Parametric Distributionally Robust Optimisation Models for Resource and Inventory Planning Problems

Ben Black, B.Sc.(Hons.), M.Sc., M.Res

Submitted for the degree of Doctor of

Philosophy at Lancaster University.

November 2022

Abstract

Parametric probability distributions are commonly used for modelling uncertain demand and other random elements in stochastic optimisation models. However, when the distribution is not known exactly, it is more common that the distribution is either replaced by an empirical estimate or a non-parametric ambiguity set is built around this estimated distribution. In the latter case, we can then hedge against distributional ambiguity by optimising against the worst-case objective value over all distributions in the ambiguity set. This methodology is referred to as distributionally robust optimisation. When applying this approach, the ambiguity set necessarily contains non-parametric distributions. Therefore, applying this approach often means that any information about the true distribution's parametric family is lost.

This thesis introduces a novel framework for building and solving optimisation models under ambiguous parametric probability distributions. Instead of building an ambiguity set for the true distribution, we build an ambiguity set for its parameters. Every distribution considered by the model is then a member of the same parametric family as the true distribution. We reformulate the model using discretisation of the ambiguity set, which can result in a large, complex problem that is slow to solve.

We first develop the parametric distributionally robust optimisation framework for a workforce planning problem under binomial demands. We then study a budgeted, multi-period newsvendor model under Poisson and normal demands. In these first two cases, we develop fast heuristic cutting surface algorithms using theoretical properties of the cost function. Finally, we extend the framework into the dynamic decision making space via robust Markov decision processes. We develop a novel projection-based bisection search algorithm that completely eliminates the need for discretisation of the ambiguity set. In each case, we perform extensive computational experiments to show that our algorithms offer significant reductions in run times with only negligible losses in solution quality.

Acknowledgements

Firstly, I want to thank my supervisors, Chris and Vikram, for all of the support over the last 3+ years. We had a rocky start, but you stuck with me all the way and helped me make the project into something I could be proud of. Thanks for always pointing me in the right direction, and staying patient during long meetings and even longer proof-reads. I would also like to thank Russell, my BT supervisor, for his support and advice over the years. Our early meetings gave me the direction and inspiration that I needed to set me on the right path.

Secondly, I'd like to thank my family and friends. Mum and Dad: thanks for being there for me 24/7. We lost a lot over the last 4 years, but your support and advice has helped me stay strong throughout. Also, thanks for letting me rant about Better Call Saul for hours without telling me to shut up. Louis, thanks for continuing to be a great friend in the 4 years since I left Cardiff. I hope our normal pub trips can resume soon. To my Swedish friend Robin - I never knew I could have such a good mate that I'd never even met. Thanks for putting up with me and my irritating behaviour over the last few years.

Finally, thanks to all of the STOR-i staff. Jon, Kevin, Idris, Wendy, Kim, Nicky... I really appreciate everything you've done to make me feel welcome at STOR-i and to provide me with all of the training I needed to succeed in my PhD research. I am grateful to have been part of this CDT and am glad that I chose to join it. Note: I am not just saying this because of the free food and wine.

Declaration

I declare that the work in this thesis has been done by myself and has not been submitted elsewhere for the award of any other degree.

This thesis is constructed as a series of papers. Therefore, Chapters 2, 3 and 4 should be read as individual pieces of research. However, Chapters 3 and 4 can be viewed as significant extensions of the concepts introduced in Chapter 2. Each chapter's appendices are included as a chapter in the Appendices at the end of the thesis.

The content of Chapter 2 has been accepted for publication as Black, B., Ainslie, R., Dokka, T., and Kirkbride, C. (2022). Distributionally robust resource planning under binomial demand intakes. *European Journal of Operational Research*.

The content of Chapter 3 has been submitted for publication as Black, B., Dokka, T., and Kirkbride, C. (2022). Parametric distributionally robust optimisation models for budgeted multi-period newsvendor problems. This paper is under review.

The content of Chapter 4 has been submitted for publication as Black, B., Dokka, T., and Kirkbride, C. (2022). Robust Markov decision processes under parametric transition distributions. This paper is also currently under review.

Please note that some of the notation in Chapters 2, 3 and 4 has been modified from the versions submitted to journals, to ensure consistency between chapters in this thesis. In addition, note that we do not provide a literature review chapter since Chapters 2, 3 and 4 each have their own detailed literature review sections.

This word count for this thesis is approximately 52,000 words.

Ben Black

Contents

List of Figures

List of Tables

List of Abbreviations

AO	Approximate objective.
APE	Absolute percentage error.
APG	Absolute percentage gap.
BS	Bisection search.
CDF	Cumulative distribution function.
CQP	Convex quadratic program(ming).
CS	Cutting surface.
CVaR	Conditional value at risk.
DF	Distribution free.
DP	Dynamic program(ming).
DR	Distributionally robust.
DRO	Distributionally robust optimisation.
EV	Expected value.
FD	Fixed distribution.
KKT	Karush-Kuhn-Tucker.
KLD	Kullback-Leibler divergence.

LHS Left-hand side.

LP Linear program(ming).

MDP Markov decision process.

MILP Mixed integer linear program(ming).

MLE Maximum likelihood estimate.

MPNVP Multi-period newsvendor problem.

OR Operational research.

P Parametric (model).

PBS Parametric bisection search.

PMF Probability mass function.

PLA Piecewise linear approximation.

QP Quadratic program(ming).

RMDP Robust Markov decision process.

RO Robust optimisation.

SCP Semi-infinite convex program.

SLSQP Sequential least-squares quadratic programming.

TC Trust constraint.

NBS Non-parametric bisection search.

NP Non-parametric (model).

