclear power.

Mease and Bingham (2006) study how to optimize the strata of a nonuniform LHS estimator, and, to our knowledge, this is the closest work in the literature to what we propose. They derive first-order optimality conditions when the dimension and number of samples are small, but they say that this approach does not scale well. So, they employ a heuristic search involving coordinate descent on a grid for larger problems.

We similarly begin by formulating a family of nonlinear optimization models. In particular, we develop methods of choosing nonuniform strata over the support

of a random variable with the goal of minimizing sampling variance for stratified sampling or, in the case of multivariate sampling, LHS. Section 4.2 formulates a nonlinear program to construct a stratified sampling estimator with minimum variance, and reformulates that optimization model as a tractable dynamic program. The development of this stratified sampling estimator is not particularly useful in its own right. Rather, we view it as a subroutine for developing the two LHS estimators that we propose in Section 4.3. While we present these estimators as nonlinear programs, we recast the first as a dynamic program, and employ a heuristic search using coordinate descent to obtain solutions to the second. Section 4.4 details empirical results of our stratified sampling and LHS schemes for a collection of sample functions. Section 4.5 concludes.

4.2 Nonuniform Stratified Sampling

We wish to estimate $\mathbb{E}[h(\xi)]$, in which ξ is a univariate random variable and $h : \mathbb{R} \to \mathbb{R}$. To do so, we use a stratified sampling routine, which partitions the support of ξ into contiguous strata, $\mathcal{S}_k$, $k = 1, 2, \ldots, K$, which we also call cells. Here, each cell has probability mass $p_k = \mathbb{P}[\xi \in \mathcal{S}_k]$, and the estimator allocates a sample size n_k to cell k. In this section, we formulate a nonlinear program to find the cell widths that yield a minimum-variance estimator, restricting attention to the univariate case. In Section 4.3, we address multivariate sampling under an LHS framework.

4.2.1 Assumptions

We assume ξ is a continuous random variable with known probability density function (pdf), $f(x)$, and cumulative distribution function (cdf), $F(x)$. We assume that we have in analytical form, $F^{-1}(u)$, for $u \in [0,1]$, or that we can numerically evaluate this inverse cdf. We assume that we can compute $p_k = \mathbb{P}[\xi \in \mathcal{S}_k]$, for each cell as well as relevant expectations. Further, we assume we know the desired number of strata, K, and our total computational budget, N. Our goal is to select the breakpoints and sample sizes to yield a stratified sampling estimator of minimum variance.

4.2.2 Nonlinear Programming Formulation

Our stratified sampling estimator has the following form:

$$h_N = \sum_{k=1}^{K} p_k \bar{h}_{n_k}, \tag{4.1}$$

in which $\bar{h}_{n_k}$ is a sample mean of n_k independent and identically distributed (i.i.d.) observations of $[h(\xi)|\xi \in \mathcal{S}_k]$; that is, i.i.d. observations of $h(\xi)$, conditioned on ξ being in cell k. We therefore have that

$$\mathbb{V}\text{ar}[h_N] = \sum_{k=1}^{K} p_k^2 \frac{\mathbb{V}\text{ar}[h(\xi)|\xi \in \mathcal{S}_k]}{n_k}. \tag{4.2}$$

The optimization model that we formulate in this section assumes the cells $\mathcal{S}_k$ are of the form $\mathcal{S}_k = (F^{-1}(b_{k-1}), F^{-1}(b_k))$, in which $0 = b_0 \leq b_1 \leq b_2 \leq \cdots \leq b_K = 1$, and with this construct aims to minimize $\mathbb{V}\text{ar}[h_N]$.

Sets and Indices

$k \in \mathcal{K} = \{1, 2, \ldots, K\}$: indices defining the strata

Functions

h: a univariate function $h : \mathbb{R} \to \mathbb{R}$

Data

f: pdf of ξ

F: cdf of ξ

N: total number of samples

Decision Variables

b_k: location of breakpoint k in the interval $[0, 1]$ used to create strata

μ_k: $\mathbb{E}[h(\xi) \,|\, \xi \in \mathcal{S}_k]$, where $\mathcal{S}_k = (F^{-1}(b_{k-1}), F^{-1}(b_k))$

σ_k^2: $\mathbb{E}[(h(\xi) - \mu_k)^2 \,|\, \xi \in \mathcal{S}_k]$

p_k: $\mathbb{P}[\xi \in \mathcal{S}_k] = b_k - b_{k-1}$

n_k: number of samples allocated to cell k

Boundary Conditions

$b_0 = 0$

$b_K = 1$

Note:

We use b, μ, σ^2, p, and n to denote the vectors $(b_0, b_1, \ldots, b_K)$, $(\mu_1, \mu_2, \ldots, \mu_K)$, etc.

Formulation

$$\min_{b,\mu,\sigma^2,p,n} \quad \sum_{k \in \mathcal{K}} \frac{p_k^2 \sigma_k^2}{n_k} \tag{4.3a}$$

$$\text{s.t.} \quad \mu_k = \frac{\int_{F^{-1}(b_{k-1})}^{F^{-1}(b_k)} h(x)f(x)dx}{b_k - b_{k-1}}, \quad \forall k \in \mathcal{K}, \tag{4.3b}$$

$$\sigma_k^2 = \frac{\int_{F^{-1}(b_{k-1})}^{F^{-1}(b_k)} (h(x) - \mu_k)^2 f(x)dx}{b_k - b_{k-1}}, \quad \forall k \in \mathcal{K}, \tag{4.3c}$$

$$p_k = b_k - b_{k-1}, \quad \forall k \in \mathcal{K}, \tag{4.3d}$$

$$\sum_{k \in \mathcal{K}} n_k = N, \tag{4.3e}$$

$$p_k \geq 0, \; n_k \geq 0, \quad \forall k \in \mathcal{K}. \tag{4.3f}$$

Discussion

The objective in (4.3a) seeks a stratification of the support of ξ to minimize the variance of the estimator, as indicated in equation (4.2). Constraints (4.3b) and (4.3c) define μ_k and σ_k^2, respectively; both constraints are nonlinear in the decision vector, b. Constraint (4.3d) relates cell k's width, p_k, to the location of its breakpoints, b_k and b_{k-1}. This, coupled with constraint (4.3f) and the boundary conditions, ensures that $0 = b_0 \leq b_1 \leq \cdots \leq b_{K-1} \leq b_K = 1$. Constraint (4.3e) restricts the sum of sample sizes to the computational budget, N. We have relaxed an integer restriction on the sample sizes, n_k, in constraint (4.3f), which allows an optimal solution to allocate a fractional number of samples for each cell.

We view model (4.3) as notional in the sense that, if we could compute exactly terms like μ_k in (4.3b), then we could compute $\mathbb{E}[h(\xi)]$ exactly, and we would not employ Monte Carlo sampling. However, as indicated above, we extend this idea in the next section to an LHS estimator, in which we assume relevant one-dimensional integrals are tractable. In what follows, we modify model (4.3) in two ways. First, we remove n_k because we can analytically optimize with respect to n for fixed break-

points. Second, we create a discrete set of candidate breakpoints from which strata may be constructed. These two steps are discussed in Sections 4.2.3 and 4.2.4, respectively.

4.2.3 Objective Function Reformulation

Suppose the breakpoints $b_k, k \in \mathcal{K}$, are known. Then constraints (4.3b)-(4.3d) can be removed and the resulting optimization problem is:

$$\min_{n} \quad \sum_{k\in\mathcal{K}} \frac{p_k^2\sigma_k^2}{n_k} \tag{4.4a}$$

$$\text{s.t.} \quad \sum_{k\in\mathcal{K}} n_k = N \tag{4.4b}$$

$$n_k \geq 0, \quad \forall k \in \mathcal{K}. \tag{4.4c}$$

An optimal solution of model (4.4) is achieved when n_k is proportional to $p_k\sigma_k$, i.e.,

$$n_k = N\left(\frac{p_k\sigma_k}{\sum_{k\in\mathcal{K}} p_k\sigma_k}\right);$$

see, e.g., Neyman (1934). Substituting this value for n_k into the objective function in (4.4a) yields:

$$\frac{1}{N}\left(\sum_{k\in\mathcal{K}} p_k\sigma_k\right)^2. \tag{4.5}$$

The revised objective function in (4.5) allows model (4.3) to simplify to:

$$\min_{b,\mu,\sigma^2} \quad \sum_{k\in\mathcal{K}} (b_k - b_{k-1})\sigma_k \tag{4.6a}$$

$$\text{s.t.} \quad 0 = b_0 \leq b_1 \leq \cdots \leq b_{K-1} \leq b_K = 1 \tag{4.6b}$$

$$\text{(4.3b)-(4.3c).} \tag{4.6c}$$

The breakpoints b_k, $k \in \mathcal{K}$, are the primary decision variables in model (4.6), as variables μ_k and σ_k^2 are determined by the specification of these breakpoints. While model (4.6) is still nonconvex, the additive form of the objective function in (4.6a) allows for the development of a dynamic programming algorithm that we describe in Section 4.2.4, at least when we restrict the choices of b_k to a prespecified univariate grid.

4.2.4 Dynamic Programming Algorithm

Let $\mathcal{B} = \{b^0, b^1, b^2, \ldots, b^L\}$ specify a set of candidate breakpoints, in which $0 \equiv b^0 < b^1 < b^2 < \cdots < b^L \equiv 1$, and where $L \gg K$. We consider the restriction of model (4.6) in which we add the constraint $b_k \in \mathcal{B}$, $k \in \mathcal{K}$. Each term in the objective function of (4.6a) then has the form $(b^{\ell'} - b^{\ell})\sigma(b^{\ell}, b^{\ell'})$, where

$$\sigma^2(b^{\ell}, b^{\ell'}) = \frac{\int_{F^{-1}(b^{\ell})}^{F^{-1}(b^{\ell'})} (h(x) - \mu(b^{\ell}, b^{\ell'}))^2 f(x) dx}{b^{\ell'} - b^{\ell}}, \tag{4.7}$$

and where

$$\mu(b^{\ell}, b^{\ell'}) = \frac{\int_{F^{-1}(b^{\ell})}^{F^{-1}(b^{\ell'})} h(x) f(x) dx}{b^{\ell'} - b^{\ell}}, \tag{4.8}$$

for $\ell = 0, 1, \ldots, L-1, \ell' = \ell + 1, \ell + 2, \ldots, L$.

We can solve this variant of model (4.6) via a dynamic programming algorithm, which can be visualized using the directed acyclic graph (DAG) shown in Figure 4.1. The DAG has nodes (k, ℓ) for $k = 0, 1, \ldots, K$, and for $\ell = 0, 1, \ldots, L$. For all $k = 0, 1, \ldots, K-1$, we create an edge from node (k, ℓ) to node $(k+1, \ell')$, for all $\ell = 0, \ldots, L$, $\ell' = \ell, \ldots, L$, with length $(b^{\ell'} - b^{\ell})\sigma(b^{\ell}, b^{\ell'})$, in which $\sigma(b^{\ell}, b^{\ell'})$ is defined

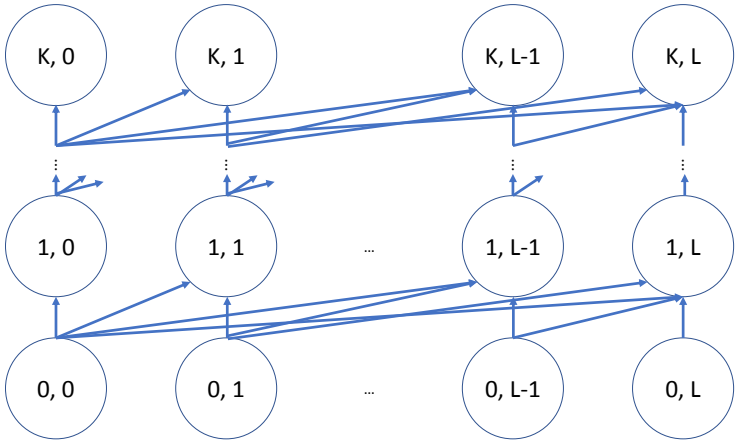

Figure 4.1: Shortest-path problem associated with the dynamic programming solution of model (4.6) under the restriction that each b_k comes from a set of finite, prespecified breakpoints. We create an edge from node (k, ℓ) to node $(k+1, \ell')$, for all $k = 0, \ldots, K-1,\ \ell = 0, \ldots, L,\ \ell' = \ell, \ldots, L$, with length $(b^{\ell'} - b^{\ell})\sigma(b^{\ell}, b^{\ell'})$, in which $\sigma^2(b^{\ell}, b^{\ell'})$ is defined in equation (4.7). If node (k, ℓ) is part of the shortest path from $(0, 0)$ to (K, L), then breakpoint $b_k = b^{\ell}$ is in the obtained optimal solution.

in equation (4.7). The shortest path from node $(0, 0)$ to (K, L) then specifies an optimal solution to model (4.6), under the restriction $b_k \in \mathcal{B}, k \in \mathcal{K}$.

We note that computing the edge lengths in the DAG of Figure 4.1 requires more effort than computing $\mathbb{E}[h(\xi)]$, because $\mathbb{E}[h(\xi)]$ is given by the sum of $(b^{\ell'} - b^{\ell})\mu(b^{\ell}, b^{\ell'})$ along any path in the DAG from $(0, 0)$ to (K, L). Therefore, we emphasize that we do not view this as useful for reducing the variance of stratified sampling estimators; rather, we view it as a subroutine for an optimized LHS estimator that we describe next.

4.3 Nonuniform LHS

This section extends the method described in Section 4.2 to higher dimensions to optimize an LHS estimator. Let $\xi = (\xi(1), \xi(2), \ldots, \xi(d))$ be a vector of indepen-

dent random variables, and let $h : \mathbb{R}^d \to \mathbb{R}$; further, suppose we plan to use LHS to estimate $\mathbb{E}[h(\xi)]$. Similar to the stratified sampling procedure in Section 4.2, we partition the support of each random variable $\xi(i)$ into K strata, $\mathcal{S}_k(i), k \in \mathcal{K}$. However, for each random variable $\xi(i)$, exactly one value $\xi^k(i)$ is sampled from each cell, $k \in \mathcal{K}$. Next, the K realizations from $\xi(1)$ are randomly paired, without replacement, with the realizations from $\xi(2)$. These are, in turn, paired at random with the other components of ξ, until we generate K d-tuples:

$$\xi^k = (\xi^k(1), \xi^k(2), \ldots, \xi^k(d)), k \in \mathcal{K}.$$

We obtain these d-tuples in the following way. For $i \in \mathcal{I} = \{1, \ldots, d\}$, let $(\pi(i,1), \pi(i,2), \ldots, \pi(i,K))$ denote a random permutation of $\{1, 2, \ldots, K\}$. Then, for $i \in \mathcal{I}$, $k \in \mathcal{K}$, let

$$\xi^k(i) \sim [\xi(i) \,|\, \xi(i) \in \mathcal{S}_{\pi(i,k)}(i)].$$

The set of possible combinations for a K-tuple, generated by $\pi(\cdot)$, represents a partition of the support of ξ into K^d cells.

Let $\Gamma^k = \mathcal{S}_{\pi(1,k)}(1) \times \mathcal{S}_{\pi(2,k)}(2) \times \cdots \times \mathcal{S}_{\pi(d,k)}(d)$ be the Cartesian product of the chosen strata for each random variable $\xi^k(i),\ i \in \mathcal{I}$, and let

$$\mathbb{P}[\xi \in \Gamma^k] = \prod_{i \in \mathcal{I}} \mathbb{P}[\xi(i) \in \mathcal{S}_{\pi(i,k)}(i)],\ k \in \mathcal{K}.$$

In this setting the LHS estimator given by

$$h_K^{LHS} = \sum_{k \in \mathcal{K}} K^{d-1} \mathbb{P}[\xi \in \Gamma^k] h(\xi^k) \tag{4.9}$$

was proposed by Iman and Conover (1980) to extend the work of McKay et al. (1979). In McKay et al., the support of each random variable has equal-probability strata, meaning

$$\mathbb{P}[\xi(i) \in \mathcal{S}_{\pi(i,k)}(i)] = \frac{1}{K},\ i \in \mathcal{I},\ k \in \mathcal{K}.$$

In this case, the weights on $h(\xi^k)$ in equation (4.9) are simply $1/K$. Under nonuniform LHS, the weights are instead random due to $\mathbb{P}[\xi \in \Gamma^k]$, because the cells Γ^k are determined by the random permutation, π, and can have unequal probability. In what follows, we develop two separate solution approaches to find minimum-variance LHS designs; one leverages the dynamic programming solution method from Section 4.2.4 for multiplicative functions, and the other uses a coordinate descent-based heuristic for more general functions. We note that unlike in the model for the stratified sampling estimator discussed in Section 4.2, the computational budget is always $N = K$ for an LHS estimator.

Similar to the assumptions of Section 4.2.1, we wish to obtain an estimator for $\mathbb{E}[h(\xi)]$ with minimum variance, except we assume that ξ is multivariate, with independent components. For each random variable $\xi(i),\ i \in \mathcal{I}$, we assume we have, or can numerically evaluate, the inverse cdf, $F_i^{-1}(u)$, for $u \in [0, 1]$. Finally, we assume that the second moment of each random variable is finite. Our goal is to select a set of breakpoints that define the strata of each random variable $\xi(i),\ i \in \mathcal{I}$, to minimize total sampling error under an LHS routine.