List of Key Symbols

t	Index of a period in an optimisation model.
T	Number of periods in model.
$\mathbb{E}$	Expectation operator.
$\mathbb{P}(\cdot)$	Probability of $\cdot$.
$\mathbb{N}_0$	Set of non-negative integers.
$\mathbb{R}_+$	Set of non-negative real numbers.
$\boldsymbol{P}, \boldsymbol{Q}$	Probability distributions.
$\boldsymbol{P}^0$	True distribution of uncertain parameter.
$\mathcal{P}$	Ambiguity set for $\boldsymbol{P}^0$.
$\boldsymbol{\theta}$	Vector of parameters of a probability distribution.
$\boldsymbol{\theta}^0$	True parameter vector of a probability distribution.
Θ	Ambiguity set for $\boldsymbol{\theta}^0$.
ϕ	ϕ-divergence function.
d_ϕ	ϕ-divergence distance measure for function ϕ.
κ	Upper bound on $d_\phi(\boldsymbol{P}, \boldsymbol{Q})$ defining an ambiguity set.
$\mathcal{P}_\kappa$	Ambiguity set obtained resulting from the constraint $d_\phi(\boldsymbol{P}, \boldsymbol{Q}) \leq \kappa$.

ϕ^*	Convex conjugate of ϕ-divergence function ϕ.
$\hat{\boldsymbol{\theta}}$	Maximum likelihood estimate (MLE) of $\boldsymbol{\theta}$
N	Number of samples from true distribution used to create MLE.
α	Confidence level used to create confidence sets.
$\boldsymbol{P}^{\boldsymbol{\theta}}$	Parametric probability distribution corresponding to parameter $\boldsymbol{\theta}$.
$\hat{\boldsymbol{P}}$	Parametric probability distribution corresponding to parameter $\hat{\boldsymbol{\theta}}$.
o	Dimension of Θ.
$\chi^2_{o,1-\alpha}$	$100(1-\alpha)^{\text{th}}$ percentile of the χ^2 distribution with o degrees of freedom.
$\mathcal{P}_\Theta$	Set of parametric distributions parameterised by some $\boldsymbol{\theta} \in \Theta$.
f_X	Probability mass function of a random variable X.
F	A cumulative distribution function.
$\boldsymbol{p}$	Binomial success probability vector. $(p_1, \ldots, p_T)$ in Chapter 2, $(p_{s,a})_{s\in\mathcal{S},a\in\mathcal{A}}$ in Chapter 4.
$\boldsymbol{P}^{\boldsymbol{p}}$	Binomial distribution corresponding to success probability $\boldsymbol{p}$.
$\boldsymbol{p}^0$	True success probability of a random variable.
Θ_{base}	A baseline ambiguity set from which contains a confidence set.
Θ'_{base}	Discretisation of Θ_{base}.
M	Number of points used to discretise an interval.
Θ_α	$100(1-\alpha)\%$ confidence set for true parameter around the MLE.
Θ'_α	Discretisation of Θ_α.
$I_{\mathbb{E}}(\boldsymbol{\theta})$	Expected Fisher information at $\boldsymbol{\theta}$.
Θ^{ext}	Set of extreme parameters used by cutting surface algorithm.

$\boldsymbol{q}$	Vector $(q_1, \ldots, q_T)$ of order quantities.
$\mathcal{T}$	Set $\{1, \ldots, T\}$ of periods in the newsvendor model of Chapter 3.
W	The newsvendor's spending budget.
w_t	Cost of ordering a unit of stock in period t in the newsvendor problem.
h	Holding cost in newsvendor problem.
b	Backorder cost in newsvendor problem.
c	Price customer pays to purchase one unit of stock from newsvendor.
X_t	Demand random variable for period t in newsvendor model.
$C_F(\boldsymbol{q})$	Expected cost of order quantity $\boldsymbol{q}$ if the CDF of $\boldsymbol{X}$ is F in Chapter 3.
$C_{\boldsymbol{\theta}}(\boldsymbol{q})$	Expected cost of order quantity $\boldsymbol{q}$ if the CDF of $\boldsymbol{X}$ is parameterised by $\boldsymbol{\theta}$ in Chapter 3.
$\boldsymbol{\lambda}$	Vector of Poisson parameters $(\lambda_1, \ldots, \lambda_T)$.
$\tilde{F}_t$	CDF of $\sum_{l=1}^{t} X_l$ in Chapter 3.
s	Index of a state in MDP problem.
a	Index of an action in MDP problem.
s'	Candidate for next state in MDP problem.
$\bar{s}$	Post-action state in MDP problem.
$\mathcal{A}$	Set of all actions in MDP problem.
S	Number of states in MDP problem.
A	Number of actions in MDP problem.
γ	Discount parameter in MDP problem.
Δ_n	Probability simplex in $\mathbb{R}^n$.

$\boldsymbol{P}_s$	Transition matrix corresponding to state s.
$P_{s,a,s'}$	Prob. of transitioning to state s' when taking action a in state s.
$r_{s,a,s'}$	One-period reward for taking action a in state s and transitioning to state s'.
$\boldsymbol{\pi}$	A policy such that $\pi_{s,a}$ specifies the probability of taking action a in state s.
$\boldsymbol{v}^n$	Value function estimate at iteration n of value iteration: $(v_1^n, \ldots, v_S^n)$.
$\mathcal{P}_s$	Ambiguity set for state s's transition matrix $\boldsymbol{P}_s^0$.
d_a	ϕ-divergence used to measure distance from $\boldsymbol{P}_{s,a}$ for any s.
$\mathfrak{P}(\hat{\boldsymbol{P}}_{s,a}; \boldsymbol{b}, \beta)$	Non-parametric simplex projection problem used to solve a robust Bellman update.
$X_{s,a}$	Exogenous random variable affecting state transitions when action a is taken in state s.
$g(x\|s,a)$	The next state given action a is taken in state s and $X_{s,a} = x$.
$\mathcal{X}_{s,a}$	Support set for $X_{s,a}$.
$\mathcal{X}_{s,a}(s')$	Elements x of $\mathcal{X}_{s,a}$ such that the next state is s' if action a is taken in state s and $X_{s,a} = x$.
$\tilde{\mathfrak{P}}(\hat{\theta}_{s,a}; \boldsymbol{b}, \beta)$	Parametric projection problem used to solve a robust Bellman update.
b'	Stockout cost in MDP newsvendor problem.
C	Capacity of MDP newsvendor problem.
$\Theta_s^{\alpha}, \Theta_s^{\text{base}}$	Confidence set and baseline ambiguity set for parameters of state s in MDP model.
Θ_s'	Discretisation of ambiguity set for parameters of state s.