4.3.1 Solution Method (i): Dynamic Programming

In order to guide design of the cells we use in our LHS estimator, we make the following approximation:

$$h(\xi) \approx \sum_{i \in \mathcal{I}} h_i(\xi), \tag{4.10}$$

in which h_i evaluates h with each random variable except for $\xi(i)$ set to a predetermined constant, i.e.,

$$h_i(\xi) = h(a(1), a(2), \ldots, a(i-1), \xi(i), a(i+1), \ldots, a(d-1), a(d)).$$

In the numerical experiments we describe in Section 4.4, we use $a(i) = \mathbb{E}[\xi(i)]$, for $i \in \mathcal{I}$. Applying the LHS estimator to the right-hand side of the approximation (4.10) amounts to performing stratified sampling on each term $h_i(\xi)$, where the stratification is only on component $\xi(i)$, albeit with one sample per cell. Thus, following equation (4.2) with $n_k = 1,\ k \in \mathcal{K}$,

$$\mathbb{V}\text{ar}[h_K^{LHS}] \approx \sum_{i \in \mathcal{I}} \sum_{k \in \mathcal{K}} p_k^2(i) \sigma_k^2(i), \tag{4.11}$$

in which $\sigma_k^2(i)$ is given by

$$\sigma_k^2(i) = \mathbb{V}\text{ar}\left[h_i(\xi) | \xi(i) \in \mathcal{S}_k(i)\right], \tag{4.12}$$

for $k \in \mathcal{K}$. Minimizing the LHS variance of the right-hand side of approximation (4.11) leads to d separate optimization problems of the form:

$$\min_{b(i), \mu(i), \sigma^2(i)} \quad \sum_{k \in \mathcal{K}} (b_k(i) - b_{k-1}(i))^2 \sigma_k^2(i) \tag{4.13a}$$

$$\text{s.t.} \quad \mu_k(i) = \frac{\int_{F_i^{-1}(b_{k-1}(i))}^{F_i^{-1}(b_k(i))} h_i(x) f_i(x) dx}{b_k(i) - b_{k-1}(i)}, \quad \forall k \in \mathcal{K} \tag{4.13b}$$

$$\sigma_k^2(i) = \frac{\int_{F_i^{-1}(b_{k-1}(i))}^{F_i^{-1}(b_k(i))} (h_i(x) - \mu_k(i))^2 f_i(x) dx}{b_k(i) - b_{k-1}(i)}, \quad \forall k \in \mathcal{K} \quad (4.13c)$$

$$0 = b_0(i) \leq b_1(i) \leq \cdots \leq b_{K-1}(i) \leq b_K(i) = 1, \quad (4.13d)$$

for $i \in \mathcal{I}$. Here, f_i denotes the marginal pdf of $\xi(i)$, and the vectors $b(i)$, $\mu(i)$, and $\sigma^2(i)$ are as defined in Section 2.2, except that $\mu(i)$, and $\sigma^2(i)$ are now defined with respect to the univariate $h_i(\xi)$.

With an objective function that separates by each component of ξ, we can apply the dynamic programming approach of Section 4.2.4 d times in solving model (4.13). We assume that we can numerically compute the univariate integrals that define the $O(dKL^2)$ edges that compose the d DAGs; i.e., we can numerically compute $\sigma^2(i)(b^\ell, b^{\ell'})$ and $\mu(i)(b^\ell, b^{\ell'})$ as defined by equations (4.7) and (4.8), respectively, for each component i. The shortest paths from the d DAGs define the K^d cells from which we obtain LHS samples.

The variance of h_K^{LHS} can be expressed as follows:

$$\mathbb{V}\text{ar}[h_K^{LHS}] = \mathbb{E}[\mathbb{V}\text{ar}[h_K^{LHS}]|\pi] + \mathbb{V}\text{ar}[\mathbb{E}[h_K^{LHS}]|\pi], \quad (4.14)$$

in which we assume that $\pi = (\pi(1,1), \ldots, \pi(d,K))$ are random permutations. Taken as a collection over all d components, these permutations occur with equal probability over the $(K!)^{d-1}$ possible LHS designs. The objective function in equation (4.13a), when summed across all d components, is equivalent to the LHS estimator variance for $d = 1$. As indicated above, a general LHS estimator has random weights that depend on which cells are selected, and so for $d > 1$ the objective function in equation (4.13a) approximates $\mathbb{E}[\mathbb{V}\text{ar}[h_K^{LHS}]|\pi]$. Now consider two types of functions: (i) linear in which

$h(\xi) = \sum_{i \in \mathcal{I}} c_i \xi(i)$; and, (ii) product in which $h(\xi) = \prod_{i \in \mathcal{I}} \xi(i)$. Table 4.1 displays both terms of the LHS estimator variance from equation (4.14) for product functions, using both uniform and optimized strata for $d = 2$. The table shows that LHS designs which minimize the objective function in (4.13a) for each random variable decreases both $\mathbb{E}[\mathbb{V}\text{ar}[h_K^{LHS}]|\pi]$ and $\mathbb{V}\text{ar}[\mathbb{E}[h_K^{LHS}]|\pi]$ across a collection of probability distributions. However, the results of Table 4.1 do not translate well to the case of a linear function, as Proposition 4.3.1 shows.

	Uniform Strata		**Optimized Strata**		**Uniform/Optimized**	
	$\mathbb{E}[\mathbb{V}\text{ar}[h_K^{LHS}]]$	$\mathbb{V}\text{ar}[\mathbb{E}[h_K^{LHS}]]$	$\mathbb{E}[\mathbb{V}\text{ar}[h_K^{LHS}]]$	$\mathbb{V}\text{ar}[\mathbb{E}[h_K^{LHS}]]$	$\mathbb{E}[\mathbb{V}\text{ar}[h_K^{LHS}]]$	$\mathbb{V}\text{ar}[\mathbb{E}[h_K^{LHS}]]$
χ_1^2	9.50E-01	5.45E-01	9.20E-02	2.60E-02	10.33	20.96
Beta(1,2)	4.31E-04	8.43E-04	2.65E-04	1.59E-04	1.62	5.29
Exponential(1)	2.47E-01	1.80E-01	4.47E-02	5.01E-03	5.52	35.95
Gamma(2,1)	2.23E+00	8.40E-01	8.67E-01	7.92E-02	2.57	10.60
Weibull(2,1)	1.44E-02	1.14E-02	9.21E-03	3.56E-03	1.57	3.21

Table 4.1: Comparison of $\mathbb{E}[\mathbb{V}\text{ar}[h_K^{LHS}]|\pi]$ and $\mathbb{V}\text{ar}[\mathbb{E}[h_K^{LHS}]|\pi]$, i.e., the components of the decomposition of variance, under LHS designs with uniform strata and optimized strata obtained by solving model (4.13) for each of $d = 2$ components, for a collection of multivariate functions in which $h(\xi) = \prod_{i \in \mathcal{I}} \xi(i)$.

Proposition 4.3.1. *Let* $(\pi(i,1), \pi(i,2), \ldots, \pi(i,K))$ *denote a random permutation of* $\{1, 2, \ldots, K\}$*, for* $i \in \mathcal{I}$*. Let* $\Gamma^k = \mathcal{S}_{\pi(1,k)}(1) \times \mathcal{S}_{\pi(2,k)}(2) \times \cdots \times \mathcal{S}_{\pi(d,k)}(d)$*, and let*

$$\mathbb{P}[\xi \in \Gamma^k] = \prod_{i \in \mathcal{I}} \mathbb{P}[\xi(i) \in \mathcal{S}_{\pi(i,k)}(i)], \ \forall k \in \mathcal{K} = \{1, 2, \ldots, K\}.$$

Define

$$h_K^{LHS} = \sum_{k \in \mathcal{K}} K^{d-1} \mathbb{P}[\xi \in \Gamma^k] h(\xi^k),$$

and let

$$h(\xi) = \sum_{i \in \mathcal{I}} c_i \xi(i),$$

in which $c_i \in \mathbb{R},\ \forall i \in \mathcal{I}$. *If*

$$\mathbb{P}[\xi(i) \in \mathcal{S}_{\pi(i,k)}(i)] = \frac{1}{K}, \forall i \in \mathcal{I},\ k \in \mathcal{K},$$

then $\mathbb{V}ar[\mathbb{E}[h_K^{LHS}]|\pi] = 0$.

Proof:

Our assumption of equal-probability strata yields $\mathbb{P}[\xi \in \Gamma^k] = (1/K)^d$, so we can rewrite h_K^{LHS} as:

$$h_K^{LHS} = \sum_{k \in \mathcal{K}} \frac{1}{K} h(\xi^k).$$

Our assumption of a linear function yields

$$\mathbb{E}[h_K^{LHS}|\pi] = \frac{1}{K} \sum_{k \in \mathcal{K}} \sum_{i \in \mathcal{I}} c_i \mathbb{E}[\xi(i)|\xi(i) \in \mathcal{S}_{\pi(i,k)}(i)] = \sum_{i \in \mathcal{I}} c_i \left(\sum_{k \in \mathcal{K}} \frac{1}{K} \mathbb{E}[\xi(i)|\xi \in \mathcal{S}_k(i)] \right).$$

Because the inner sum, $\sum_{k \in \mathcal{K}} \frac{1}{K} \mathbb{E}[\xi(i)|\xi(i) \in \mathcal{S}_k(i)]$, is equivalent to $\mathbb{E}[\xi(i)]$, the final result is

$$\mathbb{E}[h_K^{LHS}|\pi] = \sum_{i \in \mathcal{I}} c_i \mathbb{E}[\xi(i)].$$

The value of $\mathbb{E}[h_K^{LHS}|\pi]$ does not depend on the permutation π, and thus
$\mathbb{V}\text{ar}[\mathbb{E}[h_K^{LHS}|\pi]] = 0$. ■

Proposition 4.3.1 shows that $\mathbb{V}\text{ar}[\mathbb{E}[h_K^{LHS}]|\pi] = 0$ for a linear function with uniform strata, and this contrasts with the results of Table 4.1 for product functions. As a result, minimizing (4.13a) may come at the cost of increasing $\mathbb{V}\text{ar}[\mathbb{E}[h_K^{LHS}]|\pi]$, especially when h is well-approximated by a linear function. Therefore, to obtain optimized strata for a wider collection of multivariate functions, we develop a second approach, rooted in coordinate descent using another type of approximation.

4.3.2 Solution Method (ii): Coordinate Descent

This section presents a method to improve the design of cells for LHS estimators compared to uniform strata across a wider collection of multivariate functions than those for which we use the dynamic programming approach in Section 4.3.1. The objective function in this setting is the LHS estimator's second moment, which we characterize in Section 4.3.2.1 and then approximate in Section 4.3.2.2. Sections 4.3.2.3 and 4.3.2.4 develop the nonlinear programming formulation and solution method, respectively, and Section 4.3.2.5 describes a further approximation necessary to make the approach practical.

4.3.2.1 Second Moment Characterization of LHS Estimator

For $d = 1$, the variance of an LHS estimator is equivalent to that of the stratified sampling estimator as given in equation (4.2), with $n_k = 1,\ k \in \mathcal{K}$. While the variance of multivariate LHS estimators has been characterized in various ways in the literature (see, e.g., Iman and Conover (1980), Stein (1987), Homem-de-Mello (2008), Drew and Homem-de-Mello (2012)), these characterizations provide insight as opposed to lending themselves to estimation. Proposition 4.3.3 characterizes the LHS estimator's second moment, which we then minimize via the solution method in Sections 4.3.2.3-4.3.2.5.

Lemma 4.3.2. *Let* $(\pi(i,1), \pi(i,2), \ldots, \pi(i,K))$ *denote a random permutation of* $\{1, 2, \ldots, K\}$*, for* $i \in \mathcal{I}$*. Let* $\Gamma^k = \mathcal{S}_{\pi(1,k)}(1) \times \mathcal{S}_{\pi(2,k)}(2) \times \cdots \times \mathcal{S}_{\pi(d,k)}(d)$*, and let*

$$\mathbb{P}[\xi \in \Gamma^k] = \prod_{i \in \mathcal{I}} \mathbb{P}[\xi(i) \in \mathcal{S}_{\pi(i,k)}(i)],\ \forall k \in \mathcal{K} = \{1, 2, \ldots, K\}.$$

Define

$$h_K^{LHS} = \sum_{k \in \mathcal{K}} K^{d-1} \mathbb{P}[\xi \in \Gamma^k] h(\xi^k).$$

Then,

$$\mathbb{E}[(h_K^{LHS})^2] = \mathbb{E}[\mathbb{V}ar[h_K^{LHS}|\pi]] + \mathbb{E}[(\mathbb{E}[h_k^{LHS}|\pi])^2].$$

Proof:

The decomposition of variance yields

$$\begin{aligned}\mathbb{V}\text{ar}[h_K^{LHS}] &= \mathbb{E}[\mathbb{V}\text{ar}[h_K^{LHS}|\pi]] + \mathbb{V}\text{ar}[\mathbb{E}[h_k^{LHS}|\pi]] \\ &= \mathbb{E}[\mathbb{V}\text{ar}[h_K^{LHS}|\pi]] + \mathbb{E}[(\mathbb{E}[h_k^{LHS}|\pi])^2] - (\mathbb{E}[\mathbb{E}[h_k^{LHS}|\pi]])^2,\end{aligned}$$

and the law of total expectation leads to

$$\mathbb{V}\text{ar}[h_K^{LHS}] = \mathbb{E}[\mathbb{V}\text{ar}[h_K^{LHS}|\pi]] + \mathbb{E}[(\mathbb{E}[h_k^{LHS}|\pi])^2] - (\mathbb{E}[h_k^{LHS}])^2.$$

This implies that

$$\begin{aligned}\mathbb{V}\text{ar}[h_K^{LHS}] + (\mathbb{E}[h_k^{LHS}])^2 &= \mathbb{E}[\mathbb{V}\text{ar}[h_K^{LHS}|\pi]] + \mathbb{E}[(\mathbb{E}[h_k^{LHS}|\pi])^2] \\ \Rightarrow \mathbb{E}[(h_K^{LHS})^2] &= \mathbb{E}[\mathbb{V}\text{ar}[h_K^{LHS}|\pi]] + \mathbb{E}[(\mathbb{E}[h_K^{LHS}|\pi])^2]. \quad \blacksquare\end{aligned}$$

Proposition 4.3.3. *Let* $(\pi(i,1), \pi(i,2), \ldots, \pi(i,K))$ *denote a random permutation of* $\{1, 2, \ldots, K\}$*, for* $i \in \mathcal{I}$*, and let* π_q*,* $q \in \mathcal{Q}$*, denote the set of all* $(K!)^{d-1}$ *possible collections of* d *permutations that assign the* K *cells in an LHS design. Let* $\Gamma^k = \mathcal{S}_{\pi(1,k)}(1) \times \mathcal{S}_{\pi(2,k)}(2) \times \cdots \times \mathcal{S}_{\pi(d,k)}(d)$*, and let*

$$\mathbb{P}[\xi \in \Gamma^k] = \prod_{i \in \mathcal{I}} \mathbb{P}[\xi(i) \in \mathcal{S}_{\pi(i,k)}(i)], \ \forall k \in \mathcal{K} = \{1, 2, \ldots, K\}.$$

Define:

$$h_K^{LHS} = \sum_{k \in \mathcal{K}} K^{d-1} \mathbb{P}[\xi \in \Gamma^k] h(\xi^k)$$
$$p_{kq} = \mathbb{P}[\xi \in \Gamma^k | \pi = \pi_q]$$
$$\mu_{kq} = \mathbb{E}[h(\xi^k) | \pi = \pi_q].$$

Then,

$$\mathbb{E}[(h_K^{LHS})^2] = \frac{K^{2d-2}}{(K!)^{d-1}} \sum_{q \in \mathcal{Q}} \sum_{k \in \mathcal{K}} (p_{kq})^2 \cdot \mathbb{E}[(h(\xi^k))^2 | \pi = \pi_q] + \frac{2 \cdot K^{2d-2}}{(K!)^{d-1}} \sum_{q \in \mathcal{Q}} \sum_{k \in \mathcal{K}} \sum_{\substack{k' \in \mathcal{K} \\ k' > k}} (p_{kq} \cdot p_{k'q} \cdot \mu_{kq} \cdot \mu_{k'q}) . \tag{4.15}$$

Proof:

From Lemma 4.3.2, we have $\mathbb{E}[(h_K^{LHS})^2] = \mathbb{E}[\mathbb{V}\text{ar}[h_K^{LHS} | \pi]] + \mathbb{E}[(\mathbb{E}[h_K^{LHS} | \pi])^2]$, where π takes on each of the possible $(K!)^{d-1}$ collections of d permutations in the set $\mathcal{Q}$ with equal probability. The expected sampling variance is:

$$\mathbb{E}[\mathbb{V}\text{ar}[h_K^{LHS} | \pi]] = \frac{1}{(K!)^{d-1}} \sum_{q \in \mathcal{Q}} \sum_{k \in \mathcal{K}} (K^{d-1})^2 (\mathbb{P}[\xi \in \Gamma^k | \pi = \pi_q])^2 \cdot \mathbb{V}\text{ar}[h(\xi^k) | \pi = \pi_q].$$

We move the constant, $(K^{d-1})^2$, outside the summation and expand the second term on the right-hand side to obtain

$$\mathbb{E}[\mathbb{V}\text{ar}[h_K^{LHS} | \pi]] = \frac{K^{2d-2}}{(K!)^{d-1}} \sum_{q \in \mathcal{Q}} \sum_{k \in \mathcal{K}} \left[p_{kq}^2 \cdot \left(\mathbb{E}[(h(\xi^k))^2 | \pi = \pi_q] - \mu_{kq}^2 \right) \right] . \tag{4.16}$$

The second term of interest is

$$\mathbb{E}[(\mathbb{E}[h_K^{LHS} | \pi])^2] = \frac{1}{(K!)^{d-1}} \sum_{q \in \mathcal{Q}} \left(\sum_{k \in \mathcal{K}} K^{d-1} \cdot p_{kq} \cdot \mu_{kq} \right)^2 ,$$

which we expand to

$$\mathbb{E}[(\mathbb{E}[h_K^{LHS}|\pi])^2] = \frac{K^{2d-2}}{(K!)^{d-1}} \sum_{q\in\mathcal{Q}} \sum_{k\in\mathcal{K}} \left(p_{kq}^2 \cdot \mu_{kq}^2\right) + \frac{2 \cdot K^{2d-2}}{(K!)^{d-1}} \sum_{q\in\mathcal{Q}} \sum_{k\in\mathcal{K}} \sum_{\substack{k'\in\mathcal{K} \\ k'>k}} \left(p_{kq} \cdot p_{k'q} \cdot \mu_{kq} \cdot \mu_{k'q}\right). \tag{4.17}$$

Summing the right-hand sides of equations (4.16) and (4.17) yields

$$\mathbb{E}[(h_K^{LHS})^2] = \frac{K^{2d-2}}{(K!)^{d-1}} \sum_{q\in\mathcal{Q}} \sum_{k\in\mathcal{K}} p_{kq}^2 \cdot \mathbb{E}[(h(\xi^k))^2|\pi = \pi_q] + \frac{2 \cdot K^{2d-2}}{(K!)^{d-1}} \sum_{q\in\mathcal{Q}} \sum_{k\in\mathcal{K}} \sum_{\substack{k'\in\mathcal{K} \\ k'>k}} \left(p_{kq} \cdot p_{k'q} \cdot \mu_{kq} \cdot \mu_{k'q}\right). \quad \blacksquare$$

4.3.2.2 Objective Function Reformulation

To improve the design of cells for LHS estimators compared to uniform strata across a wider collection of multivariate functions than the test cases in Section 4.3.1, we consider the following collection of approximations:

$$h(\xi) \approx \frac{1}{(K!)^{d-1}} \sum_{q\in\mathcal{Q}} \sum_{k\in\mathcal{K}} K^{d-1} \mathbb{P}[\xi \in \Gamma^k|\pi = \pi_q] h_{ikq}(\xi(i)),\ \forall i \in \mathcal{I}. \tag{4.18}$$

Here, we let $q \in \mathcal{Q}$ denote the collection of all permutations that assigns the K cells in an LHS design. Further, in equation (4.18) we let

$$h_{ikq}(\xi(i)) = h(a_{kq}(1), \ldots, a_{kq}(i-1), \xi(i), a_{kq}(i+1), \ldots, a_{kq}(d-1), a_{kq}(d)), \tag{4.19}$$

$$\mathbb{P}[\xi \in \Gamma^k|\pi = \pi_q] = \prod_{i\in\mathcal{I}} \mathbb{P}[\xi(i) \in \mathcal{S}_{\pi_q(i,k)}(i)],$$

and

$$a_{kq}(i) = \mathbb{E}[\xi(i)|\xi(i) \in \mathcal{S}_{\pi_q(i,k)}(i)],\ \forall i \in \mathcal{I}, q \in \mathcal{Q}, k \in \mathcal{K}. \tag{4.20}$$

Similar to equation (4.10), a constant is used in the place of all but a single random variable, $\xi(i)$, but we use a separate approximation for each assigned cell, $k \in \mathcal{K}$, and for each permutation, $q \in \mathcal{Q}$. While we assume the set $\mathcal{Q}$ consists of all $(K!)^{d-1}$ cell assignments in the characterization of the LHS estimator's second moment in Section 4.3.2.1, as we describe in Section 4.3.2.5, a smaller collection replaces $\mathcal{Q}$ in our numerical experiments.