Chapter 1

Introduction

In this chapter, we provide motivation for the research in this thesis and introduce a number of relevant concepts and methodologies. Section 1.1 describes the motivation for the content of this thesis. Following this, Section 1.2 introduces basic deterministic optimisation. Section 1.3 then introduces optimisation under parameter uncertainty, describing existing methodologies and the parametric framework that we develop in this thesis. Following this, Section 1.4 details the contributions that we have made. Finally, Section 1.5 gives an outline of the structure of the thesis.

1.1 Motivation

While this thesis considers a more general class of uncertain optimisation problems, its initial direction was motivated by the planning problems faced by a large UK service industry company. In particular, this company is a telecommunications company,

providing services such as television, broadband and telephone to customers across the entirety of the UK. The company employs around 20,000 technical engineers that are tasked with installing and maintaining the equipment that allows customers to consume their services. In order to maximise profits and minimise the number of dissatisfied customers, extensive planning is employed.

The particular planning problem that we are motivated by is optimising the use of the workforce's hours in order to meet as much demand as possible over a given planning horizon. In particular, we consider deciding upon how many jobs to complete on each day by completing some jobs prior to their due dates. Two types of jobs must be planned for: *installation* and *repair* jobs. Installation jobs correspond to engineers installing new equipment, such as routers or cable networks, that allow new customers to use the company's services. Repair jobs correspond to engineers repairing existing equipment for current customers, so that they can continue to use the services. At the time of planning, there are some jobs due on each day in the planning horizon that we know about. However, due to the unpredictable nature of breakages in equipment and new customer arrivals, the number of jobs that will arrive in the system between the time of planning and the period concerned (*intake* jobs) is uncertain.

Dealing with the uncertainty caused by the number of intake jobs is of great importance to the performance of plans. However, incorporating uncertainty often leads to greatly increased solution times for large models such as these. A simple approach used in some papers has been to forecast the number of intake jobs and use the forecast as the truth (Ainslie et al., 2015, 2018). However, this makes the performance

of the plan directly dependent on the accuracy of the forecast. Over-predicting the number of intakes may lead to wasted capacity, and under-predicting the number of intakes can cause jobs to be left incomplete. In any case, the company either wastes money or is left with dissatisfied customers. Another approach is to treat the number of intakes for each day as a random variable with a known distribution (Zhu and Sherali, 2009; Ross, 2016). In practice, however, these distributions cannot be known exactly and must be estimated from historical data. In this thesis, we therefore do not assume that any distributions are known exactly. Instead, we explicitly incorporate distributional ambiguity in order to hedge against the effects of poor estimation.

In this planning problem, it is important that our models utilise all information on the intake random variable's distribution in order to ensure that its behaviour is adequately represented. We make the following observations on this random variable. Firstly, it is a count variable, meaning that it is discrete. Secondly, the number of repair jobs is bounded by the number of devices and the maximum number of installation jobs can be fixed by the company. Hence, the number of intakes for a given day is bounded above. Finally, jobs arrive independently of one another. Therefore, this random variable is well-described by a binomial distribution.

Parametric distributions are commonly used for demand in stochastic optimisation problems, particularly in the newsvendor problem when the distribution is known (Nahmias, 1994; Agrawal and Smith, 1996; Gallego et al., 2007; Rossi et al., 2014). However, in problems where the distribution of demand is not known exactly (such as our planning problem), the information that the true distribution lies in a given

parametric family is often ignored. This is done since incorporating this additional information results in greatly increased model complexity. Instead of incorporating this information, models usually represent distributional ambiguity using sets of candidate non-parametric or semi-parametric distributions. However, sets of this kind can contain distributions that do not belong to the same parametric family as the true distribution was assumed to lie in. This means that any modelling conveniences that led to the use of parametric distributions are lost in the model's output.

This thesis is more generally motivated by a wider class of problems similar to this planning problem. In particular, we focus on optimisation problems with the following characteristics. Firstly, some of the model's parameters are random. Secondly, we cannot know their distributions exactly. Finally, parametric distributions provide a logical and appropriate way to model their behaviour. The main goal of this thesis is to improve the modelling of such problems by not only representing distributional ambiguity, but also incorporating and preserving all information about the parametric family in which the true distribution lies. While many approaches exist for inorporating distributional ambiguity, the most common approaches do not preserve all distributional information. This thesis develops methods for doing so, and assesses the benefit of using such methods over more common ones.

Our framework for accomplishing this goal can be described as follows. We explicitly incorporate and preserve the parametric family of the true distribution, only considering distributions that lie in this family. We use maximum likelihood estimation in order to build confidence sets for the true parameters. Our models then optimise

against the worst-case parameters in this set. This allows us to integrate estimation and optimisation, hedge against distributional ambiguity and parameter uncertainty, and also ensure that any distributions returned by the model lie in the same parametric family that the true distribution was assumed to lie in.

1.2 Deterministic Optimisation

In many real-world decision-making problems, we are tasked with selecting the best decision from a set of potential choices. Since this set can become very large and complex, solving the real problem directly is often challenging. Operational Research (OR) consists of creating and solving mathematical models for such problems, which are easier to solve than the problems themselves.

Denote by $\boldsymbol{x}$ a vector of n *decision variables*[1]. These represent the values that we are in control of and wish to decide on the values of. The set of values that $\boldsymbol{x}$ can take is represented by a set of *constraints*. We measure the performance of the $\boldsymbol{x}$ using an *objective function*, denoted by f. Provided that the model adequately represents the true problem, solving the real-world problem is then equivalent to finding the value of $\boldsymbol{x}$ that either minimises or maximises the objective function over all those that satisfy the constraints. A general mathematical model representing a decision-making problem is of the form shown in (1.2.1)-(1.2.3).

$$\min_{\boldsymbol{x}} h(\boldsymbol{x}) \tag{1.2.1}$$

$$\text{s.t. } g_i(\boldsymbol{x}) \leq \mathbf{0} \ \forall \ i \in \{1, \ldots, m\} \tag{1.2.2}$$

[1]Note that the notation used in this chapter may be re-defined elsewhere in the thesis.

$$\boldsymbol{x} \in \mathbb{R}_{+}^{n_1} \times \mathbb{N}_{0}^{n_2}. \tag{1.2.3}$$

Here, (1.2.1) represents the objective of the model, (1.2.2) defines the constraints, and (1.2.3) defines which variables are allowed to be continuous and which must be integer (with $n_1+n_2 = n$). The methodology used to solve a model of this form, where all parameters are known, is referred to as *deterministic optimisation*. This framework encompasses a large selection of methods and algorithms that differ depending on the form that f and the g_i take, and also on the values of n_1 and n_2. The most simple model occurs when all variables are continuous and f and the g_i are all *linear*, i.e.

$$f(\boldsymbol{x}) = \sum_{j=1}^{n} c_j x_j, \quad g_i(\boldsymbol{x}) = \sum_{j=1}^{n} A_{i,j} x_j - b_i \; \forall \; i \in \{1, \ldots, m\}, \quad n_1 = n, \tag{1.2.4}$$

where $\boldsymbol{c} \in \mathbb{R}^n$ is a known vector of *objective coefficients*, $A \in \mathbb{R}^{m \times n}$ is a known *constraint matrix*, and $\boldsymbol{b} \in \mathbb{R}^m$ provides the constraints' right-hand sides. In this case, the model is referred to as a *linear program* (LP)[2] and can be solved to optimality via the simplex algorithm (Dantzig, 1960).

Any deviation from the LP format results in increased model complexity. For example, when some variables must be integer, the model becomes a *mixed integer linear program* (MILP). Such problems can no longer be solved via the simplex algorithm, and other methods such as branch and bound (Land and Doig, 1960) must be applied. This method consists of solving a sequence of LP relaxations, each time generating bounds on the optimal objective value. When either f or g are non-linear, the problem becomes a *non-linear program* (NLP). For such problems, an entirely different

[2]Definitions of key terminology and abbreviations are given again in later chapters, in case readers prefer not to read the thesis in order.

set of methods based on a set of first-order optimality conditions referred to as the *Karush-Kuhn-Tucker* (KKT) conditions (Kuhn and Tucker, 1951) are employed.

This thesis considers LPs, MILPs and NLPs. While different methods are applied to each, one characteristic is common among them all. Although optimal algorithms often exist, they can solve slowly when n and/or m are large. Hence, we develop our own *heuristic* algorithms to solve our models. A heuristic algorithm is one that is designed to provide near-optimal solutions in a short time.

1.3 Optimisation Under Parameter Uncertainty

The models discussed in Section 1.2 assume that all parameters are known. However, uncertainty is inherent in many real-world OR problems since it is often impossible to calculate all of a model's parameters exactly. Of particular importance to this thesis is uncertain demand, which is perhaps the most common. This section introduces some methodologies that have been developed to handle such uncertainties in optimisation models, along with the novel methodology that we develop in this thesis.

1.3.1 Existing Methodologies

Many methodologies for dealing with uncertainty in optimisation problems have been developed over the years. In order to introduce these methodologies, consider the optimisation problem given in (1.2.1)-(1.2.3). Furthermore, suppose that some vector or matrix of parameters, $\boldsymbol{Y}$, is unknown. In the case of an LP, we may have that $\boldsymbol{Y} = \boldsymbol{c}$ or $\boldsymbol{Y} = A$, for example. We will treat $\boldsymbol{Y}$ as a random variable with support

set $\mathcal{Y}$. The first methodology that we discuss for modelling uncertainty is *robust optimisation* (Ben-Tal and Nemirovski, 1998) (RO). RO constructs an *uncertainty set* $\mathcal{U} \subseteq \mathcal{Y}$ containing potential values for the uncertain parameters. It then optimises against the worst-case value of the objective function over all realisations of $\boldsymbol{Y}$ in the uncertainty set, while ensuring that the solution of the problem is feasible for all realisations. A robust version of (1.2.1)-(1.2.3) is formulated as follows:

$$\begin{aligned} &\min_{\boldsymbol{x}} \ \max_{\boldsymbol{y} \in \mathcal{U}} h(\boldsymbol{x}, \boldsymbol{y}) \\ &\text{s.t. } g_i(\boldsymbol{x}, \boldsymbol{y}) \leq \boldsymbol{0} \ \forall \ i \in \{1, \ldots, m\}, \ \forall \ \boldsymbol{y} \in \mathcal{U} \\ &\qquad \boldsymbol{x} \in \mathbb{R}_+^{n_1} \times \mathbb{N}_0^{n_2}. \end{aligned}$$

RO hedges against uncertainty by protecting against the very worst realisations of the uncertain parameter. However, due to the risk-averse nature of this worst-case approach, RO can produce overly conservative solutions.

Stochastic programming (SP) instead assumes that the unknown parameters are random variables with known probability distributions (Kall et al., 1994). Suppose that the distribution of $\boldsymbol{Y}$ is $\boldsymbol{P}^0$. The most basic stochastic programming model assumes that $\boldsymbol{Y}$ only affects the objective function, and can be formulated as:

$$\begin{aligned} &\min_{\boldsymbol{x}} \ \mathbb{E}_{\boldsymbol{P}^0} \left[h(\boldsymbol{x}, \boldsymbol{Y}) \right] \\ &\text{s.t. } g_i(\boldsymbol{x}) \leq \boldsymbol{0} \ \forall \ i \in \{1, \ldots, m\} \\ &\qquad \boldsymbol{x} \in \mathbb{R}_+^{n_1} \times \mathbb{N}_0^{n_2}. \end{aligned}$$

Where RO models can be viewed as hedging against the worst possible realisation of $\boldsymbol{Y}$, SP models such as this one account for the effects of all potential realisations

without assuming that the worst will always occur.