We note that the right-hand side of equation (4.18) depends on the index i, but the left-hand side does not. In what follows, we make use of the approximation based on component i when attempting to optimize the location of the strata for component i. In this same sense, h_{ikq} denotes an approximation of the *entire* function h, with a focus on the component i, rather than the i-th term in an additive approximation, as in the approximation of Section 4.3.1. We also note that for the purpose of optimizing, we can drop the constant $K^{d-1}/(K!)^{d-1}$ term in equation (4.18).

In the context of approximation (4.18), we make the following approximations:

$$\mathbb{E}[h(\xi^k)|\pi = \pi_q] \approx \mathbb{E}[h_{ikq}(\xi^k(i))|\pi = \pi_q],\ \forall i \in \mathcal{I},\ k \in \mathcal{K},\ q \in \mathcal{Q}, \tag{4.21}$$

and

$$\begin{aligned}\mathbb{E}[(h(\xi^k))^2|\pi = \pi_q] \approx\ & \mathbb{E}[(h_{ikq}(\xi^k(i)))^2|\pi = \pi_q] \\ & + \sum_{\substack{i' \in \mathcal{I} \\ i' \neq i}} \left(\frac{d}{d\xi(i')} h_{i'kq}(a_{kq}(i')) \right)^2 \mathbb{V}\mathrm{ar}[\xi(i')|\xi(i') \in \mathcal{S}_{\pi_q(i',k)}], \\ & \forall i \in \mathcal{I},\ k \in \mathcal{K},\ q \in \mathcal{Q}. \end{aligned} \tag{4.22}$$

The form of these approximations is motivated by the fact that they are exact when h is a linear function, as shown in Proposition 4.3.4. Moreover, even though, in general,

we have a different approximation for each component i, they are all identical when h is linear.

Proposition 4.3.4. *Let $(\pi(i,1), \pi(i,2), \ldots, \pi(i,K))$ denote a random permutation of $\{1, 2, \ldots, K\}$, for $i \in \mathcal{I}$, and let π_q, $q \in \mathcal{Q}$, denote the set of all $(K!)^{d-1}$ possible collections of d permutations that assign the K cells in an LHS design. Let $\Gamma^k = \mathcal{S}_{\pi(1,k)}(1) \times \mathcal{S}_{\pi(2,k)}(2) \times \cdots \times \mathcal{S}_{\pi(d,k)}(d)$, and let*

$$\mathbb{P}[\xi \in \Gamma^k] = \prod_{i \in \mathcal{I}} \mathbb{P}[\xi(i) \in \mathcal{S}_{\pi(i,k)}(i)], \ \forall k \in \mathcal{K}.$$

Define:

$$h_K^{LHS} = \sum_{k \in \mathcal{K}} K^{d-1} \mathbb{P}[\xi \in \Gamma^k] h(\xi^k),$$

$$h(\xi) = \sum_{i \in \mathcal{I}} c_i \xi(i),$$

in which $c_i \in \mathbb{R}$, $\forall i \in \mathcal{I}$, and define

$$h_{ikq}(\xi(i)) = h(a_{kq}(1), \ldots, a_{kq}(i-1), \xi(i), a_{kq}(i+1), \ldots, a_{kq}(d)),$$

in which $a_{kq}(i) = \mathbb{E}[\xi(i) | \xi(i) \in \mathcal{S}_{\pi(i,k)}(i)]$, $\forall i \in \mathcal{I}$. Then,

(i) $\quad \mathbb{E}[h(\xi^k) | \pi = \pi_q] = \mathbb{E}[h_{ikq}(\xi^k) | \pi = \pi_q], \quad \forall i \in \mathcal{I}, \ k \in \mathcal{K}, \ q \in \mathcal{Q}, \ \text{and}$

(ii) $\mathbb{E}[(h(\xi^k))^2 | \pi = \pi_q] = \mathbb{E}[(h_{ikq}(\xi^k))^2 | \pi = \pi_q] + \sum_{\substack{i' \in \mathcal{I} \\ i' \neq i}} \left(c_{i'}^2 \mathbb{V}ar[\xi(i') | \xi(i') \in \mathcal{S}_{\pi_q(i',k)}(i')] \right),$

$$\forall i \in \mathcal{I}, \ k \in \mathcal{K}, \ q \in \mathcal{Q}.$$

Proof:

(i) For a linear function, the expectation is

$$\begin{aligned}\mathbb{E}[h(\xi^k)|\pi=\pi_q] &= \mathbb{E}\left[\sum_{i\in\mathcal{I}} c_i\xi^k(i)\middle|\,\pi=\pi_q\right]\\ &= \sum_{i\in\mathcal{I}} c_i\mathbb{E}\left[\xi^k(i)|\pi=\pi_q\right]\\ &= \sum_{i\in\mathcal{I}} c_i a_{kq}(i),\end{aligned}$$

and the approximation in equations (4.19)-(4.20) yields

$$\begin{aligned}\mathbb{E}[h_{ikq}(\xi^k)|\pi=\pi_q] &= \mathbb{E}\left[c_i\xi^k(i)+\sum_{\substack{i'\in\mathcal{I}\\ i'\neq i}} c_{i'}a_{kq}(i')\middle|\,\pi=\pi_q\right]\\ &= \mathbb{E}\left[c_i\xi^k(i)|\pi=\pi_q\right]+\sum_{\substack{i'\in\mathcal{I}\\ i'\neq i}} c_{i'}a_{kq}(i')\\ &= \sum_{i\in\mathcal{I}} c_i a_{kq}(i)\\ &= \mathbb{E}[h(\xi^k)|\pi=\pi_q].\end{aligned}$$

(ii) The second moment of an LHS sample for a given randomly generated LHS cell assignment is

$$\begin{aligned}\mathbb{E}[(h(\xi^k))^2|\pi=\pi_q] &= \mathbb{E}\left[\left(\sum_{i\in\mathcal{I}} c_i\xi^k(i)\right)^2\middle|\,\pi=\pi_q\right]\\ &= \sum_{i\in\mathcal{I}} c_i^2\mathbb{E}\left[(\xi^k(i))^2|\pi=\pi_q\right]\\ &\qquad + 2\sum_{i\in\mathcal{I}}\sum_{\substack{i'\in\mathcal{I}\\ i'>i}} c_ic_{i'}\mathbb{E}[\xi^k(i)\cdot\xi^k(i')|\pi=\pi_q]\\ &= \sum_{i\in\mathcal{I}} c_i^2\mathbb{E}\left[(\xi^k(i))^2\middle|\pi=\pi_q\right]+2\sum_{i\in\mathcal{I}}\sum_{\substack{i'\in\mathcal{I}\\ i'>i}} c_ic_{i'}a_{kq}(i)a_{kq}(i'),\end{aligned}$$

while the second moment of the approximation given π_q is

$$\begin{aligned}\mathbb{E}[(h_{ikq}(\xi^k))^2|\pi=\pi_q] &= \mathbb{E}\left[\left(c_i\xi^k(i)+\sum_{\substack{i'\in\mathcal{I}\\ i'\neq i}} c_{i'}a_{kq}(i')\right)^2\middle|\pi=\pi_q\right]\\ &= c_i^2\mathbb{E}\left[(\xi^k(i))^2|\pi=\pi_q\right]+\sum_{\substack{i'\in\mathcal{I}\\ i'\neq i}} c_{i'}^2a_{kq}^2(i')\\ &\quad + 2\sum_{i\in\mathcal{I}}\sum_{\substack{i'\in\mathcal{I}\\ i'>i}} c_ic_{i'}a_{kq}(i)a_{kq}(i').\end{aligned}$$

This implies

$$\mathbb{E}[(h_{ikq}(\xi^k))^2|\pi=\pi_q] = \mathbb{E}[(h(\xi^k))^2|\pi=\pi_q] - \sum_{\substack{i'\in\mathcal{I}\\ i'\neq i}} c_{i'}^2\left(\mathbb{E}\left[(\xi^k(i'))^2|\pi=\pi_q\right]-a_{kq}^2(i')\right). \quad \blacksquare$$

Corollary 4.3.5 shows that minimizing the right-hand side of equation (4.15) via the approximations in equations (4.22)-(4.21) minimizes the mean squared error of the resulting LHS estimator with respect to $\mathbb{E}[h(\xi)]$ for a linear function.

Corollary 4.3.5. *Let $(\pi(i,1),\pi(i,2),\ldots,\pi(i,K))$ denote a random permutation of $\{1,2,\ldots,K\}$, for $i\in\mathcal{I}$, and let π_q, $q\in\mathcal{Q}$, denote the set of all $(K!)^{d-1}$ possible collections of d permutations that assign the K cells in an LHS design. Let $\Gamma^k=\mathcal{S}_{\pi(1,k)}(1)\times\mathcal{S}_{\pi(2,k)}(2)\times\cdots\times\mathcal{S}_{\pi(d,k)}(d)$, and let*

$$\mathbb{P}[\xi\in\Gamma^k]=\prod_{i\in\mathcal{I}}\mathbb{P}[\xi(i)\in\mathcal{S}_{\pi(i,k)}(i)],\ \forall k\in\mathcal{K}=\{1,2,\ldots,K\}.$$

Define:

$$h_K^{LHS}=\sum_{k\in\mathcal{K}}K^{d-1}\mathbb{P}[\xi\in\Gamma^k]h(\xi^k)$$

$$p_{kq} = \mathbb{P}[\xi \in \Gamma^k | \pi = \pi_q]$$
$$\mu_{kq} = \mathbb{E}[h(\xi^k) | \pi = \pi_q]$$
$$h(\xi) = \sum_{i \in \mathcal{I}} c_i \xi(i),$$

in which $c_i \in \mathbb{R},\ \forall i \in \mathcal{I}$,

$$h_{ikq}(\xi(i)) = h(a_{kq}(1), \ldots, a_{kq}(i-1), \xi(i), a_{kq}(i+1), \ldots, a_{kq}(d)),$$

in which $a_{kq} = \mathbb{E}[\xi(i') | \xi(i') \in \mathcal{S}_{\pi(i',k)}(i')],\ \forall i' \in \mathcal{I} \setminus \{i\}$. *Then, minimizing the quantity*

$$\frac{K^{2d-2}}{(K!)^{d-1}} \sum_{q \in \mathcal{Q}} \sum_{k \in \mathcal{K}} (p_{kq})^2 \cdot \left(\mathbb{E}[(h_{ikq}(\xi^k(i)))^2 | \pi = \pi_q] + \sum_{\substack{i' \in \mathcal{I} \\ i' \neq i}} c_{i'}^2 \mathbb{V}ar[\xi(i') | \xi(i') \in \mathcal{S}_{\pi_q(i',k)}] \right) + \frac{2 \cdot K^{2d-2}}{(K!)^{d-1}} \sum_{q \in \mathcal{Q}} \sum_{k \in \mathcal{K}} \sum_{\substack{k' \in \mathcal{K} \\ k' > k}} (p_{kq} \cdot p_{k'q} \cdot \mu_{kq} \cdot \mu_{k'q}) \tag{4.23}$$

minimizes the mean squared error of the estimator h_K^{LHS}.

Proof:

Proposition 4.3.3 proves that the objective function in (4.23) is equivalent to the right-hand side of equation (4.15), and Proposition 4.3.4 proves that the function is therefore equivalent to $\mathbb{E}[(h_K^{LHS})^2]$. Iman and Conover (1980) show that h_K^{LHS} is an unbiased estimator of $\mathbb{E}[h(\xi)]$, so (i) minimizing $\mathbb{E}[(h_K^{LHS})^2]$ minimizes

$$\mathbb{E}[(h_K^{LHS})^2] - (\mathbb{E}[h_K^{LHS}])^2 = \mathbb{E}[(h_K^{LHS})^2] - (\mathbb{E}[h(\xi)])^2 = \mathbb{V}\text{ar}[h_K^{LHS}],$$

and, in turn, (ii) $\mathbb{V}\text{ar}[h_K^{LHS}]$ is equivalent to the mean squared error of h_K^{LHS}. ■

4.3.2.3 Nonlinear Programming Formulation

The terms of the LHS estimator's second moment in the right-hand side of equation (4.15) do not separate by stratum, which precludes the use of a dynamic program similar to the model in Section 4.3.1. We obtain an LHS design by finding the minimum-variance strata for one random variable at a time while assuming the breakpoints defining the cells for the other random components are fixed. This leads to iteratively solving d separate optimization models, which we describe below.

Sets

$i \in \mathcal{I} = \{1, \ldots, d\}$:	random variables
$k \in \mathcal{K} = \{1, \ldots, K\}$:	strata
$q \in \mathcal{Q}$:	cell assignments, i.e., collections of d permutations of $\mathcal{K}$
$q \in \hat{\mathcal{Q}} \subseteq \mathcal{Q}$:	selected collections of permutations

Functions

h:	a multivariate function $h : \mathbb{R}^d \to \mathbb{R}$
h_{ikq}:	a univariate function, as defined in equations (4.19)-(4.20)
$\dot{h}_{ikq}$:	$\frac{dh_{ikq}}{d\xi(i)}$

Data

$\pi_q(i,k)$:	cell assigned to variable i, observation k in permutation q
$\mathcal{S}_k(i)$:	$(F^{-1}(b_{\pi_q(i',k)-1}(i')), F^{-1}(b_{\pi_q(i',k)}(i'))$
$a_{kq}(i)$:	$\mathbb{E}[\xi(i) \mid \xi(i) \in \mathcal{S}_{\pi_q(i,k)}(i)]$
f_i:	marginal pdf of ξ with respect to component i
F_i:	marginal cdf of ξ with respect to component i
p_{ikq}:	$\prod_{\substack{i' \in \mathcal{I} \\ i' \neq i}} \mathbb{P}[\xi(i') \in \mathcal{S}_{\pi_q(i',k)}(i')]$
$u_{kq}(i)$:	$\sum_{\substack{i' \in \mathcal{I} \\ i' \neq i}} (\dot{h}_{i'kq}(a_{kq}(i')))^2 \cdot \mathbb{V}\mathrm{ar}[\xi(i') \mid \xi(i') \in \mathcal{S}_{\pi_q(i',k)}(i')]$

Decision Variables

$b_k(i)$:	selection of breakpoint k in the interval [0,1] used to create strata on random variable i
$\mu_{kq}(i)$:	$\mathbb{E}[h_{ikq}(\xi(i)) \,\vert\, \xi(i) \in \mathcal{S}_{\pi_q(i,k)}(i)]$
$\sigma^2_{kq}(i)$:	$\mathbb{E}[(h_{ikq}(\xi(i)) - \mu_{kq}(i))^2 \,\vert\, \xi(i) \in \mathcal{S}_{\pi_q(i,k)}(i)]$

Boundary Conditions

$b_0(i) = 0$

$b_K(i) = 1$

Formulation

$$\min_{b(i),\mu(i),\sigma^2(i)} \sum_{q\in\hat{\mathcal{Q}}}\sum_{k\in\mathcal{K}} \left(p^2_{kq}(i)\cdot(b_k(i)-b_{k-1}(i))^2\cdot\left(\sigma^2_{kq}(i)+\mu^2_{kq}(i)+u_{kq}(i)\right)\right)$$
$$+ 2\sum_{q\in\hat{\mathcal{Q}}}\sum_{k\in\mathcal{K}}\sum_{\substack{k'\in\mathcal{K}\\k'>k}}\Bigg((b_k(i)-b_{k-1}(i))(b_{k'}(i)-b_{k'-1}(i))$$
$$\cdot p_{kq}(i)\cdot p_{k'q}(i)\cdot\mu_{kq}(i)\cdot\mu_{k'q}(i)\Bigg) \tag{4.24a}$$

$$\text{s.t.}\quad \mu_{kq}(i) = \frac{\int_{F_i^{-1}(b_{k-1}(i))}^{F_i^{-1}(b_k(i))} h_{ikq}(x)f_i(x)dx}{b_k(i)-b_{k-1}(i)}, \forall q\in\hat{\mathcal{Q}}, k\in\mathcal{K} \tag{4.24b}$$

$$\sigma^2_{kq}(i) = \frac{\int_{F_i^{-1}(b_{k-1}(i))}^{F_i^{-1}(b_k(i))} (h_{ikq}(x)-\mu_{kq}(i))^2 f_i(x)dx}{b_k(i)-b_{k-1}(i)}, \forall\ q\in\hat{\mathcal{Q}}, k\in\mathcal{K} \tag{4.24c}$$

$$0 = b_0(i) \le b_1(i) \le \cdots \le b_{K-1}(i) \le b_K(i) = 1. \tag{4.24d}$$

Discussion

Our goal is to obtain strata on random component i that minimizes the objective

function in (4.24a), which, for $\hat{\Omega} = \Omega$, is proportional to the LHS estimator's second moment in equation (4.15), approximated via equations (4.19) and (4.20). Constraints (4.24b)-(4.24c) are analogous to constraints (4.13b)-(4.13c), but use the approximate function h_{ikq}, which sets constants for each other random variable to a cell-specific conditional mean in the place of that variable's mean. The ordering of breakpoints enforced by constraint (4.24d) is identical to that of constraint (4.13d).

4.3.2.4 Coordinate Descent Algorithm

Because model (4.24) is nonconvex, and its objective function has cross terms that involve multiple strata, we obtain solutions via Algorithm 3, a coordinate descent-based heuristic.

4.3.2.5 Permutation Reduction

The number of terms in the objective function in (4.24a) becomes computationally intractable as d and K grow large if we use $|\hat{\Omega}| = |\Omega| = (K!)^{d-1}$, the full set of LHS permutations. This section presents a method to find a small, i.e., $|\hat{\Omega}| \approx K$, collection of permutations that sufficiently cover the multivariate variable space to approximate the true estimator's second moment.