However, similarly to how it is not always possible to know a parameter's value exactly, it is also not always possible to know its probability distribution exactly. In such cases, *distributionally robust optimisation* (DRO) is an appropriate methodology. DRO combines elements from RO and SP; it constructs an *ambiguity set* $\mathcal{P}$ containing candidates for the true distribution $\boldsymbol{P}^0$ of the uncertain parameters, and optimises against the worst-case distribution in this set (Ben-Tal et al., 2013). A general DRO model is of the form:

$$\begin{aligned} &\min_{\boldsymbol{x}} \max_{\boldsymbol{P}\in\mathcal{P}} \mathbb{E}_{\boldsymbol{P}}\left[f(\boldsymbol{x},\boldsymbol{Y})\right] \\ &\text{s.t. } g_i(\boldsymbol{x}) \leq \boldsymbol{0} \;\forall\; i \in \{1,\ldots,m\} \\ &\boldsymbol{x} \in \mathbb{R}_+^{n_1} \times \mathbb{N}_0^{n_2}. \end{aligned}$$

The ambiguity set $\mathcal{P}$ incorporates any available information on the true distribution. For example, early models assumed that the first two moments were known (Scarf, 1957), and the ambiguity set contained all distributions with said moments. The most common ambiguity sets in the recent DRO literature, however, have been *distance-based* non-parametric ambiguity sets. Distance-based ambiguity sets contain all distributions that lie within some pre-prescribed distance from a nominal distribution, as measured by some distance measure. Examples of distance measures used to construct ambiguity sets include ϕ-divergences (Ben-Tal et al., 2013), ζ-structure probability metrics (Zhao and Guan, 2015) and the Wasserstein distance (Mohajerin Esfahani and Kuhn, 2018).

Non-parametric distance-based DRO methods typically do not incorporate any information about the moments or parameters of the true distribution. Other DRO methods allow users to specify and control such parameters, enforcing that all distributions in the ambiguity set share these parameters. Marginal distribution models allow users to provide complete marginal distributions, and then build ambiguity sets containing all joint distributions with these marginals. The first example of a marginal distribution DRO model came from Meilijson and Nádas (1979), who studied calculating critical paths in a PERT network when only the marginal distributions of project lengths were known. The framework has recently been used as a more general framework for solving discrete choice models with uncertain objective coefficients (Natarajan et al., 2009).

Marginal distribution models allow us to ensure that the worst-case distribution has the required marginals, but allows the covariance among the random variables to vary freely. While assuming that we know the marginal distribution is not always realistic, this may be appropriate when we have strong empirical estimates of the marginals and are more concerned about how they interact via correlation or covariance. In the case where parametric distributions are appropriate for the marginals, this approach does ensure that the marginals of the worst-case distribution lie within the correct family. However, it can only be used when the parameter estimates are accurate.

Semi-parametric models are another type of DRO model that differ from the distance-based ones that we have discussed. Semi-parametric models consider scenarios where the distribution is not known, but some of its moments can be controlled by the

decision maker. The ambiguity set then contains all distributions with the specified moment information. The parameters controlled by the decision maker can encompass any number of the distribution's moments. An example of a semi-parametric model comes from Ahipasaoglu et al. (2019), who studied distributionally robust project crashing. In their model, they assumed that they could control the first two moments of the project duration. The covariance matrix of the project durations was considered as uncontrollable, but if it was known then it could be specified. Then, the ambiguity set would only contain distributions with the given covariance matrix.

Semi-parametric models are applicable when the random variables of interest are under at least partial control of the decision maker. In the problem of Ahipasaoglu et al. (2019), the project durations can be controlled by allocating more or less resources to them. In the context of demand for a product or service, it may be possible to control the mean of the distribution by increasing expenditure on advertising. However, since demand is exogenous, it is not guaranteed that the mean demand will be affected by this in the way intended. Therefore, it is more common in demand modelling to treat such parameters as unknown and uncontrollable. In this context, it is more appropriate to estimate the parameters than to try to control them.

In recent years, DRO has gained significant traction in the OR literature for a number of reasons. For example, it maintains some of the risk-aversion of RO whilst still incorporating probabilistic information about the uncertain parameters and not assuming that the worst-case realisations will occur. In addition, when distance-based ambiguity sets are used, maximum likelihood estimation can be utilised to create con-

fidence sets for the true distribution (Ben-Tal et al., 2013). This allows the decision maker to directly control the risk-aversion associated with the model via tuning the confidence level of the confidence set. Furthermore, DRO models with distance-based ambiguity sets often yield linear or convex quadratic programming reformulations via Lagrangian duality (Bayraksan and Love, 2015). When the reformulations are slow to solve, fast iterative methods such as *cutting surface* algorithms (Mehrotra and Papp, 2014) can be applied. Cutting surface algorithms iteratively solve the DRO model over increasing discrete subsets of the ambiguity set. Therefore, DRO models provide a way to incorporate distributional ambiguity without sacrificing tractability.

Another key methodology for modelling uncertainty is *Markov decision processes* (Bellman, 1957) (MDPs). In an MDP, a decision maker is in partial control of a stochastically evolving system. At each *epoch* $t \in \mathcal{T} = \{0, \ldots, \infty\}$, they take an *action* $a_t \in \mathcal{A}$ based on some limited information about the *state* of the system. This information is represented by the state variable $s_t \in \mathcal{S}$. The state of the system then becomes $s_{t+1} \in \mathcal{S}$, and the decision maker receives a *one-period reward* $r_{s_t,a_t,s_{t+1}}$. The evolution of the system is described by a *transition matrix* $\boldsymbol{P}^0$, where $P^0_{s,a,s'}$ gives the probability of transitioning to state s' given that action a is taken when in state s, for each $(s, a, s') \in \mathcal{S} \times \mathcal{A} \times \mathcal{S}$.