Owen (1992) and Tang (1993) develop LHS designs through the use of orthogonal arrays, which allows for bivariate stratification of the design. Ye (1998) presents orthogonal LHS designs, which consist of a collection of vectors that are each orthogonal to each other, and are used to assign samples to cells; the technique yields designs with low correlation between random components, but can lead to large regions of the

Algorithm 3 Attempts to find breakpoints, $b_k(i)$, to subdivide each random component, $\xi(i), i \in \mathcal{I} = \{1, \ldots, d\}$, into strata to form a minimum-variance LHS estimator, given the number of strata, K, distributions that compose the random vector ξ, and the multivariate function, h, as input. Let $b^\ell,\ \ell \in \mathcal{L}$ denote a finite collection of equidistant candidate breakpoints, such that $b^\ell = (\ell/|\mathcal{L}|)$, and let $\mathcal{K} = \{1, \ldots, K\}$.

procedure CoordinateDescent($h, K, \mathcal{L}, \xi, N$) ▷ function, strata, ▷ candidate breakpoints, random variables, maximum iterations

  $b_k^0(i) = b^{\frac{k|\mathcal{L}|}{K}},\ \forall i \in \mathcal{I}, k \in \mathcal{K}$ ▷ Initialize $b_k(i)$ with uniform strata

  $n \leftarrow 1$

  while $n \leq N$ **do**

    for $i \in \mathcal{I}$ **do**

      $b_0^n(i) \leftarrow b^0,\ b_K^n(i) \leftarrow b^{|\mathcal{L}|}$

      for $k \in \mathcal{K}$ **do**

        $b_k^n(i) \leftarrow \underset{b^\ell, \ell \in \mathcal{L}: b_{k-1}^n(i) < b^\ell < b_{k+1}^{n-1}(i)}{\operatorname{argmin}} \{z^\ell\}$ ▷ z^ℓ is calculated via ▷ the objective function in (4.24a), which, in turn, ▷ is updated with each breakpoint selection

    if $b_k^n(i) = b_k^{n-1}(i),\ \forall i \in \mathcal{I}, k \in \mathcal{K}$ **or** $n = N$ **then**

      return $b_k^n(i),\ i \in \mathcal{I}, k \in \mathcal{K}$

      ▷ algorithm terminates if solution does not change

  return $b_k^n(i),\ i \in \mathcal{I}, k \in \mathcal{K}$

multivariate variable space with no observations. Given d, the techniques of Owen (1992) and Tang (1993) are only feasible for specific values of K, and the orthogonal array-based procedure does not lend itself well to creating multiple complementary designs. van Dam et al. (2007) develop a framework for *maximin* LHS designs in two dimensions, in which the minimum distance between any two points in the LHS design is maximized for a collection of different norms used to define the distance measure. They provide upper bounds on the best possible distance specific for $d = 2$, as well as a mixed-integer programming (MIP) formulation in which the distance is defined as the ℓ_1 norm between two cells. van Dam et al. (2009) provide general upper bounds on the maximin distance for a design when $d \geq 2$, and Husslage et al. (2011) develop designs with an optimal maximin distance for a collection of cases in which $d \leq 10,\ K \leq 300$.

In what follows, we develop a MIP formulation that obtains a maximin LHS design with the ℓ_1 norm as our distance measure. The formulation includes a generalization of the one developed by van Dam et al. (2007) for $d = 2$. To ensure that the collection of LHS designs covers the multivariate variable space, the formulation accepts previous solutions to the optimization model as input, and penalizes common cell assignments to pairs of random variables.

Sets and Indices

$i \in \mathcal{I}$:	components
$k \in \mathcal{K}$:	strata
$j \in \mathcal{J}$:	observations in the LHS design; $\lvert\mathcal{J}\rvert = \lvert\mathcal{K}\rvert$
$\mathcal{P} \subset (\mathcal{I} \times \mathcal{I} \times \mathcal{K} \times \mathcal{K})$:	2-d projections of previous solutions

Data

p_k:	location of stratum k; here, $p_k = k$
w:	weight of maximin distance in objective function

Decision Variables

X_{ijk}:	1 if component i of observation j is assigned to stratum k, 0 o.w.
P_{ij}:	position of component i, observation j
$D_{ijj'}$:	distance between observations j and j' wrt component i
$B_{ijj'}$:	1 if $P_{ij} > P_{ij'}$, 0 o.w.
Z:	minimum ℓ_1 distance between any two observations
$S_{ii'jkk'}$:	1 if components i and i' are assigned to strata k and k', respectively, in observation j; 0 o.w.

Formulation

$$\max \quad wZ - (1-w) \sum_{(i,i',k,k') \in \mathcal{P}} \sum_{j \in \mathcal{J}} S_{ii'jkk'} \tag{4.25a}$$

$$\text{s.t.} \quad \sum_{j \in \mathcal{J}} X_{ijk} = 1, \ \forall i \in \mathcal{I}, k \in \mathcal{K} \tag{4.25b}$$

$$\sum_{k \in \mathcal{K}} X_{ijk} = 1, \ \forall i \in \mathcal{I}, j \in \mathcal{J} \tag{4.25c}$$

$$P_{ij} = \sum_{k \in \mathcal{K}} p_k X_{ijk}, \ \forall i \in \mathcal{I}, j \in \mathcal{J} \tag{4.25d}$$

$$D_{ijj'} \leq P_{ij'} - P_{ij} + 2(K-1) \cdot B_{ijj'}, \ \forall i \in \mathcal{I}, j \in \mathcal{J}, j' \in \mathcal{J} : j < j' \tag{4.25e}$$

$$D_{ijj'} \leq P_{ij} - P_{ij'} + 2(K-1) \cdot (1 - B_{ijj'}), \ \forall i \in \mathcal{I}, j \in \mathcal{J}, j' \in \mathcal{J} : j < j' \tag{4.25f}$$

$$Z \leq \sum_{i \in \mathcal{I}} D_{ijj'}, \ \forall j \in \mathcal{J}, j' \in \mathcal{J} : j < j' \tag{4.25g}$$

$$S_{ii'jkk'} \geq (X_{ijk} + X_{i'jk'}) - 1, \ \forall (i,i',k,k') \in \mathcal{P}, j \in \mathcal{J} \tag{4.25h}$$

$$X_{ijk} \in \{0,1\}, \ \forall i \in \mathcal{I}, j \in \mathcal{J}, k \in \mathcal{K} \tag{4.25i}$$

$$B_{ijj'} \in \{0,1\},\ \forall i \in \mathcal{I}, j \in \mathcal{J}, j' \in \mathcal{J} : j < j' \tag{4.25j}$$

$$S_{ii'kk'} \geq 0,\ \forall (i,i',k,k') \in \mathcal{P} \tag{4.25k}$$

$$P_{ij} \text{ unrestricted},\ \forall i \in \mathcal{I}, j \in \mathcal{J} \tag{4.25l}$$

$$D_{ijj'} \text{ unrestricted},\ \forall i \in \mathcal{I}, j \in \mathcal{J}, j' \in \mathcal{J} : j < j' \tag{4.25m}$$

$$Z \text{ unrestricted} \tag{4.25n}$$

Discussion

Our goal, which we quantify via the objective function in (4.25a), is to obtain an LHS design with the largest possible minimum ℓ_1-norm distance between any two points in the design, and the fewest intersections of cell assignment 2-d projections with previous solutions. Constraints (4.25b)-(4.25c) restrict the assignment of cells to valid Latin hypercube designs. Constraint (4.25d) quantifies the position of each cell assignment via the variable P_{ijk}, and constraints (4.25e) and (4.25f) determine the distance between two cells with respect to component i by placing exact bounds on the distance variable, $D_{ijj'}$ when $P_{ij} < P_{ij'}$ and $P_{ij} > P_{ij'}$, respectively. Constraint (4.25g) sets Z to the minimum ℓ_1 distance between any two cells in the LHS design. Constraint (4.25h) tracks common 2-d projections of cell assignments with those of previous solutions. Constraints (4.25i)-(4.25n) provide binary restrictions and simple bounds.

We obtain a collection of designs by initializing $\mathcal{P} = \varnothing$, and iteratively solving model (4.25) and adding the solution's cell assignments to $\mathcal{P}$ until we have obtained K designs. Figure 4.2 shows the collection of designs obtained by this procedure for $d = 2$ random components and $K = 20$ strata. The procedure obtains full coverage

of the 2-d projection in 20 designs when $d = 2$.

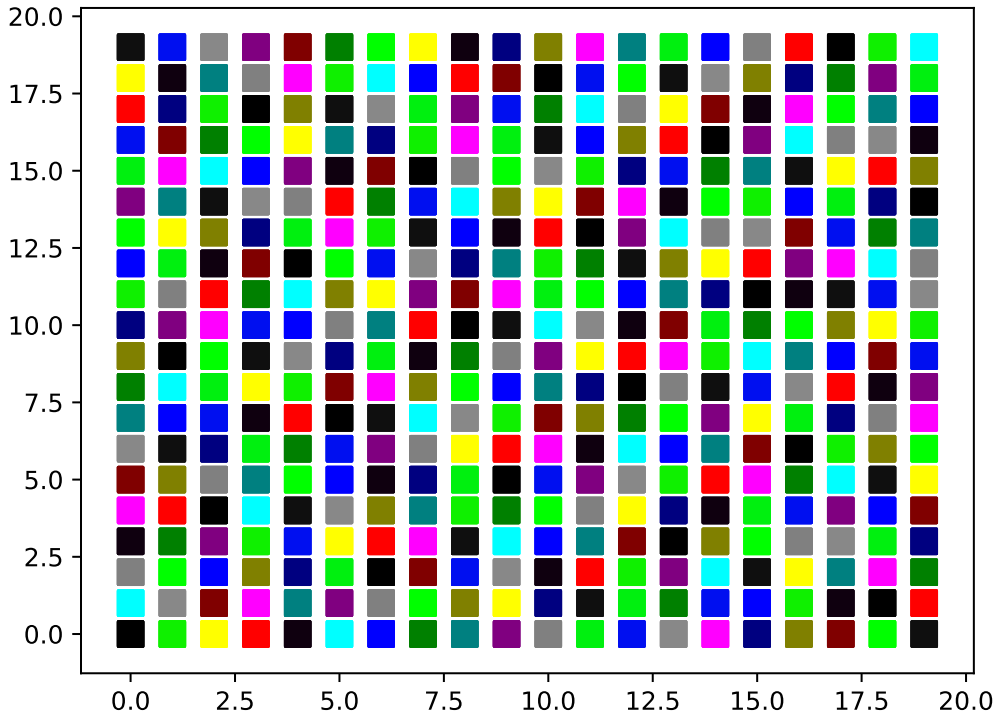

Figure 4.2: LHS cell assignments obtained by iteratively solving model (4.25) and updating the collection of solutions for $d = 2,\ K = 20$. A unique color denotes the cells assigned to each of the LHS designs.

4.4 Results

In this section, we solve model (4.3) for a collection of univariate functions for stratified sampling, we solve model (4.13) d times for a collection of multiplicative functions for LHS, and we generate LHS strata via the heuristic from Section 4.3.2 for a wider collection of multivariate functions. We then compare the sampling error

under the strata we obtain to that of equal-probability strata and, in the case of LHS, to that of naïve Monte Carlo sampling. Finally, we implement our LHS estimator in a maximum reliability path application. The results demonstrate that sampling with strata that are optimized to approximately minimize the variance of the corresponding estimators via the techniques of Sections 4.2 and 4.3 can yield significant variance reduction compared to using equal-probability strata.

4.4.1 Stratified Sampling

We illustrate the potential of our procedure by first applying the dynamic programming procedure of Section 4.2.4 to a collection of univariate functions. We do not perform any Monte Carlo simulation for the results in this section. Rather, we numerically compute the optimal value of model (4.3), which we denote z^*. Then, we compute the same objective function, i.e., the variance of a stratified estimator, using equal-probability strata; we denote that value z^u, and we form the relative efficiency z^u/z^*. All experiments use $K = 10$ cells and $L = 100$ candidate breakpoints, and we note that the ratio we report is independent of the total sample size, N.

Table 4.2 displays the results for a collection of univariate functions that use the notation from Section 4.2.2. Figure 4.3 shows the relative efficiency achieved as a function of the skewness of each univariate function, and suggests that the value of optimized nonuniform strata increases with skewness. Figure 4.4 plots the optimal breakpoints for a subset of the univariate cases, and illustrates the variety of solutions we obtain from different functions of the same random variable. The maximum time to create and solve the dynamic program for a test case was less than one minute, using

a Cray XC40 compute node with two Intel E5-2690 v3 12-core (Haswell) processors and 64 GB of DDR4 memory.

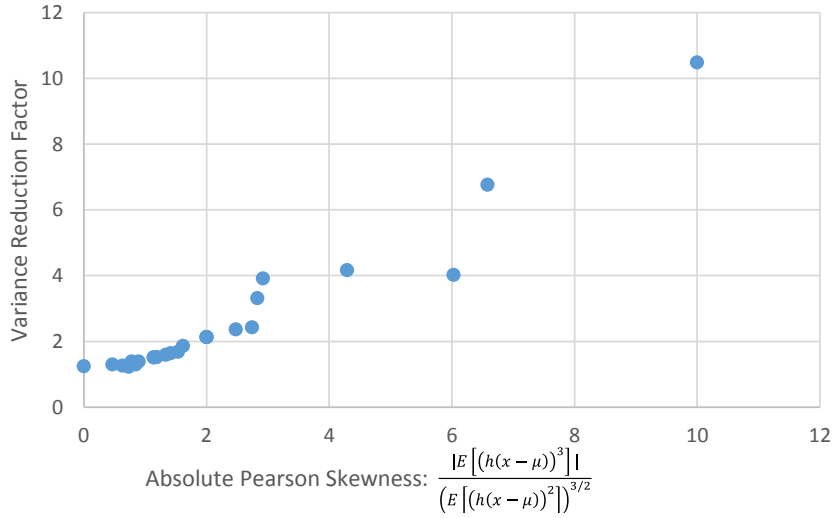

Figure 4.3: Relative efficiency of optimized vs. equal-probability strata, plotted as a function of skewness for the collection of univariate functions given in Table 4.2.

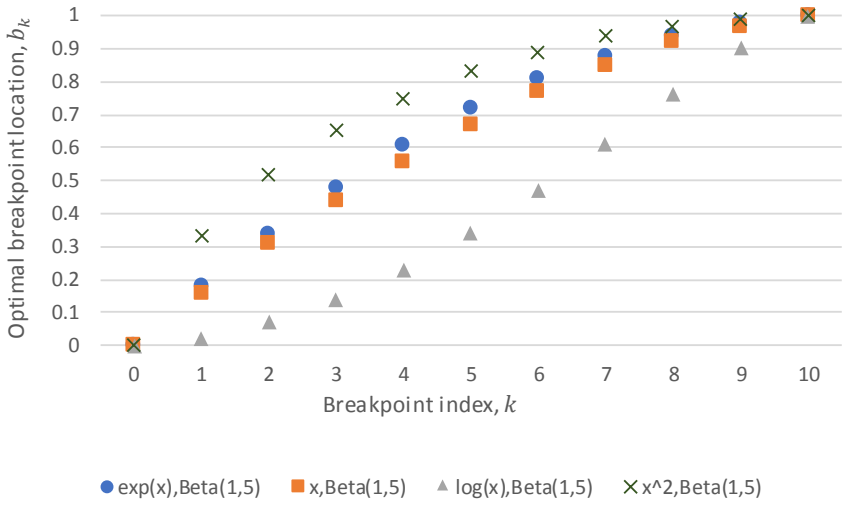

Figure 4.4: Plot of optimal breakpoints for a collection of functions of a Beta(1,5) random variable.

$h(x)$	Distribution	z^u/z^*	$b = (b_0, b_1, \ldots, b_K)$
x	χ^2_1	3.32	[0.00,0.29,0.47,0.61,0.72,0.81,0.88,0.93,0.97,0.99,1.00]
$\log(x)$	χ^2_1	1.69	[0.00,0.02,0.06,0.13,0.22,0.33,0.46,0.61,0.76,0.90,1.00]
x^2	χ^2_1	10.48	[0.00,0.51,0.69,0.79,0.86,0.91,0.94,0.96,0.98,0.99,1.00]
e^x	$N(0,1)$	4.02	[0.00,0.22,0.42,0.59,0.72,0.82,0.89,0.94,0.97,0.99,1.00]
x	$N(0,1)$	1.25	[0.00,0.04,0.12,0.23,0.36,0.50,0.64,0.77,0.88,0.96,1.00]
e^x	Beta(1,5)	1.86	[0.00,0.18,0.34,0.48,0.61,0.72,0.81,0.88,0.94,0.98,1.00]
x	Beta(1,5)	1.52	[0.00,0.16,0.31,0.44,0.56,0.67,0.77,0.85,0.92,0.97,1.00]
$\log(x)$	Beta(1,5)	1.59	[0.00,0.02,0.07,0.14,0.23,0.34,0.47,0.61,0.76,0.90,1.00]
x^2	Beta(1,5)	3.91	[0.00,0.33,0.52,0.65,0.75,0.83,0.89,0.94,0.97,0.99,1.00]
e^x	Beta(5,1)	1.30	[0.00,0.04,0.10,0.18,0.27,0.37,0.48,0.60,0.73,0.86,1.00]
x	Beta(5,1)	1.52	[0.00,0.03,0.08,0.15,0.23,0.33,0.44,0.56,0.69,0.84,1.00]
$\log(x)$	Beta(5,1)	2.13	[0.00,0.01,0.04,0.09,0.16,0.25,0.36,0.49,0.64,0.81,1.00]
x^2	Beta(5,1)	1.23	[0.00,0.04,0.10,0.18,0.27,0.37,0.48,0.60,0.73,0.86,1.00]
x	Exp(1)	2.13	[0.00,0.19,0.36,0.51,0.64,0.75,0.84,0.91,0.96,0.99,1.00]
$\log(x)$	Exp(1)	1.52	[0.00,0.02,0.07,0.15,0.25,0.37,0.51,0.65,0.79,0.92,1.00]
x^2	Exp(1)	6.77	[0.00,0.39,0.59,0.72,0.81,0.88,0.93,0.96,0.98,0.99,1.00]
x	Gamma(2,1)	1.64	[0.00,0.12,0.26,0.41,0.55,0.67,0.78,0.87,0.94,0.98,1.00]
$\log(x)$	Gamma(2,1)	1.38	[0.00,0.02,0.07,0.15,0.26,0.39,0.53,0.67,0.81,0.93,1.00]
x^2	Gamma(2,1)	4.16	[0.00,0.27,0.46,0.61,0.73,0.82,0.89,0.94,0.97,0.99,1.00]
x	Gamma(5,1)	1.39	[0.00,0.09,0.22,0.36,0.50,0.63,0.75,0.85,0.93,0.98,1.00]
$\log(x)$	Gamma(5,1)	1.30	[0.00,0.03,0.10,0.20,0.32,0.45,0.59,0.72,0.84,0.94,1.00]
x^2	Gamma(5,1)	2.36	[0.00,0.17,0.34,0.50,0.64,0.75,0.84,0.91,0.96,0.99,1.00]
e^x	Weibull(2,1)	2.43	[0.00,0.15,0.32,0.48,0.62,0.74,0.84,0.91,0.96,0.99,1.00]
x	Weibull(2,1)	1.26	[0.00,0.09,0.21,0.34,0.47,0.60,0.72,0.82,0.91,0.97,1.00]
$\log(x)$	Weibull(2,1)	1.52	[0.00,0.02,0.07,0.15,0.25,0.37,0.51,0.65,0.79,0.92,1.00]
x^2	Weibull(2,1)	2.13	[0.00,0.19,0.36,0.51,0.64,0.75,0.84,0.91,0.96,0.99,1.00]

Table 4.2: Relative efficiency and optimized strata boundary points for a collection of univariate functions of random variables. Notation z^* and z^u denote the stratified sampling estimator's population variance under optimized and equal-probability strata, respectively.