A solution of an MDP is a *policy* $\boldsymbol{\pi}$ such that $\pi_{s,a}$ gives the probability of taking action a when in state s, for each $s \in \mathcal{S}$ and $a \in \mathcal{A}$. The goal of the decision maker is to determine the policy that maximises their total discounted expected reward:

$$\max_{\boldsymbol{\pi} \in \Pi} \mathbb{E}_{\boldsymbol{P}^0, \boldsymbol{\pi}} \left[\sum_{t=0}^{\infty} \gamma^t r_{s_t,a_t,s_{t+1}} \middle| S_0 \sim \boldsymbol{Q} \right],$$

where γ is a *discount factor* and $\boldsymbol{Q}$ is the known distribution of the initial state random variable S_0. When $\boldsymbol{P}^0$ is known, the MDP can be solved via dynamic programming (Bellman, 1966). Of more interest to this thesis, however, is the case where $\boldsymbol{P}^0$ is not known. In such cases, it has been found that replacing the true transition distributions with estimates can result in decision-making policies that fail drastically when implemented (Le Tallec, 2007), and large variance and bias in the estimates of outputs (Mannor et al., 2007). As a result of this, *robust MDPs* (RMDPs) were introduced (Satia and Lave, 1973). They employ similar concepts to DRO; they construct an ambiguity set $\mathcal{P}$ for $\boldsymbol{P}^0$ and optimise against the worst-case:

$$\max_{\boldsymbol{\pi}\in\Pi} \min_{\boldsymbol{P}\in\mathcal{P}} \mathbb{E}_{\boldsymbol{P},\boldsymbol{\pi}} \left[\sum_{t=0}^{\infty} \gamma^t r_{s_t,a_t,s_{t+1}} \middle| S_0 \sim \boldsymbol{Q} \right].$$

While early ambiguity sets focused on bounding probabilities (Satia and Lave, 1973; Givan et al., 2000), distance-based ambiguity sets soon became the standard for RMDPs. Examples of distances measures used to construct RMDP ambiguity sets include the Kullback-Leibler divergence, χ^2 distance and L_1-norms (Iyengar, 2005), and more recently the general class of ϕ-divergences (Ho et al., 2022). The most common algorithm for solving RMDPs is *robust value iteration*, an iterative algorithm that solves a DRO problem (referred to as a *robust Bellman update*) in each iteration. Since a robust Bellman update is a DRO problem, each iteration of robust value iteration is often as simple as solving a linear or convex quadratic program.

1.3.2 Parametric Framework for DRO and RMDPs

The main issue that this thesis aims to address with respect to optimisation under uncertainty is the treatment of parametric distributions in DRO and RMDPs. In many uncertain optimisation problems, the true distribution is assumed to lie in some parametric family. When the distribution or its parameters are unknown, DRO models build ambiguity sets containing candidates for the true distribution.

It is generally understood that all information that helps to characterise the true distribution should be incorporated into the ambiguity set. We can then enforce that all distributions in the ambiguity set share this information. For example, when some moments of the distribution are assumed to be known, semi-parametric models can enforce that the worst-case distribution has these moments. When the marginal distributions are assumed to be known, marginal distribution models enforce that the worst-case distribution has these marginals. However, when the family in which the true distribution lies is assumed to be known, no DRO methodologies exist that enforce that the worsts-case distribution lies in this family.

PDRO is a framework that is designed for problems where parametric distributions provide strong models for the uncertain parameters, but nothing else about the true distribution is known. In such cases, marginal distribution and semi-parametric models are not appropriate. For such problems, one might consider using distance-based non-parametric ambiguity sets. While these do not make assumptions about any of the model's parameters or marginals, they also ignore the assumption made about the family of the true distribution. Therefore, worst-case distributions from such models

do not necessarily satisfy this assumption. This may result in worst-case distributions that are not reasonable models for the true parameters. PDRO alters the formulations used in DRO and RMDPs to strictly enforce that every distribution considered lies in the assumed parametric family.

We enforce this by optimising against the worst-case parameters instead of the worst-case distribution, and using the probability mass/density function (PMF/PDF) in the objective function directly. More specifically, suppose that $\boldsymbol{P}^0 = \boldsymbol{P}^{\boldsymbol{\theta}^0}$ is assumed to lie in some set $\mathcal{P}_\Theta = \left\{\boldsymbol{P}^{\boldsymbol{\theta}} : \boldsymbol{\theta} \in \Theta\right\}$. In other words, the distribution $\boldsymbol{P}^{\boldsymbol{\theta}}$ can be calculated entirely from $\boldsymbol{\theta}$ and $\mathcal{Y}$ via a PMF or PDF $f_{\boldsymbol{Y}}$:

$$\boldsymbol{P}^{\boldsymbol{\theta}} = \left(f_{\boldsymbol{Y}}(\boldsymbol{y}|\boldsymbol{\theta})\right)_{\boldsymbol{y}\in\mathcal{Y}},$$

for each $\boldsymbol{\theta} \in \Theta$. Let us define the feasible region for $\boldsymbol{x}$ as:

$$\mathcal{X} = \left\{\boldsymbol{x} \in \mathbb{R}_+^{n_1} \times \mathbb{N}_0^{n_2} : g_i(\boldsymbol{x}) \leq \boldsymbol{0}\ \forall\ i \in \{1, \ldots, m\}\right\}.$$

Then, a general PDRO model can then be written as:

$$\min_{\boldsymbol{x}\in\mathcal{X}} \max_{\boldsymbol{\theta}\in\Theta} \mathbb{E}_{\boldsymbol{\theta}}\left[h(\boldsymbol{x}, \boldsymbol{Y})\right], \tag{1.3.1}$$

where:

$$\mathbb{E}_{\boldsymbol{\theta}}\left[h(\boldsymbol{x}, \boldsymbol{Y})\right] = \sum_{\boldsymbol{y}\in\mathcal{Y}} h(\boldsymbol{x}, \boldsymbol{y}) f_{\boldsymbol{Y}}(\boldsymbol{y}|\boldsymbol{\theta}) \tag{1.3.2}$$

when $\mathcal{Y}$ is discrete, or:

$$\mathbb{E}_{\boldsymbol{\theta}}\left[h(\boldsymbol{x}, \boldsymbol{Y})\right] = \int_{\mathcal{Y}} h(\boldsymbol{x}, \boldsymbol{y}) f_{\boldsymbol{Y}}(\boldsymbol{y}|\boldsymbol{\theta}) \mathrm{d}\boldsymbol{y}. \tag{1.3.3}$$

if it is continuous. PDRO is our framework for building and solving models of the form (1.3.1). In general, the framework works as follows:

1. **Decide on parametric family.** We must first decide upon the parametric family that best models our random variable $\boldsymbol{Y}$. This can be done based on expert knowledge or model fitting.