4.4.2 Nonuniform LHS: Dynamic Programming

To illustrate the impact of our method on reducing the variance of a multivariate LHS estimator, we use repeated experiments, implemented in Python (Rossum 1995) using uniform random variates generated via the WELL512 implementation developed by Panneton et al. (2006) and available in the Stochastic Simulation in Java library created by L'Ecuyer et al. (2002). We use a product function $h(\xi) = \prod_{i \in \mathcal{I}} \xi(i)$, and estimate $\mathbb{E}[h(\xi)]$ for different values of d across a collection of probability distributions.

We consider three estimators in total: (i) naïve Monte Carlo; (ii) LHS with equal-probability cells; and, (iii) LHS as defined in equation (4.9). For estimator (iii), we optimize the strata for each component of ξ by the dynamic programming scheme of Sections 4.2.4 and 4.3.1.

For each of the three estimators, we form M i.i.d. replicates of the estimator which, in turn, use K samples. For example, under the LHS estimator of equation (4.9), we form $\bar{h}_K^m$, $m = 1, 2, \ldots, M$, i.i.d. replicates and form the sample mean estimator

$$\bar{h}_{K,M} = \frac{1}{M} \sum_{m=1}^{M} \bar{h}_K^m. \tag{4.26}$$

Analogous estimators are formed for the uniform LHS estimator and for naïve Monte Carlo, and for all three estimators we use common random numbers, i.e., identical streams of uniform random variates.

We then compare the sample variance of $\bar{h}_{K,M}$ obtained under each method to obtain empirical relative efficiencies. In the results we report, we use $M = 10,000$

replicates, $K = 100$ strata, and $L = 1,000$ candidate breakpoints, and we report 95% confidence intervals (CIs) for the relative efficiencies in Table 4.3 by replicating this experiment 100 times, comparing estimator (iii) with both (i) and (ii). More specifically, our relative efficiencies correspond to the ratio of sample variances of $1,000,000 = 100 \cdot 10,000$ terms, $\bar{h}_K^m$, from the right-hand side of equation (4.26) for estimators (i), (ii), and (iii). We note that while an LHS application would likely use a much lower value for M, and instead increase K according to the computational budget, our sample sizes are inflated to illustrate the value of optimizing the LHS cells in higher dimensions, for which the estimated sampling error can be volatile.

Table 4.3 demonstrates the significant variance reduction offered by the LHS procedure in Section 4.3.1, when compared to LHS with equal-probability strata. The lower bound of the 95% confidence interval estimate for the relative efficiency versus uniform LHS is greater than one for all cases, and it exceeds an order of magnitude for more than half the test cases. Further, the factor tends to grow as the dimension d grows larger. The maximum time to create and solve the dynamic programs for a test case was 15 minutes, using the same computational resources as in Section 4.4.1.

4.4.3 Nonuniform LHS: Coordinate Descent

We use a similar procedure to compare the performance of the LHS estimators formed by selecting a collection of permutations for set Ω by solving model (4.25), then using Ω to find optimized strata via Algorithm 3, which we test by obtaining estimates of relative efficiency versus probability-equal strata using the collection of functions in Mease and Bingham (2006). In these experiments, we use $M = 500$

		Relative efficiency vs. equal-probability strata		Relative efficiency vs. naïve Monte Carlo	
Distribution	d	Point estimate	95% CI half-width (%)	Point estimate	95% CI half-width (%)
χ^2_1	2	129.1	0.9	251.1	1.0
χ^2_1	4	341.4	5.6	372.5	7.3
χ^2_1	6	836.5	12.0	1,477.2	64.6
χ^2_1	8	2,510.1	48.1	3,478.3	43.3
Beta(1,2)	2	5.1	0.7	25.3	0.6
Beta(1,2)	4	6.1	1.2	11.9	1.1
Beta(1,2)	6	7.5	1.8	10.6	1.7
Beta(1,2)	8	9.6	2.9	11.3	3.4
Exponential	2	35.8	0.8	104.3	0.7
Exponential	4	57.8	2.7	77.9	2.8
Exponential	6	97.5	5.7	122.6	16.3
Exponential	8	180.1	14.5	219.0	19.8
Gamma(2,1)	2	9.7	0.7	47.0	0.6
Gamma(2,1)	4	11.7	1.7	22.8	1.5
Gamma(2,1)	6	14.7	2.9	21.1	4.0
Gamma(2,1)	8	19.2	5.0	23.8	7.2
Weibull(2,1)	2	2.2	0.8	17.7	0.6
Weibull(2,1)	4	2.3	1.2	6.8	1.1
Weibull(2,1)	6	2.4	1.7	4.8	1.5
Weibull(2,1)	8	2.6	2.2	4.1	2.4

Table 4.3: Empirically obtained point estimates and 95% CI half-widths of relative efficiencies associated with estimating $\mathbb{E}[h(\xi)]$, in which $h(x) = \prod_{i \in \mathcal{J}} x_i$, using optimized LHS strata obtained by the dynamic programming procedure in Section 4.3.1, compared to LHS with equal-probability strata and naïve Monte Carlo sampling. Confidence intervals were obtained via 100 repeated experiments, each of which use $M = 10,000$ replicates, $K = 100$ strata, and $L = 1,000$ candidate breakpoints for each dynamic programming routine. CI half-width values are reported as a percentage of the corresponding point estimate. The experiments are implemented in Python 3.5.4 using the WELL512 generator developed by Panneton et al. (2006), via the Stochastic Simulation in Java library created by L'Ecuyer et al. (2002). Common random numbers are generated for each experiment; separate substreams are used for permutations and for the uniform random variates used to generate realizations of ξ.

replicates and 20 repeated experiments. Table 4.4 reports the relative efficiency of the LHS estimator compared to uniform strata, and demonstrates that our procedure exhibits improved efficiency across a wide collection of functions for $d = 2$ and $d = 4$,

using the entire set of permutations in the latter case.

	Relative efficiency versus equal-probability strata					
	$K = 4,\ \lvert\hat{\Omega}\rvert = 4$		$K = 20,\ \lvert\hat{\Omega}\rvert = 20$		$K = 3,\ \lvert\hat{\Omega}\rvert = 216$	
	Normal	Exponential(1)	Weibull(0.5,1)	Exponential(1)	Weibull(0.5,1)	Exponential(1)
$h(\cdot)$	d=2	d=2	d=2	d=2	d=4	d=4
$\xi(1)$	1.29	3.62	37.39	10.26	8.77	2.59
$(\xi(1))^2$	1.72	15.13	18.97	34.2	29.15	8.71
$(\xi(1))^3$	3.00	32.46	10.54	23.74	22.91	21.52
$\sin(\xi(1))$	1.30	3.62	41.49	10.26	8.76	2.59
$\sin(3\xi(1))$	3.59	3.6	44.29	10.23	8.15	2.58
$\sum_{i\in\mathcal{I}}\xi(i)$	1.03	1.44	3.01	1.03	1.94	1.13
$\sum_{i\in\mathcal{I}}(\xi(i))^2$	1.10	3.68	10.35	3.19	4.21	1.92
$(\sum_{i\in\mathcal{I}}\xi(i))^2$	1.32	4.56	11.85	1.63	3.38	2.19
$\prod_{i\in\mathcal{I}}(\xi(i))$	1.36	8.78	13.81	10.09	27.63	7.51
$\sin(\sum_{i\in\mathcal{I}}(\xi(i)))$	1.25	1.44	2.86	0.78	1.90	1.13
$\cos(2\sum_{i\in\mathcal{I}}(\xi(i)))$	0.99	1.33	1.78	1.29	1.03	0.97
$\exp(-\sum_{i\in\mathcal{I}}(\lvert\xi(i)-1\rvert))$	1.36	1.19	0.75	0.91	1.16	1.03
$\exp(-\sum_{i\in\mathcal{I}}(\lvert\xi(i)\rvert))$	1.38	1.1	1.2	0.71	1.04	1.03
$\exp(-\sum_{i\in\mathcal{I}}(\lvert\xi(i)-0.5\rvert))$	1	1.19	1.63	0.91	1.15	1.03
$\exp(-\sum_{i\in\mathcal{I}}((\xi(i)-1)^2))$	0.91	1.03	1.57	1.09	1.29	1.09
$\exp(-\sum_{i\in\mathcal{I}}((\xi(i))^2))$	1.01	1.15	2.17	1.52	1.03	0.73
$\exp(-\sum_{i\in\mathcal{I}}((\xi(i)-0.5)^2))$	1.14	1.19	1.24	0.9	1.14	1.03

Table 4.4: Empirically obtained point estimates of relative efficiency associated with estimating $\mathbb{E}[h(\xi)]$, using optimized LHS strata obtained by the coordinate descent procedure in Section 4.3.2, compared to LHS with equal-probability strata for a collection of functions from Mease and Bingham (2006). Point estimates are obtained via 20 repeated experiments, each of which uses $M = 500$ replicates, and $L = 10 \cdot K$ candidate breakpoints for each instance. CI half-width values did not exceed 30% of the point estimate in any case. The experiments are implemented in Python 3.5.4 using the WELL512 generator developed by Panneton et al. (2006), via the Stochastic Simulation in Java library created by L'Ecuyer et al. (2002). Common random numbers are generated for each experiment; separate substreams are used for permutations and for the uniform random variates used to generate realizations of ξ.

4.4.4 Application: Maximum Reliability Path

We implement our procedure in a maximum reliability path application, using a network structure from Avramidis et al. (1991) which we show in Figure 4.5. In our model, the decision maker must traverse a path from node 1 to node 9, and there is a chance of detection when each arc in the graph is traversed. The decision maker

seeks a path with the maximum probability of successfully evading detection. We use optimized LHS, uniform LHS, and naïve Monte Carlo sampling to estimate the probability of successfully traversing the network with knowledge of the realized arc detection probabilities prior to path selection.

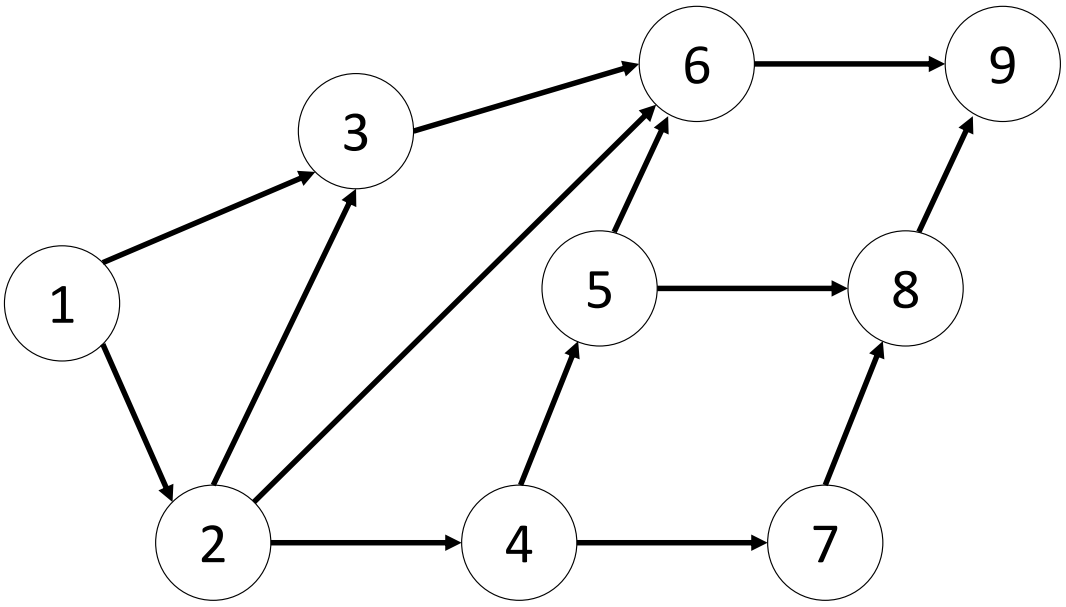

Figure 4.5: Network structure for maximum reliability application. The probability of detection at each arc is distributed as a beta(1,b) random variable.

Table 4.5 reports the estimated probability of success and expected value of information using optimized LHS, uniform LHS, and naïve Monte Carlo sampling. In this application, we use K=50 strata and M=200 replicates, versus $10,000$ simple Monte Carlo samples. We develop the arc success probabilities for each instance as follows. A beta(1,b) random variable governs the chance of detection at each arc, in which $b = 2$ for arcs (1,3), (3,6), and (6,9). For all other arcs, b is provided by instance in the left-most column of Table 4.5. Optimized LHS designs were developed using

the dynamic programming procedure in Section 4.3.1, with $h(\cdot)$ equal to the product of the evasion probabilities of arcs in the maximum reliability path for an instance in which all arc probabilities are equal to their means. The results show that the optimized LHS designs offer a more efficient estimator than uniform LHS and simple Monte Carlo. The benefits of optimized strata are reduced as b approaches 2 for the arcs outside of the approximation function, as the arcs included in the product for $h(\cdot)$ become less likely to be present in the chosen path for a randomly generated graph.

4.5 Conclusion

We have presented a method of minimizing the variance of a stratified sampling estimator by formulating a nonlinear program, and solving this problem exactly via a dynamic program using a discrete set of candidate stratum boundary points. We extend this technique to the multivariate setting and reduce the variance of an LHS estimator compared to equal-probability strata, using an approximation of the estimator's variance and solving a dynamic program for each random component. Finally, we have detailed empirical results that exhibit significant LHS variance reduction under this technique, compared to that of equal-probability strata and naïve Monte Carlo sampling.

	Probability of success							
	Optimized LHS		Uniform LHS		Simple Monte Carlo			
b	Mean	Variance	Mean	Variance	Mean	Variance	$\frac{\text{Uniform LHS Var}}{\text{Optimized LHS Var}}$	$\frac{\text{Simple MC Var}}{\text{Optimized LHS Var}}$
30	3.79E-02	7.31E-06	3.79E-02	2.83E-05	3.65E-02	3.06E-03	3.87	418.68
20	3.79E-02	7.27E-06	3.80E-02	2.82E-05	3.66E-02	3.06E-03	3.89	420.76
10	3.83E-02	6.96E-06	3.84E-02	2.68E-05	3.69E-02	3.04E-03	3.85	436.98
9	3.85E-02	6.79E-06	3.85E-02	2.60E-05	3.70E-02	3.04E-03	3.83	447.22
8	3.88E-02	6.50E-06	3.88E-02	2.50E-05	3.72E-02	3.03E-03	3.85	466.10
7	3.92E-02	6.10E-06	3.92E-02	2.38E-05	3.75E-02	3.02E-03	3.91	495.25
5	3.99E-02	5.63E-06	3.99E-02	2.23E-05	3.81E-02	3.01E-03	3.96	534.88
4.5	4.11E-02	5.39E-06	4.10E-02	2.03E-05	3.92E-02	3.01E-03	3.77	557.92
4	4.21E-02	5.31E-06	4.19E-02	1.88E-05	4.01E-02	3.03E-03	3.54	569.77
3.5	4.34E-02	5.37E-06	4.32E-02	1.72E-05	4.14E-02	3.08E-03	3.20	572.41
3	4.54E-02	5.54E-06	4.52E-02	1.56E-05	4.33E-02	3.18E-03	2.82	574.34
2.75	4.84E-02	6.49E-06	4.81E-02	1.45E-05	4.62E-02	3.38E-03	2.23	520.83
2.5	5.05E-02	7.33E-06	5.02E-02	1.43E-05	4.82E-02	3.55E-03	1.95	485.17
2.25	5.32E-02	8.49E-06	5.30E-02	1.45E-05	5.09E-02	3.83E-03	1.71	450.44
2.125	5.90E-02	1.19E-05	5.88E-02	1.69E-05	5.67E-02	4.48E-03	1.42	376.46
2	6.16E-02	1.35E-05	6.15E-02	1.83E-05	5.93E-02	4.79E-03	1.36	355.57

Table 4.5: Empirically obtained point estimates of mean and variance of the expected maximum likelihood of evading detection when traversing the graph in Figure 4.5, using $K = 50$ optimized LHS strata obtained by the dynamic programming procedure in Section 4.3.1, compared to LHS with equal-probability strata and simple Monte Carlo. The probability of evading detection at each arc is distributed as a beta(1,b) random variable, with $b = 2$ for arcs (1,3), (3,6), and (6,9), and b for all other arcs is provided in the left-most column. The approximation function, $h(\cdot)$, used for determining optimized strata is the product of the evasion probabilities for arcs (1,3), (3,6), and (6,9). Point estimates are obtained via $M = 200$ replicates, using $L = 10 \cdot K$ candidate breakpoints for each instance. The experiments are implemented in Python 3.5.4 using the WELL512 generator developed by Panneton et al. (2006), via the Stochastic Simulation in Java library created by L'Ecuyer et al. (2002). Common random numbers are generated for each experiment; separate substreams are used for permutations and for the uniform random variates used to generate realizations of ξ.

Chapter 5

Future Work

In Chapter 2, we propose a partitioning scheme that approximates constraints containing bilinear terms with a tighter relaxation than similar formulations in the literature. Further, we develop a decomposition method that separates the problem into smaller subproblems, using Lagrangian relaxation to obtain a lower bound and fixing inventory levels at regular intervals to obtain an upper bound. We present empirical results that demonstrate that our techniques significantly improve on alternative approaches. Future work may include the exploration of additional applications for our partitioning scheme, such as the pooling problem; additionally, we may explore alternative applications for the decomposition technique we discuss, such as long-term staffing and scheduling problems.

In Chapter 3, we present a test case of the microgrid design and dispatch problem under load and PV uncertainty, and preliminary results show that optimal designs for such a model are more likely to meet load requirements that solving a deterministic model with a rigid soldier schedule. Future work will include: (i) developing and solving instances for a collection of additional locations; and, (ii) introducing uncertainty in the plug loads in each building on the base, in addition to the uncertainty in ECU loads that we model.

In Chapter 4, we present a methodology that optimizes the stratification of an LHS estimator, and empirical results show that the strata we obtain exhibit improved variance for a collection of sample multivariate functions, including an application to the maximum reliability path problem. Topics of future research include the exploration of new applications of the LHS design procedures in Sections 4.3.1 and 4.3.2 to problems in the literature. In particular, we are interested in investigating the impact of optimizing nonuniform LHS on reducing the bias and variance associated with the estimator for the cost of an optimal solution to stochastic programming problems, such as those in Freimer et al. (2012), Stockbridge and Bayraksan (2016), and Bayraksan (2018), and comparing the performance of our method to that of the variance reduction techniques the authors use. It would be interesting to explore conditions under which either procedure in Section 4.3 guarantees variance reduction compared to LHS using equal-probability cells, for both equation (4.10) and for other approximations. Finally, extension of the methods in Section 4.3 to design optimal LHS strata with the independence assumption relaxed is an opportunity for further research.

Appendices

Appendix A

Microgrid Design and Dispatch Problem

A.1 Microgrid Design and Dispatch Problem

In this section, we describe our adaptation of the microgrid design and dispatch problem of Scioletti et al. (2017), the mapping of this detailed notation to that used in the main text, and the constraints we add to develop model ($\mathcal{P}$). The model notation below follows that of Scioletti et al., to which we refer the reader for a more detailed discussion of the model.

A.1.1 Full ($\mathcal{P}$) Formulation

Sets

$t \in \mathcal{T}$	time periods
$\ell \in \mathcal{L} = \{1, 2, \ldots, \lvert\mathcal{L}\rvert\}$	time blocks indexing a partition of $\mathcal{T}$; i.e., $\cup_{\ell \in \mathcal{L}} \mathcal{T}_\ell = \mathcal{T}$ and $\mathcal{T}_\ell \cap \mathcal{T}_{\ell'} = \emptyset$, $\ell \neq \ell'$
$j \in \mathcal{J}$	battery and generator technologies
$g \in \mathcal{G} \subset \mathcal{J}$	generator technologies
$b \in \mathcal{B} \subset \mathcal{J}$	battery technologies
$s \in \mathcal{S}$	PV panel types
$k \in \tilde{\mathcal{J}}_j \subset \mathcal{J}$	identical twins of technology j, given by size, type, and manufacturer
$k \in \tilde{\mathcal{G}}_g \subset \mathcal{G}$	generator twins of type g
$k \in \tilde{\mathcal{B}}_b \subset \mathcal{B}$	battery twins of type b

Timing Parameters

τ	length of one time period [hours]
υ	number of time periods per block [hours]
ν	ratio of base operation duration to time horizon length [fraction]

Optimization Model Penalty Parameters

$\tilde{c}_j$	cost of procuring one twin of technology type j [\$/twin]
c_s	cost of procuring one panel of technology type s [\$/panel]
δ_t^f	fuel cost penalty in time period t [\$/gal]
ε_j	cycle cost penalty for technology type j [\$/(hours, cycles)]

Power System Parameters

d_t^P	power demand in time period t [W]
$\bar{k}$	overage load coefficient [fraction]
k^s	spinning reserve required relative to PV power [fraction]

Technology Parameters

η_j^+, η_j^-	electric efficiency of power into and out of technology type j, respectively [fraction]
$\underline{p}_j, \bar{p}_j$	minimum and maximum power rating, respectively, of technology type j [W]

Generator Parameters

b_g^f, c_g^f	fuel consumption coefficients for generator g $[\frac{gal}{W^2h}, \frac{gal}{Wh}, \frac{gal}{h}]$

PV Parameters

γ_{st}	power output of PV technology type s in time period t, based on solar irradiance $[\frac{W}{panel}]$
$\bar{n}_s$	maximum allowable number of PV panels of technology type s [panels]

Battery Parameters

a_b^v, b_b^v	battery b voltage model coefficients [V]
d_b^{soc}, a_b^{soc}	battery b lifetime model coefficients [unitless]
b_b^0	battery b state-of-charge used in initial condition constraints [fraction]
c_b^{ref}	battery b manufacturer-specified capacity [Ah]
c_b^+, c_b^-	battery b charge and discharge capacity rate coefficients, respectively [hours]
r_b^{int}	battery b internal resistance [Ohms]
i_b^{avg}	typical current expected from battery b for both charge and discharge activities [A]
$\underline{s}_b, \bar{s}_b$	battery b state-of-charge minimum and maximum operational bounds, respectively [fraction]
i_b^{L+}, i_b^{U+}	battery b charge current lower and upper bound, respectively [A]
i_b^{L-}, i_b^{U-}	battery b discharge current lower and upper bound, respectively [A]

where, for our application, the above parameter values are computed as:

$$\begin{aligned} i_b^{L+} &= 0, \quad \forall b \in \mathcal{B} \\ i_b^{U+} &= \frac{c_b^{ref}}{c_b^+}, \quad \forall b \in \mathcal{B}. \\ i_b^{L-} &= 0, \quad \forall b \in \mathcal{B} \\ i_b^{U-} &= \frac{c_b^{ref}}{c_b^- + \tau}, \quad \forall b \in \mathcal{B} \end{aligned}$$

Continuous Variables

Power Variables

P_{jkt}^+, P_{jkt}^-	aggregate power into and out of technology type j, twin k in time period t, respectively [W]

P_{st}^{PV} aggregate power out of PV technology type s in time period t [W]

Generator Variables

$\tilde{F}_t$ amount of fuel used in time period t [gal]

Battery Variables

B_{bkt}^{soc} state-of-charge of battery type b, twin k at end of time period t [fraction]

I_{bkt}^{+}, I_{bkt}^{-} battery b, twin k current for charge and discharge, respectively, in time period t [A]

R battery inventory at the beginning and end of each block [Ah]

R^{ℓ} battery inventory at the beginning and end of block ℓ [Ah]

Boundary Condition

$B_{bk,0}^{soc}$ initial state-of-charge of battery type b, twin k [fraction]

Binary and Integer Variables

Power Procurement Variables

W_{jk} 1 if technology j, twin k is procured, 0 otherwise

X_s integer number of PV panels of technology type s procured [panels]

W_{jk}^{ℓ} 1 if technology j, twin k is procured in block ℓ, 0 otherwise

X_s^{ℓ} integer number of PV panels of technology type s procured in block ℓ [panels]

Generator Variables

G_{gkt} 1 if technology type g, twin k is operating in time period t, 0 otherwise

Battery Variables

B_{bkt}^{+} 1 if battery type b, twin k is charging in time period t, 0 otherwise

B^-_{bkt}	1 if battery type b, twin k is discharging in time period t, 0 otherwise

Problem $(\mathcal{P})$

Objective Function:

Minimize

$$\frac{1}{|\mathcal{L}|}\left(\sum_{\ell\in\mathcal{L}}\sum_{j\in\mathcal{J}}\sum_{k\in\tilde{\mathcal{J}}_j}\tilde{c}_j W^{\ell}_{jk}+\sum_{\ell\in\mathcal{L}}\sum_{s\in\mathcal{S}}c_s X^{\ell}_{s}\right)+\nu\sum_{t\in\mathcal{T}}\delta^f_t\tilde{F}_t+\nu\tau\sum_{g\in\mathcal{G}}\sum_{k\in\tilde{\mathcal{G}}_g}\sum_{t\in\mathcal{T}}\varepsilon_g G_{gkt}+$$
$$\nu\tau\sum_{b\in\mathcal{B}}\sum_{k\in\tilde{\mathcal{B}}_b}\sum_{t\in\mathcal{T}}\varepsilon_b\left(a^{soc}_b\frac{I^+_{bkt}+I^-_{bkt}}{2c^{ref}_b}-d^{soc}_b\frac{Z^+_{bkt}+Z^-_{bkt}}{2c^{ref}_b}\right) \tag{A.1}$$

subject to

System Operations:

$$\sum_{j\in\mathcal{J}}\sum_{k\in\tilde{\mathcal{J}}_j}\eta^-_j P^-_{jkt}-\sum_{b\in\mathcal{B}}\sum_{k\in\tilde{\mathcal{B}}_b}P^+_{bkt}+\sum_{s\in\mathcal{S}}P^{PV}_{st}\geq(1+\bar{k})d^P_t,\quad\forall t\in\mathcal{T} \tag{A.2a}$$

$$\sum_{b\in\mathcal{B}}\sum_{k\in\tilde{\mathcal{B}}_b}\eta^-_b\bar{p}_b B^{soc}_{bkt}+\sum_{g\in\mathcal{G}}\sum_{k\in\tilde{\mathcal{G}}_g}\left(\bar{p}_g G_{gkt}-P^-_{gkt}\right)\geq k^s\sum_{s\in\mathcal{S}}P^{PV}_{st},\quad\forall t\in\mathcal{T} \tag{A.2b}$$

$$W^{\ell}_{j,k-1}\geq W^{\ell}_{jk},\quad\forall j\in\mathcal{J},k\in\tilde{\mathcal{J}}_j:k>1,\ \ell\in\mathcal{L} \tag{A.2c}$$

Generator Operations:

$$\underline{p}_g G_{gkt}\leq P^-_{gkt}\leq\bar{p}_g G_{gkt},\quad\forall g\in\mathcal{G},k\in\tilde{\mathcal{G}}_g,t\in\mathcal{T} \tag{A.3a}$$

$$\tilde{F}_t\geq\tau\sum_{g\in\mathcal{G}}\sum_{k\in\tilde{\mathcal{G}}_g}(b^f_g P^-_{gkt}+c^f_g G_{gkt}),\quad\forall t\in\mathcal{T} \tag{A.3b}$$

$$G_{gkt}\leq W^{\ell}_{gk},\quad\forall g\in\mathcal{G},k\in\tilde{\mathcal{G}}_g,t\in\mathcal{T}_\ell,\ \ell\in\mathcal{L} \tag{A.3c}$$

$$G_{g,k-1,t}\leq G_{gkt},\quad\forall g\in\mathcal{G},k\in\tilde{\mathcal{G}}_g,t\in\mathcal{T}:k>1 \tag{A.3d}$$

$$P^-_{g,k-1,t} \leq P^-_{gkt}, \quad \forall g \in \mathcal{G}, k \in \tilde{\mathcal{G}}_g, t \in \mathcal{T} : k > 1 \tag{A.3e}$$

PV Operations:

$$P^{PV}_{st} \leq \gamma_{st} X^\ell_s, \quad \forall s \in \mathcal{S}, t \in \mathcal{T}_\ell, \ \ell \in \mathcal{L} \tag{A.4a}$$

$$X^\ell_s \leq \bar{n}_s, \quad \forall s \in \mathcal{S}, \ \ell \in \mathcal{L} \tag{A.4b}$$

Battery Operations:

$$P^+_{bkt} = a^v_b B^{soc}_{bk,t-1} I^+_{bkt} + (b^v_b + i^{avg}_b r^{int}_b) I^+_{bkt} \quad \forall b \in \mathcal{B}, k \in \tilde{\mathcal{B}}_b, t \in \mathcal{T} \tag{A.5a}$$

$$P^-_{bkt} = a^v_b B^{soc}_{bk,t-1} I^-_{bkt} + (b^v_b - i^{avg}_b r^{int}_b) I^-_{bkt} \quad \forall b \in \mathcal{B}, k \in \tilde{\mathcal{B}}_b, t \in \mathcal{T} \tag{A.5b}$$

$$B^{soc}_{bkt} = B^{soc}_{bk,t-1} + \frac{\tau(\eta^+_b I^+_{bkt} - I^-_{bkt})}{c^{ref}_b}, \quad \forall b \in \mathcal{B}, k \in \tilde{\mathcal{B}}_b, t \in \mathcal{T} \tag{A.5c}$$

$$\underline{s}_b W^\ell_{bk} \leq B^{soc}_{bkt} \leq \bar{s}_b W^\ell_{bk}, \quad \forall b \in \mathcal{B}, k \in \tilde{\mathcal{B}}_b, t \in \mathcal{T}_\ell, \ \ell \in \mathcal{L} \tag{A.5d}$$

$$B^{soc}_{bkt} \leq B^{soc}_{b,k-1,t} + (1 - W^\ell_{bk}), \quad \forall b \in \mathcal{B}, k \in \tilde{\mathcal{B}}_b, t \in \mathcal{T}_\ell, \ell \in \mathcal{L} : k > 1 \tag{A.5e}$$

$$B^{soc}_{bkt} \geq B^{soc}_{b,k-1,t} - (1 - W^\ell_{bk}), \quad \forall b \in \mathcal{B}, k \in \tilde{\mathcal{B}}_b, t \in \mathcal{T}_\ell, \ell \in \mathcal{L} : k > 1 \tag{A.5f}$$

$$\sum_{b \in \mathcal{B}} \sum_{k \in \mathcal{B}_b} c^{ref}_b B^{soc}_{bk,(\ell-1)\upsilon} = R^\ell, \ \forall \ell \in \mathcal{L} \setminus \{1\} \tag{A.5g}$$

$$\sum_{b \in \mathcal{B}} \sum_{k \in \mathcal{B}_b} c^{ref}_b B^{soc}_{bk,\ell\upsilon} = R^\ell, \ \forall \ell \in \mathcal{L} \tag{A.5h}$$

$$\underline{p}_b B^+_{bkt} \leq P^+_{bkt} \leq \bar{p}_b B^+_{bkt}, \ \forall b \in \mathcal{B}, k \in \tilde{\mathcal{B}}_b, t \in \mathcal{T} \tag{A.5i}$$

$$\underline{p}_b B^-_{bkt} \leq P^-_{bkt} \leq \bar{p}_b B^-_{bkt}, \ \forall b \in \mathcal{B}, k \in \tilde{\mathcal{B}}_b, t \in \mathcal{T} \tag{A.5j}$$

$$i^{L+}_b B^+_{bkt} \leq I^+_{bkt} \leq i^{U+}_b B^+_{bkt}, \quad \forall b \in \mathcal{B}, k \in \tilde{\mathcal{B}}_b, t \in \mathcal{T} \tag{A.5k}$$

$$i^{L-}_b B^-_{bkt} \leq I^-_{bkt} \leq i^{U-}_b B^-_{bkt}, \quad \forall b \in \mathcal{B}, k \in \tilde{\mathcal{B}}_b, t \in \mathcal{T} \tag{A.5l}$$

$$I^-_{bkt} \leq i^{U-}_b B^{soc}_{bk,t-1}, \quad \forall b \in \mathcal{B}, k \in \tilde{\mathcal{B}}_b, t \in \mathcal{T} \tag{A.5m}$$

$$B^+_{bkt} + B^-_{bkt} \leq W^\ell_{bk}, \quad \forall b \in \mathcal{B}, k \in \tilde{\mathcal{B}}_b, t \in \mathcal{T}_\ell, \ell \in \mathcal{L} \tag{A.5n}$$

$$B^+_{bkt} + B^-_{b'k't} \leq 1, \quad \forall b, b' \in \mathcal{B}; k, k' \in \tilde{\mathcal{B}}_b; t \in \mathcal{T} : b \neq b', k \neq k' \tag{A.5o}$$

Nonanticipativity:

$$W^\ell_{jk} = W_{jk}, \ \forall j \in \mathcal{J}, k \in \tilde{\mathcal{J}}_j, \ell \in \mathcal{L} \tag{A.6a}$$

$$X^\ell_s = X_s, \ \forall s \in \mathcal{S}, \ell \in \mathcal{L} \tag{A.6b}$$

$$R^\ell = R, \ \forall \ell \in \mathcal{L} \setminus \{1\} \tag{A.6c}$$

Boundary Condition:

$$B^{soc}_{bk,0} = b^0_b W_{bk}, \quad \forall b \in B, k \in \tilde{\mathcal{B}}_b \tag{A.7}$$

Nonnegativity and Integrality:

$$P^+_{jkt}, P^-_{jkt} \geq 0, \quad \forall j \in \mathcal{J}, k \in \tilde{\mathcal{J}}_j, t \in \mathcal{T} \tag{A.8a}$$

$$P^{PV}_{st} \geq 0, \quad \forall s \in \mathcal{S}, t \in \mathcal{T} \tag{A.8b}$$

$$\tilde{F}_t \geq 0, \quad t \in \mathcal{T} \tag{A.8c}$$

$$B^{soc}_{bkt}, I^+_{bkt}, I^-_{bkt} \geq 0, \quad \forall b \in \mathcal{B}, k \in \tilde{\mathcal{B}}_b, t \in \mathcal{T} \tag{A.8d}$$

$$R \ \text{ unrestricted} \tag{A.8e}$$

$$R^\ell \ \text{ unrestricted}, \quad \forall \ell \in \mathcal{L} \tag{A.8f}$$

$$B^{soc}_{bk,0} \geq 0, \quad \forall b \in \mathcal{B}, k \in \tilde{\mathcal{B}}_b \tag{A.8g}$$

$$W_{jk} \in \{0, 1\}, \quad \forall j \in \mathcal{J}, k \in \tilde{\mathcal{J}}_j \tag{A.8h}$$

$$X_s \in \mathbb{Z}^+, \quad \forall s \in \mathcal{S} \tag{A.8i}$$

$$W_{jk}^{\ell} \in \{0,1\}, \quad \forall j \in \mathcal{J}, k \in \tilde{\mathcal{J}}_j, \ell \in \mathcal{L} \tag{A.8j}$$

$$X_s^{\ell} \in \mathbb{Z}^+, \quad \forall s \in \mathcal{S}, \ell \in \mathcal{L} \tag{A.8k}$$

$$G_{gkt} \in \{0,1\}, \quad \forall g \in \mathcal{G}, k \in \tilde{\mathcal{G}}_g, t \in \mathcal{T} \tag{A.8l}$$

$$B_{bkt}^{+}, B_{bkt}^{-} \in \{0,1\}, \quad \forall b \in \mathcal{B}, k \in \tilde{\mathcal{B}}_b, t \in \mathcal{T} \tag{A.8m}$$

Discussion

A.1.1.1 Objective Function

We minimize the objective function in (A.1), which includes the cost of (i) procuring diesel generators, batteries and PV systems, (ii) consuming fuel, and (iii) using the technologies in dispatch, measured in lifecycles; a generator uses one lifecycle for each hour it is running, while a battery uses lifecycles according to a function of the charge current and state-of-charge of the battery in each time period. Our implementation is analogous to assigning values to variable L_{jk} via constraints (6a) and (14) from Scioletti et al. (2017); however, we do not impose a restriction on the number of lifecycles spent per constraint (6c) from their paper. The parameter ν reconciles the time horizon of the optimization model with that of the FOB's duration.