2. **Re-write objective function.** Explicitly calculating the expected value and performing simplifications can result in a more convenient expression.

3. **Analyse objective function as a function of $\boldsymbol{\theta}$.** We aim to characterise the worst-case parameter for a given $\boldsymbol{x}$ decision.

4. **Take samples and calculate MLE.** Denote by $\hat{\boldsymbol{\theta}}$ the MLE of $\boldsymbol{\theta}^0$.

5. **Construct ambiguity set.** We typically use an approximate $100(1-\alpha)\%$ confidence set, Θ_α, for $\boldsymbol{\theta}^0$ centred around $\hat{\boldsymbol{\theta}}$ as an ambiguity set.

6. **Discretise ambiguity set.** This gives a discrete subset of the ambiguity set, $\Theta' \subseteq \Theta$.

7. **Reformulate and solve.** We reformulate (1.3.1) approximately as follows:

$$\min_{\boldsymbol{x}\in\mathcal{X},\vartheta} \left\{\vartheta : \vartheta \geq \mathbb{E}_{\boldsymbol{\theta}}\left[h(\boldsymbol{x},\boldsymbol{Y})\right] \ \forall\, \boldsymbol{\theta} \in \Theta'\right\}. \tag{1.3.4}$$

   Model (1.3.4) can then be solved using an appropriate method.

The steps involved in PDRO each need to be adjusted depending on the problem at hand. The first characteristic of the problem that affects these steps is whether $\mathcal{Y}$ is discrete and finite, discrete and infinite, or continuous. In the first case, step 2. simply entails calculating the entire distribution for any $\boldsymbol{\theta}$ and treating the PMFs as constant coefficients. The model will be linear if $h(\boldsymbol{x},\boldsymbol{y})$ is linear in $\boldsymbol{x}$. In the

second and third cases, step 2. will need to evaluate all sums and integrals in order to simplify the objective function as much as possible. In general, we aim to re-write the objective function in a simple format with finitely many terms.

The behaviour of the objective function after the simplifications in step 1. affects many of the other steps in PDRO. For example, $\mathbb{E}_{\boldsymbol{\theta}}[h(\boldsymbol{x}, \boldsymbol{Y})]$ may be non-linear in $\boldsymbol{x}$ even if $h(\boldsymbol{x}, \boldsymbol{y})$ is linear in $\boldsymbol{x}$. This can happen when $\mathcal{Y}$ is infinite and/or continuous, and in such cases (1.3.4) may need to be adapted in order to be solvable using standard solvers. If the objective function is neither linear nor quadratic, we will need to apply methods such as piecewise linear approximations or KKT conditions in order to solve it. Step 3. is also affected by the behaviour of the objective function. In step 3. we typically differentiate the objective function w.r.t. each θ_t in order to find theoretical results such as convexity or monotonicity. These results can help build heuristics for solving the model. When the objective function is not differentiable w.r.t. some θ_t, we may need to use less rigorous methods such as plotting the objective function in order to better characterise the worst-case parameter.

Other mathematical properties of the family of distributions chosen in step 1. can also affect the steps used in PDRO. For many families of distributions, the MLEs in step 4. can easily be found using closed form equations. However, for some families such as Gamma or Cauchy, no closed form solution exists. In such cases, finding the MLEs can only be done numerically, making step 4. more complex. In addition, in step 5. we typically use the MLEs to construct confidence sets based on the Wald test statistic. Computing this statistic requires the Fisher information matrix, which is only defined

under certain regularity conditions. One such condition is that the support does not depend on $\boldsymbol{\theta}$. Therefore, for marginal distributions such as the uniform distribution on $[0, \theta_t]$, we cannot use the Wald-based confidence set. In such cases, one might use the likelihood ratio statistic to construct a confidence set instead.

This parametric formulation has a number of benefits. Firstly, finding the worst-case parameters as opposed to the worst-case distribution itself greatly reduces the dimension of the vector that we search for. The worst-case parameter is a small vector consisting of only a few values, whereas the worst-case distribution is a vector with as many entries as the uncertain parameter has realisations. Secondly, it improves the realism of the output, since the worst-case distribution will share the same structural properties as the true distribution since it lies in the same family. In addition, the output is more explainable, since the worst-case distribution can be characterised by only a small number of values. Finally, the use of parametric distributions means that our approach allows us to combine maximum likelihood estimation and optimisation via our ambiguity sets. In particular, we can use MLEs to construct confidence sets for use as ambiguity sets, meaning they have asymptotic probabilistic guarantees.

However, the parametric framework comes with a number of challenges. PMFs/PDFs of parametric distributions as functions of the parameters are usually polynomials or contain exponential functions or logarithms. Therefore, treating the parameters as decision variables means that the model does not have a convenient reformulation[3]

[3]We use the term *convenient reformulation* to loosely mean a reformulation that is solvable using standard solvers, such as an LP, MILP, quadratic or second-order cone program.

via Lagrangian duality. As a result, the standard reformulation methods in DRO and RMDPs no longer apply. Due to this, our methods discretise the ambiguity set of parameters, allowing us to represent the worst-case objective value via a set of expected value constraints. For discrete distributions, this method often relies on computing the corresponding PMF for every parameter in the ambiguity set.

In any case, when this discretised ambiguity set is large, our models have a large number of constraints and present a significant computational burden, making them slow to build and solve. Therefore, a key focus of this thesis is to develop exact and heuristic algorithms for our parametric models that reduce solution times. For every problem considered, we perform computational experiments to show that our algorithms solve the models to near optimality and drastically reduce solution times when compared with solving the models directly.