A.1.1.2 System Operations

Constraint (A.2a) requires that the dispatch strategy meets demand for each hour. Constraint (A.2b) enforces a spinning reserve requirement that is equal to a fraction of PV power output in each time period. Alternative approaches model spinning reserve requirements as a function of the demand not met by PV systems; see, e.g., Husted et al. (2018). Constraint (A.2c) breaks symmetry (Sherali and Smith

2001).

A.1.1.3 Generator Operations

Constraint (A.3a) restricts power output of generators to their specific minimum and maximum power ratings. Constraint (A.3b) determines fuel consumption as a function of whether each generator in the microgrid design is running, and, if so, its power output. While Scioletti et al. (2017) include a second-order term in the relationship, i.e.,

$$\tilde{F}_t \geq \tau \sum_{g \in \mathcal{G}} \sum_{k \in \tilde{\mathcal{G}}_g} (a_g^f (P_{gkt}^-)^2 + b_g^f P_{gkt}^- + c_g^f G_{gkt}), \quad \forall t \in \mathcal{T},$$

their implementation sets the value of the parameter a_g^f to zero. Constraint (A.3c) connects procurement to dispatch, and constraints (A.3d) and (A.3e) break symmetry.

A.1.1.4 PV Operations

Constraint (A.4a) enforces an upper bound on PV output power per panel according to the solar irradiance in each hour. Constraint (A.4b) limits the number of systems in the design according to spatial restrictions.

A.1.1.5 Battery Operations

Constraints (A.5a) and (A.5b) restrict the net power output of the battery to the product of the battery's voltage and its net current for each time period; this is consistent with the battery modeling paradigm developed by Scioletti et al. (2016a). Our implementation does not use a decision variable for voltage, but rather, uses the right-hand side of constraint (5g) from Scioletti et al. (2017) to model battery

voltage. While the implementation from Scioletti et al. includes the products of terms $B^{+}_{bk,t-1} \cdot I^{+}_{bkt}$ and $B^{-}_{bk,t-1} \cdot I^{-}_{bkt}$ in the right-hand sides of our constraints (A.5a) and (A.5b), respectively. The bounds provided in constraints (A.5k)-(A.5l) require that $I^{+}_{bkt} = 0$ when the battery is not charging, and that $I^{-}_{bkt} = 0$ when the battery is not discharging, i.e., $B^{+}_{bkt} \cdot I^{+}_{bkt} = I^{+}_{bkt}$ and $B^{-}_{bkt} \cdot I^{-}_{bkt} = I^{-}_{bkt}$. As a result, our constraints (A.5a)-(A.11) are an equivalent formulation to equations (5a), (5b), and (5g) from the predecessor paper.

Constraint (A.5c) updates the battery state-of-charge according to charge and discharge currents, the former of which is modified by an efficiency term. Constraint (A.5d) enforces upper and lower bounds on the battery's state-of-charge. Constraints (A.5e) and (A.5f) force all batteries in the design to maintain the same state-of-charge. Constraints (A.5g)-(A.5h) enforce the reset policy described in Section 2.2.1. Constraints (A.5i) and (A.5j) provide upper and lower bounds for net power, while constraints (A.5k) through (A.5m) provide similar bounds for net current. Constraints (A.5n) and (A.5o) preclude simultaneous charging and discharging of batteries.

A.1.1.6 Nonanticipativity and Boundary Condition

Constraints (A.6) require a single design and reset inventory level. In the lower bound model, ($\underline{\mathcal{P}}$), we implement a Lagrangian relaxation of these constraints with multipliers that we update via Algorithm 1. In the upper bound model, ($\bar{\mathcal{P}}$), we provide (fixed) values for: $W_{bk}, \forall b \in \mathcal{B},\ k \in \tilde{\mathcal{B}}_b$; $X_s, \forall s \in \mathcal{S}$; and, R. Constraint (A.7) enforces a specific initial state-of-charge for each battery in the design.

A.1.1.7 Nonnegativity and Integer Restrictions

Constraints (A.8) provide nonnegativity and integer restrictions for the design and dispatch decision variables. Our implementation relaxes constraint (A.8i) to allow fractional values for X_s^ℓ, $\forall s \in \mathcal{S},\ \ell \in \mathcal{L}$.

A.1.2 Mapping Microgrid Design and Dispatch Problem to Model ($\mathcal{P}$)

We map the technology purchase decision variables $W_{jk}^\ell,\ j \in \mathcal{J}, k \in \tilde{\mathcal{J}}_j$, and $X_s^\ell,\ s \in \mathcal{S}$, to the decision vector X^ℓ in our formulation and implement R^ℓ directly, for all subproblems $\ell \in \mathcal{L}$. We map $B_{bk,t-1}^{soc}$ and $B_{bkt}^{soc},\ b \in \mathcal{B}, k \in \tilde{\mathcal{B}}_b$, to $\underline{Y}_t$ and $\bar{Y}_t$, respectively, $t \in \mathcal{T}$. We map all other decision variables to $Y_t,\ t \in \mathcal{T}$. Constraints (A.2c), (A.4b), and (A.8h)-(A.8k) correspond to constraint (2.9b). Constraint (A.5c) corresponds to constraints (2.9d)-(2.9e), which are condensed by substituting the variable for each battery's starting state-of-charge in period t with that of its ending state-of-charge at period $t-1$. Constraints (A.5g)-(A.5h) correspond to constraints (2.9f)-(2.9g). Nonanticipativity constraints (A.6) correspond to constraints (2.9h) and (2.9i), and the boundary condition (A.7) corresponds to constraint (2.9j). All other constraints constitute (2.9c) in model ($\mathcal{P}$). We model the inventory variables $\bar{Y}_t$ as total battery capacity (in Ah) via:

$$\bar{Y}_t = \sum_{b \in \mathcal{B}} \sum_{k \in \tilde{\mathcal{B}}_b} c_b^{ref} B_{bkt}^{soc},\ \forall t \in \mathcal{T}, \tag{A.9}$$

and calculate $\underline{Y}_t$ and R similarly. This allows us to use a single value for R when we perform the bisection search in Algorithm 1.

A.1.3 Mapping Linearization to Model ($\mathcal{U}$)

Because voltage is a function of battery state-of-charge, we substitute the products $B^{soc}_{bk,t-1} \cdot I^{+}_{bkt}$ and $B^{soc}_{bk,t-1} \cdot I^{-}_{bkt}$ with the linearization variables Z^{+}_{bkt} and Z^{-}_{bkt}, respectively, and constraints (A.10) and (A.11) enforce the bilinear relationships in the nonlinear model ($\mathcal{P}$):

$$Z^{+}_{bkt} = B^{soc}_{bk,t-1} I^{+}_{bkt}, \quad \forall b \in \mathcal{B}, k \in \tilde{\mathcal{B}}_b, t \in \mathcal{T} \tag{A.10}$$

$$Z^{-}_{bkt} = B^{soc}_{bk,t-1} I^{-}_{bkt}, \quad \forall b \in \mathcal{B}, k \in \tilde{\mathcal{B}}_b, t \in \mathcal{T}. \tag{A.11}$$

We map the state-of-charge variables, $B^{soc}_{bk,t-1}$, and current variables, I^{+}_{bkt}, to Y_{1t} and Y_{2t}, respectively, for each battery, twin and time period; we perform an analogous mapping for B^{soc}_{bkt} and I^{-}_{bkt}. Rather than just having simple lower bounds, variable I^{+}_{bkt} has a lower bound of i^{L+}_{b} if binary variable B^{+}_{bkt} is one and otherwise the lower bound is zero, with an analogous bound of i^{L-}_{b} for I^{-}_{bkt} if $B^{-}_{bkt} = 1$ and zero otherwise; see constraints (A.5k)-(A.5l). Likewise, variable B^{soc}_{bkt} has a lower bound of $\underline{s}_b$ if $W^{\ell}_{bk} = 1$, and zero otherwise. The following variant of constraints (2.8) incorporates these restrictions using binary variables B^{+}_{bkt}, B^{-}_{bkt}, and W^{ℓ}_{bk}:

Sets

$n \in \mathcal{N}$	set of all subregions in partitioning of support of current

New Battery Parameters

i^{L+}_{bn}	minimum charge current to battery b in subregion n (A)
i^{U+}_{bn}	maximum charge current to battery b in subregion n (A)
i^{L-}_{bn}	minimum discharge current from battery b in subregion n (A)
i^{U-}_{bn}	maximum discharge current from battery b in subregion n (A)

New Battery Variables

Z^+_{bkt}, Z^-_{bkt}	battery b, twin k auxiliary variable for product, and charge and discharge current, respectively, and starting state-of-charge in period t [A]
λ^+_{bknt}	1 if charge current for battery b, twin k is in subregion n at period t, 0 otherwise
λ^-_{bknt}	1 if discharge current for battery b, twin k is in subregion n at period t, 0 otherwise

Constraints

$$\sum_{n\in\mathcal{N}} \lambda^+_{bknt} = B^+_{bkt}, \quad \forall b \in \mathcal{B}, k \in \tilde{\mathcal{B}}_b, t \in \mathcal{T} \tag{A.12a}$$

$$\sum_{n\in\mathcal{N}} \lambda^-_{bknt} = B^-_{bkt}, \quad \forall b \in \mathcal{B}, k \in \tilde{\mathcal{B}}_b, t \in \mathcal{T} \tag{A.12b}$$

$$I^+_{bkt} \geq i^{L+}_b B^+_{bkt} + (i^{L+}_{bn} - i^{L+}_b)\lambda^+_{bknt}, \quad \forall b \in \mathcal{B}, k \in \tilde{\mathcal{B}}_b, n \in \mathcal{N}, t \in \mathcal{T} \tag{A.12c}$$

$$I^+_{bkt} \leq i^{U+}_b B^+_{bkt} - (i^{U+}_b - i^{U+}_{bn})\lambda^+_{bknt}, \quad \forall b \in \mathcal{B}, k \in \tilde{\mathcal{B}}_b, n \in \mathcal{N}, t \in \mathcal{T} \tag{A.12d}$$

$$I^-_{bkt} \geq i^{L-}_b B^-_{bkt} + (i^{L-}_{bn} - i^{L-}_b)\lambda^-_{bknt}, \quad \forall b \in \mathcal{B}, k \in \tilde{\mathcal{B}}_b, n \in \mathcal{N}, t \in \mathcal{T} \tag{A.12e}$$

$$I^-_{bkt} \leq i^{U-}_b B^-_{bkt} - (i^{U-}_b - i^{U-}_{bn})\lambda^-_{bknt}, \quad \forall b \in \mathcal{B}, k \in \tilde{\mathcal{B}}_b, n \in \mathcal{N}, t \in \mathcal{T} \tag{A.12f}$$

$$Z^+_{bkt} \geq i^{U+}_{bn} B^{soc}_{bk,t-1} + \bar{s}_b I^+_{bkt} - \bar{s}_b i^{U+}_{bn} W^\ell_{bk} - (\bar{s}_b - \underline{s}_b)(i^{U+}_b - i^{U+}_{bn})(W^\ell_{bk} - \lambda^+_{bknt}),$$
$$\forall b \in \mathcal{B}, k \in \tilde{\mathcal{B}}_b, n \in \mathcal{N}, t \in \mathcal{T}_\ell, \ell \in \mathcal{L} \tag{A.12g}$$

$$Z^+_{bkt} \geq i^{L+}_{bn} B^{soc}_{bk,t-1} + \underline{s}_b I^+_{bkt} - \underline{s}_b i^{L+}_{bn} W^\ell_{bk} - (\bar{s}_b - \underline{s}_b)(i^{L+}_{bn} - i^{L+}_b)(W^\ell_{bk} - \lambda^+_{bknt}),$$
$$\forall b \in \mathcal{B}, k \in \tilde{\mathcal{B}}_b, n \in \mathcal{N}, t \in \mathcal{T}_\ell, \ell \in \mathcal{L} \tag{A.12h}$$

$$Z^+_{bkt} \leq i^{L+}_{bn} B^{soc}_{bk,t-1} + \bar{s}_b I^+_{bkt} - \bar{s}_b i^{L+}_{bn} W^\ell_{bk} + (\bar{s}_b - \underline{s}_b)(i^{L+}_{bn} - i^{L+}_b)(W^\ell_{bk} - \lambda^+_{bknt}),$$
$$\forall b \in \mathcal{B}, k \in \tilde{\mathcal{B}}_b, n \in \mathcal{N}, t \in \mathcal{T}_\ell, \ell \in \mathcal{L} \tag{A.12i}$$

$$Z^+_{bkt} \leq i^{U+}_{bn} B^{soc}_{bk,t-1} + \underline{s}_b I^+_{bkt} - \underline{s}_b i^{U+}_{bn} W^\ell_{bk} + (\bar{s}_b - \underline{s}_b)(i^{U+}_b - i^{U+}_{bn})(W^\ell_{bk} - \lambda^+_{bknt}),$$

$$\forall b \in \mathcal{B}, k \in \tilde{\mathcal{B}}_b, n \in \mathcal{N}, t \in \mathcal{T}_\ell, \ell \in \mathcal{L} \tag{A.12j}$$

$$Z^-_{bkt} \geq i^{U-}_{bn} B^{soc}_{bk,t-1} + \bar{s}_b I^-_{bkt} - \bar{s}_b i^{U-}_{bn} W^\ell_{bk} - (\bar{s}_b - \underline{s}_b)(i^{U-}_b - i^{U-}_{bn})(W^\ell_{bk} - \lambda^-_{bknt}),$$

$$\forall b \in \mathcal{B}, k \in \tilde{\mathcal{B}}_b, n \in \mathcal{N}, t \in \mathcal{T}_\ell, \ell \in \mathcal{L} \tag{A.12k}$$

$$Z^-_{bkt} \geq i^{L-}_{bn} B^{soc}_{bk,t-1} + \underline{s}_b I^-_{bkt} - \underline{s}_b i^{L-}_{bn} W^\ell_{bk} - (\bar{s}_b - \underline{s}_b)(i^{L-}_{bn} - i^{L-}_b)(W^\ell_{bk} - \lambda^-_{bknt}),$$

$$\forall b \in \mathcal{B}, k \in \tilde{\mathcal{B}}_b, n \in \mathcal{N}, t \in \mathcal{T}_\ell, \ell \in \mathcal{L} \tag{A.12l}$$

$$Z^-_{bkt} \leq i^{L-}_{bn} B^{soc}_{bk,t-1} + \bar{s}_b I^-_{bkt} - \bar{s}_b i^{L-}_{bn} W^\ell_{bk} + (\bar{s}_b - \underline{s}_b)(i^{L-}_{bn} - i^{L-}_b)(W^\ell_{bk} - \lambda^-_{bknt}),$$

$$\forall b \in \mathcal{B}, k \in \tilde{\mathcal{B}}_b, n \in \mathcal{N}, t \in \mathcal{T}_\ell, \ell \in \mathcal{L} \tag{A.12m}$$

$$Z^-_{bkt} \leq i^{U-}_{bn} B^{soc}_{bk,t-1} + \underline{s}_b I^-_{bkt} - \underline{s}_b i^{U-}_{bn} W^\ell_{bk} + (\bar{s}_b - \underline{s}_b)(i^{U-}_b - i^{U-}_{bn})(W^\ell_{bk} - \lambda^-_{bknt}),$$

$$\forall b \in \mathcal{B}, k \in \tilde{\mathcal{B}}_b, n \in \mathcal{N}, t \in \mathcal{T}_\ell, \ell \in \mathcal{L} \tag{A.12n}$$

$$Z^+_{bkt} \leq \bar{s}_b I^+_{bkt}, \quad \forall b \in \mathcal{B}, k \in \tilde{\mathcal{B}}_b, t \in \mathcal{T} \tag{A.12o}$$

$$Z^-_{bkt} \leq \bar{s}_b I^-_{bkt}, \quad \forall b \in \mathcal{B}, k \in \tilde{\mathcal{B}}_b, t \in \mathcal{T} \tag{A.12p}$$

$$\lambda^-_{bknt}, \lambda^+_{bknt} \in \{0,1\}, \quad \forall b \in \mathcal{B}, k \in \tilde{\mathcal{B}}_b, n \in \mathcal{N}, t \in \mathcal{T} \tag{A.12q}$$

$$Z^+_{bkt}, Z^-_{bkt} \geq 0, \quad \forall b \in \mathcal{B}, k \in \tilde{\mathcal{B}}_b, t \in \mathcal{T}. \tag{A.12r}$$

Constraints (A.12a)-(A.12n) replicate the linearization constraints (2.8) applied to the microgrid design and dispatch problem for each battery technology and twin. Constraints (A.12o)-(A.12p) require that $Z^+_{bkt} = 0$ when battery b, twin k is not charging in period t, and that $Z^-_{bkt} = 0$ when the battery is not discharging.

Appendix B

Remote Microgrid Design Optimization Under Photovoltaic And Load Uncertainty

B.1 Special Events in FOB Occupancy Model

This section describes the special events listed in Table 3.2 that impact the FOB population beyond the schedule described in Section 3.2.2.1.

B.1.1 Holidays

We use the fixed dates of February 4th (Super Bowl), November 25th (Thanksgiving), and December 25th (Christmas) as holidays for which there is reduced activity outside of the base. For each of these dates, we assume one of the following takes place with equal probability: (i) regular operations; (ii) regular operations with 15 fewer soldiers off-base; (iii) regular operations with 30 fewer soldiers off-base; and, (iv) nobody leaves the base for missions.

B.1.2 Rebuilding

In the event of an attack on the base, we assume that a group of 20-35 engineers visit the base for 5-9 days to repair the perimeter and any damaged buildings. The team size and duration of stay assume any of the values in their respective ranges with equal probability. These events take place only during high op-tempo; zero or

one attacks occur with probability 0.45 each, while two attacks occur with probability 0.1.

B.1.3 Resupply

Resupply convoys arrive at the FOB throughout the year. Several factors may influence the frequency of resupply convoys, so we assume that the number of resupply convoys varies between two and five in each month, with two and three occurring with probability 0.35 each, and four and five resupplies occurring with probabilities of 0.20 and 0.10, respectively. If there are very few resupplies within a given month, then we assume that the convoy must be bigger. Therefore, we let the number of soldiers in the convoy, which we denote as N_s, vary randomly and as a function of the number of resupplies, which we denote as N_r, as follows:

$$N_s = 134 - 17 \cdot N_r + \varepsilon,$$

in which $\varepsilon \sim N(0, 10)$, rounded to the nearest integer. We also assume that the convoy can arrive at anytime of day and the duration of its stay, which we denote L, is between 5 and 48 hours according to the following distribution:

$$\mathbb{P}(L = i) = \frac{i^{-p}}{\sum_{j=l}^{u} j^{-p}}, \tag{B.1}$$

in which $l = 5$, $u = 48$, and $p = 3$, the last of which provides greater weight to shorter durations.

B.1.4 Turnover Events

We assume that one troop turnover event occurs in a given year with probability 0.75, and two events occur in a given year with probability 0.25. If two turnovers occur, we assume the first happens in the first half of the year, and the second occurs at least six months after the first. We assume the additional number of soldiers varies between 100 to 156, with any outcome in that range being equally likely. The duration of overlap is assumed to last between 36 and 96 hours, with outcomes distributed according to equation (B.1), with $l = 36$, $u = 96$, and $p = 4$.

B.1.5 Task Force Missions

We assume that staging missions occur approximately once every two months during high op-tempo. Specifically, if the high op-tempo period is all 12 months, 9 months, or 6 months, then the number of staging missions is 5 to 8, 4 to 6, and 3 to 5, respectively. The number of additional soldiers follows a Triangle(60,80,100) distribution, rounded to the nearest number. The length of time that the soldiers are staged at the FOB is between 6 and 24, following the distribution given in equation (B.1) with $l = 6$, $u = 24$, and $p = 3$.

B.1.6 Training

Training sessions occur when soldiers obtain a new piece of equipment or software that requires training to operate. These sessions can be held anytime during the year, and they last anywhere between 3 and 7 days, with 5 days most likely with a probability of 0.4, 4 and 6 days occur with probability 0.2, and 3 and 7 days occur

with probability 0.1. We assume that either 2 or 3 training sessions occur during the year with equal probability. Any number between 15 and 35 additional soldiers is equally likely to visit from outside the base for training.

B.1.7 Fighting Missions

A fighting mission is a joint-base effort in which a large proportion of the base population departs simultaneously. We assume that these occur approximately once per month during the high op-tempo period. During these events, the number of soldiers that leave the base is governed by a triangle(90,135,180) distribution, and the length of time that the soldiers are off-base follows the distribution given in equation (B.1), with $l = 20$, $u = 96$, and $p = 3$.

B.1.8 Miscellaneous Single Movers

We assume that anywhere between 20 and 40 soldiers depart the base individually for various reasons and lengths of time throughout the year. The reasons are for (i) a two-week leave; (ii) a part retrieval, which may take anywhere between 12 and 24 hours; (iii) an illness, which may take anywhere between one and three days with equal probability; or (iv) an injury, which may take between five and ten days with equal probability. A random sample between 20 and 40 soldiers is first taken, and then a reason for their departure is sampled with a 50% chance for leave; a 25% chance for part retrieval; a 10% chance for illness; and, a 15% chance for injury.

Bibliography

A123 (2018). A123 battery manufacturer. `http://www.a123systems.com/`. accessed 2018-05-15.

Al-Khayyal, F. A. and J. E. Falk (1983). Jointly constrained biconvex programming. *Mathematics of Operations Research 8*(2), 273–286.

Androulakis, I. P., C. D. Maranas, and C. A. Floudas (1995). αBB: A global optimization method for general constrained nonconvex problems. *Journal of Global Optimization 7*(4), 337–363.

Antonanzas, J., N. Osorio, R. Escobar, R. Urraca, F. M. de Pison, and F. Antonanzas-Torres (2016). Review of photovoltaic power forecasting. *Solar Energy 136*, 78–111.

ASHRAE (2002). International weather for energy calculations (IWEC weather files) user's manual. Technical report, ASHRAE, Atlanta, Georgia.

Avramidis, A. N., K. W. Bauer, and J. R. Wilson (1991). Simulation of stochastic activity networks using path control variates. *Naval Research Logistics 38*, 183–201.

Baker, S. F., D. P. Morton, R. E. Rosenthal, and L. M. Williams (2002). Optimizing military airlift. *Operations Research 50*(4), 582–602.

Bala, B. and S. Siddiqui (2009). Optimal design of a PV-diesel hybrid system for electrification of an isolated island in Sandwip, Bangladesh using a genetic algorithm. *Energy for Sustainable Development 13*(3), 137–142.

Barbier, T., M. Anjos, and G. Savard (2014). Optimization of diesel, wind, and battery hybrid power systems. Technical Report G-2014-02, Group for Research and Decision Analysis and Department of Mathematics and Industrial Engineering Polytechnique Montréal.

Barley, C. D. and C. B. Winn (1996). Optimal dispatch strategy in remote hybrid power systems. *Solar Energy 58*(4-6), 165–179.

Bayraksan, G. (2018). An improved averaged two-replication procedure with Latin hypercube sampling. *Operations Research Letters 46*, 173–178.

Belotti, P. (2009). Couenne: A user's manual. Technical report, Lehigh University.

Bergamini, M. L., P. Aguirre, and I. E. Grossman (2005). Logic-based outer approximation for globally optimal synthesis of process networks. *Computers and Chemical Engineering 72*(9), 1914–1933.

Blair, N., A. Dobos, J. Freeman, T. Neises, and M. Wagner (2014). System Advisor Model,

SAM 2014.1.14: General description. Technical report, National Renewable Energy Laboratory.

Bonami, P. and J. Lee (2009, November). BONMIN Version 1.4 User's Manual.

Brown, G. G., R. F. Dell, and A. M. Newman (2004). Optimizing military capital planning. *Interfaces 34*(6), 415–425.

Castro, P. (2015). Tightening piecewise McCormick relaxations for bilinear problems. *Computers and Chemical Engineering 72*, 300–311.

Department of Defense (2002, August). *Guide to Container Inspection for Commercial and Military Intermodal Containers*. `http://everyspec.com/MIL-HDBK/MIL-HDBK-0099-0199/MIL-HDBK-138B_4075/`, accessed 2018-07-15.

Dey, S. S. and A. Gupte (2015). Analysis of MILP techniques for the pooling problem. *Operations Research 63*(2), 412–427.

Dobos, A. P. (2013). PVWatts Version 1 Technical Reference. Technical report, National Renewable Energy Laboratory.

Dolan, E. D. and J. J. Moré (2002). Benchmarking optimization software with performance profiles. *Mathematical Programming 92*(1), 201–213.

Dolara, A., S. Leva, and G. Manzolini (2015). Comparison of different physical models for PV power output prediction. *Solar Energy 119*, 83–99.

Drew, S. and T. Homem-de-Mello (2012). Some large deviations results for Latin hypercube sampling. *Methodology and Computing in Applied Probability 14*(2), 203–232.

Dufo-López, R. and J. L. Bernal-Agustín (2005). Design and control strategies of PV-Diesel systems using genetic algorithms. *Solar Energy 79*(1), 33–46.

Dvorkin, Y., R. Fernndez-Blanco, D. S. Kirschen, H. Pandi, J. P. Watson, and C. A. Silva-Monroy (2017, Jan). Ensuring profitability of energy storage. *IEEE Transactions on Power Systems 32*(1), 611–623.

Erwin, S. I. (2010, April). How much does the Pentagon pay for a gallon of gas? `http://www.nationaldefensemagazine.org/archive/2010/April/Pages/HowMuchforaGallonofGas.aspx`, accessed 2016-03-01.

Escudero, L. F., M. A. Garín, and A. Unzueta (2016). Cluster Lagrangean decomposition in multistage stochastic optimization. *Computers and Operations Research 67*(2016), 48–62.

Freimer, M. B., J. Linderoth, and D. J. Thomas (2012). The impact of sampling methods on bias and variance in stochastic linear programs. *Computational Optimization and Applications 51*(1), 51–75.

Gade, D., G. Hackebeil, S. M. Ryan, J.-P. Watson, R. J.-B. Wets, and D. L. Woodruff (2014). Obtaining lower bounds from the progressive hedging algorithm for stochastic

mixed-integer programs. Technical report, Graduate School of Management, University of California-Davis, Davis, CA (US).

Gelaro, R., W. McCarty, M. J. Surez, R. Todling, A. Molod, L. Takacs, C. A. Randles, A. Darmenov, M. G. Bosilovich, R. Reichle, K. Wargan, L. Coy, R. Cullather, C. Draper, S. Akella, V. Buchard, A. Conaty, A. M. da Silva, W. Gu, G.-K. Kim, R. Koster, R. Lucchesi, D. Merkova, J. E. Nielsen, G. Partyka, S. Pawson, W. Putman, M. Rienecker, S. D. Schubert, M. Sienkiewicz, and B. Zhao (2017). The modern-era retrospective analysis for research and applications, version 2 (MERRA-2). *Journal of Climate 30*(14), 5419–5454.

Gildea, G. S., P. D. Carpenter, B. J. Campbell, J. Quigley, J. Diaz, J. Langley, N. Putnam, B. Hargreaves, K. Donahue, W. Lindo, W. F. Harris, and J. A. Miletti (2017, September). SLB-STO-D analysis report: Modeling and simulation analysis of fuel, water, and waste reductions in base camps: 50, 300, and 1000 persons. Technical Report AD1039173, Army Natick Soldier Research Development and Engineering Center, Natick, MA, United States.

Gounaris, C. E., R. Misener, and C. A. Floudas (2009). Computational comparison of piecewise-linear relaxations for pooling problems. *Industrial & Engineering Chemistry Research 48*(12), 5742–5766.

Green, H. J. and J. F. Manwell (1995). Hybrid2: A versatile model of the performance of hybrid power systems. Technical report.

Gupta, A., R. Saini, and M. Sharma (2011). Modeling of hybrid energy system Part II: Combined dispatch strategies and solution algorithm. *Renewable Energy 36*(2), 466–473.

Gupte, A., S. Ahmed, M. S. Cheon, and S. Dey (2013). Solving mixed integer bilinear problems using MILP formulations. *SIAM Journal on Optimization 23*(2), 721–744.

Hasan, M. and I. Karimi (2010). Piecewise linear relaxation of bilinear programs using bivariate partitioning. *AIChE Journal 56*(7), 1880–1893.

HDT Global (2016a, August). *F100-60K Commercial ECU*. Rev. 2.

HDT Global (2016b, November). *HDT AirBeam Temper Shelters*. Rev. 1.

Helton, J. C. and F. J. Davis (2003). Latin hypercube sampling and the propagation of uncertainty in analyses of complex systems. *Reliability Engineering and System Safety 81*, 23–69.

Homem-de-Mello, T. (2008). On rates of convergence for stochastic optimization problems under non-independent and identically distributed sampling. *SIAM Journal on Optimization 19*(2), 524–551.

HOMER (2015). Micropower system modeling with HOMER. `http://homerenergy.com/documents/MicropowerSystemModelingWithHOMER.pdf` accessed 2015-08-08.

Huneke, F., J. Henkel, J. A. B. González, and G. Erdmann (2012). Optimisation of hybrid off-grid energy systems by linear programming. *Energy, Sustainability and Society 2*(1), 1–19.

Husslage, B. G. M., G. Rennen, E. R. van Dam, and D. den Hertog (2011, Dec). Space-filling Latin hypercube designs for computer experiments. *Optimization and Engineering 12*(4), 611–630.

Husted, M. A., B. Suthar, G. H. Goodall, A. M. Newman, and P. A. Kohl (2018). Coordinating procurement decisions with a dispatch strategy featuring a concentration gradient. *Applied Energy 219*, 394–407.

IBM (2017). CPLEX 12.6.2 User Documentation.

Iman, R. L. and W. J. Conover (1980). Small sample sensitivity analysis techniques for computer models, with an application to risk assessment. *Communications in Statistics - Theory and Methods 9*(17), 1749–1842.

Karuppiah, R. and I. E. Grossman (2006). Global optimization for the synthesis of integrated water systems in chemical processes. *Computers and Chemical Engineering 30*(4), 650–673.

Katsigiannis, Y. A. and P. S. Georgilakis (2008). Optimal sizing of small isolated hybrid power systems using tabu search. *Journal of Optoelectronics and Advanced Materials 10*(5), 1241–1245.

Khodaei, A., S. Bahramirad, and M. Shahidehpour (2015). Microgrid planning under uncertainty. *IEEE Transactions on Power Systems 30*(5), 2417–2425.

L'Ecuyer, P., L. Meliani, and J. Vaucher (2002). SSJ: A framework for stochastic simulation in Java. In E. Yücesan, C.-H. Chen, J. L. Snowdon, and J. M. Charnes (Eds.), *Proceedings of the 2002 Winter Simulation Conference*, pp. 234–242. IEEE Press.

Mak, W., D. P. Morton, and R. K. Wood (1999). Monte Carlo bounding techniques for determining solution quality in stochastic programs. *Operations Research Letters 24*(1), 47–56.

McCormick, G. P. (1976). Computability of global solutions to factorable nonconvex programs: Part I: Convex underestimating problems. *Mathematical Programming 10*(1), 147–175.

McKay, M. D., R. J. Beckman, and W. J. Conover (1979). A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics 21*(2), 239–245.

Mease, D. and D. Bingham (2006). Latin hyperrectangle sampling for computer experiments. *Technometrics 48*(4), 467–477.

Misener, R., J. Thompson, and C. A. Floudas (2011). APOGEE: Global optimization of standard, generalized, and extended pooling problems via linear and logarithmic partitioning schemes. *Computers and Chemical Engineering 35*, 876–892.

Morais, H., P. Kádár, P. Faria, Z. A. Vale, and H. M. Khodr (2010). Optimal scheduling of a renewable micro-grid in an isolated load area using mixed-integer linear programming. *Renewable Energy 35*(1), 151–156.

Morton, D., B. Letellier, J. Tejada, D. Johnson, Z. Mohaghegh, E. Kee, V. Moiseytseva, S. Reihani, and A. Zolan (2014). Sensitivity analyses of a simulation model for estimating fiber-induced sump screen and core failure rates. In *International Conference on Nuclear Engineering*, Volume 6. American Society of Mechanical Engineers.

Neyman, J. (1934). On the two different aspects of the representative method: the method of stratified sampling and the method of purposive selection. *Journal of the Royal Statistical Society 97*(4), 558–625.

Olsson, A., G. Sandberg, and O. Dahlblom (2003). On Latin hypercube sampling for structural reliability analysis. *Structural Safety 25*, 47–68.

Owen, A. (1992). A central limit theorem for Latin hypercube sampling. *Journal of the Royal Statistical Society. Series B (Methodological) 54*(2), 541–551.

Packham, N. and W. M. Schmidt (2010, Spring). Latin hypercube sampling with dependence and applications in finance. *The Journal of Computational Finance 13*(3), 81–111.

Panneton, F., P. L'Ecuyer, and M. Matsumoto (2006). Improved long-period generators based on linear recurrences modulo 2. *ACM Transactions on Mathematical Software 32*(1), 1–16.

Pruitt, K. A., R. J. Braun, and A. M. Newman (2013). Evaluating shortfalls in mixed-integer programming approaches for the optimal design and dispatch of distributed generation systems. *Applied Energy 102*(2013), 386–398.

Rienecker, M. M., M. J. Suarez, R. Gelaro, R. Todling, J. Bacmeister, E. Liu, M. G. Bosilovich, S. D. Schubert, L. Takacs, G.-K. Kim, S. Bloom, J. Chen, D. Collins, A. Conaty, A. Da Silva, W. Gu, J. Joiner, R. D. Koster, R. Lucchesi, A. Molod, T. Owens, S. Pawson, P. Pegion, C. R. Redder, R. Reichle, F. R. Robertson, A. G. Ruddick, M. Sienkiewicz, and J. Woollen (2011, Jul 15). MERRA: NASA's modern-era retrospective analysis for research and applications. *Journal of Climate 24*(14), 3624–3648. Copyright - Copyright American Meteorological Society Jul 15, 2011; Document feature - Tables; ; Graphs; Diagrams; Maps; Last updated - 2017-11-18.

Rockafellar, R. T. and R. J.-B. Wets (1991). Scenarios and policy aggregation in optimization under uncertainty. *Mathematics of Operations Research 16*(1), 119–147.

Rossum, G. (1995). Python reference manual. Technical report, Amsterdam, The Netherlands.

Sahinidis, N. V. (1996). BARON: A general purpose global optimization software package. *Journal of Global Optimization 8*, 201–205.

Schroedter-Homscheidt, M., A. Arola, N. Killius, M. Lefvre, L. Saboret, W. Wandji, L. Wald, and E. Wey (2016). The Copernicus atmosphere monitoring service (CAMS) radiation service in a nutshell. *22nd Proceedings of the SolarPACES conference*.

Scioletti, M. S. (2016a). *A mixed-integer program for the design and dispatch of a hybrid power generation system*. Ph. D. thesis, Colorado School of Mines.

Scioletti, M. S., J. K. Goodman, P. A. Kohl, and A. M. Newman (2016b). A physics-based integer linear battery modeling paradigm. *Applied Energy 176*, 245–257.

Scioletti, M. S., J. K. Goodman, A. M. Newman, A. J. Zolan, and S. Leyffer (2017). Optimal design and dispatch of a system of diesel generators, photovoltaics and batteries for remote locations. *Optimization and Engineering 18*(3), 755–792.

Sherali, H. D. and J. C. Smith (2001). Improving discrete model representations via symmetry considerations. *Management Science 47*(10), 1396–1407.

Stein, M. (1987). Large sample properties of simulations using Latin hypercube sampling. *Technometrics 29*(2), 143–151.

Stockbridge, R. and G. Bayraksan (2016). Variance reduction in Monte Carlo sampling-based optimality gap estimators for two-stage stochastic linear programming. *Computational Optimization and Applications 64*(2), 407–431.

Tang, B. (1993). Orthogonal array-based Latin hypercubes. *Journal of the American Statistical Association 88*(424), 1392–1397.

van Dam, E. R., B. Husslage, D. den Hertog, and H. Melissen (2007). Maximin Latin hypercube designs in two dimensions. *Operations Research 55*(1), 158–169.

van Dam, E. R., G. Rennen, and B. Husslage (2009). Bounds for maximin Latin hypercube designs. *Operations Research 57*(3), 595–608.

van der Kam, M. and W. van Sark (2015). Smart charging of electric vehicles with photovoltaic power and vehicle-to-grid technology in a microgrid; a case study. *Applied Energy 152*, 20–30.

Vielma, J. P., S. Ahmed, and G. Nemhauser (2010a). Mixed-integer models for nonseparable piecewise-linear optimization: unifying framework and extensions. *Operations Research 58*, 303–315.

Vielma, J. P. and G. Nemhauser (2010b). Modeling disjunctive constraints with a logarithmic number of binary variables and constraints. *Mathematical Programming 128*, 49–72.

Wicaksono, D. S. and I. A. Karimi (2008). Piecewise MILP under- and overestimators for global optimization of bilinear programs. *AIChE Journal 54*(4), 991–1008.

Ye, K. Q. (1998). Orthogonal column Latin hypercubes and their application in computer experiments. *Journal of the American Statistical Association 93*(444), 1430–1439.

Zhang, D., S. Evangelisti, P. Lettieri, and L. G. Papageorgiou (2016). Economic and environmental scheduling of smart homes with microgrid: DER operation and electrical tasks. *Energy Conversion and Management 110*, 113–124.