1.4 Contributions

The major contributions of this thesis are as follows. In this thesis, we:

1. Introduce the novel framework of parametric DRO.
2. Show how to build and solve parametric DRO models under binomial, Poisson and normal random variables, linear and non-linear objective functions, and continuous and integer decision variables.
3. Show how to use maximum likelihood estimation to construct confidence sets for the true parameters, and use them as ambiguity sets in our models.

4. Present a selection of novel heuristic cutting surface algorithms for the parametric DRO models. These algorithms use theoretical results concerning the objective function as a function of the distribution's parameters to develop a small subset of extreme parameters in which to search for the worst-case.

5. Introduce the parametric RMDP framework, based on the aforementioned parametric DRO framework.

6. Develop a bisection search algorithm that solves the parametric robust Bellman updates to ϵ-optimality and does not rely on discretisation of ambiguity sets.

7. Carry out extensive computational experiments to test the efficacy of our algorithms, and compare with non-parametric DRO and RMDP approaches.

1.5 Thesis Outline

In Chapter 2, which is an adapted version of Black et al. (2022a), we study a distributionally robust resource planning problem under binomial demands. We introduce the framework of parametric DRO and develop a novel heuristic cutting surface algorithm for solving the problem. We also present a heuristic that is based on removing intakes that always have a low probability of occurring. We perform computational experiments in order to test our algorithms against the optimal parametric solution and also provide comparisons of the parametric and non-parametric solutions.

In Chapter 3, we extend the framework to solve a static, multi-period, budgeted, distributionally robust newsvendor problem. This newsvendor problem has a non-

linear objective function and we consider both continuous and discrete parametric demand distributions with infinite support. We first develop a heuristic algorithm for the model under a known distribution and test this algorithm against off-the-shelf solvers. Then, we develop cutting surface algorithms for the problem under normal and Poisson demands and test their efficacy using computational experiments. Chapter 3 is adapted from our paper that we submitted to a journal using double-blind reviewing. Hence, no citation for this paper is currently available.

In Chapter 4, which is an adapted version of Black et al. (2022b), we present parametric RMDPs. We extend the concept of parametric DRO into the MDP framework and develop a number of algorithms for carrying out robust value iteration. The main algorithm is a bisection search algorithm that solves each parametric robust Bellman update to ϵ-optimality. We test our algorithms against the non-parametric RMDP solutions, comparing solution times, policies and worst-case distributions. Finally, in Chapter 5, we conclude the thesis and discuss areas for future research.

Chapter 2

Distributionally Robust Resource Planning Under Binomial Demand Intakes

In this chapter, we consider a distributionally robust resource planning model inspired by the real-world service industry problem discussed in Section 1.1. In this problem, there is a mixture of known demand and uncertain future demand. Prior to having full knowledge of the demand, we must decide upon how many jobs we plan to complete on each day in the planning horizon. Any jobs that are not completed by the end of their due date incur a cost and become due the following day.

We present two distributionally robust optimisation (DRO) models for this problem. The first is a non-parametric model with a ϕ-divergence based ambiguity set. The second is a parametric model, where we treat the number of uncertain jobs due on

each day as a binomial random variable with an unknown success probability. We reformulate the parametric model as a mixed integer program and find that it scales poorly with the sizes of the ambiguity and uncertainty sets.

Hence, we make use of theoretical properties of the binomial distribution to derive fast heuristics based on dimension reduction. One algorithm is based on cutting surface algorithms commonly seen in the DRO literature. The other operates on a small subset of the uncertainty set for the future demand. We perform extensive computational experiments to establish the performance of our algorithms. We compare decisions from the parametric and non-parametric models, to assess the benefit of including the binomial information.

2.1 Introduction

In this chapter, we consider a resource planning problem motivated by a real-world telecommunications service company. This real problem consists of optimising the use of a large workforce of service engineers, in the face of a mixture of known and uncertain jobs.

2.1.1 Problem Setting

The planning process for a service company is subject to three stages, named *the three stages of planning*. Each serves a different purpose, covers a different time horizon, and creates results that feed into the next. The three stages are *strategic*, *tactical*, and *operational* planning. Strategic planning covers a period of multiple years, and

concerns long term decisions such as how many employees to be hired and in which skills they should be trained. Tactical planning concerns a period of weeks or months. It involves aggregate decisions such as deciding upon the capacity needed in each period, or how many jobs can and cannot be completed in each period. Operational planning concerns short-term decisions such as scheduling the day-to-day activities of the workforce at the individual level. We focus on the tactical planning stage in this chapter. The decisions that we make are at the *aggregate level*, i.e. we do not plan the specific activities of every worker but we instead aggregate their availability into a daily capacity value. We are tasked with planning the use of this capacity to maximise job completions, or equivalently minimise the number of jobs left incomplete. Since it is typically not possible to move capacity between days, planners manipulate demand to make the best use of what they have.

In the telecommunications industry, jobs can be divided into two categories: *repair* jobs and *installation* jobs. Repair jobs correspond to service engineers being tasked with fixing broken equipment for existing customers, such as broadband routers and telephone systems. Installation jobs correspond to engineers installing equipment in order to obtain new customers. For example, this may be installing new cabling cabinets and networks in order to provide broadband to a new geographical area. Repair jobs are treated as emergency jobs and they are given a high priority for completion. Installation jobs are treated as additional jobs that a company can plan to complete in order to generate more profit.

In this chapter, we will consider planning the activities of a telecommunications work-

force carrying out repair jobs. Since breakages in equipment and services are not planned, these jobs offer a source of uncertainty. In particular, for any given planning period we have knowledge of a fixed number of repair jobs that are already in the system at the time of planning (*workstack jobs*). However, the number of breakages between the time of planning and the date concerned is subject to uncertainty. The jobs generated by these future breakages are referred to as *intake jobs* or *intakes*.

At the time of planning, we have an aggregate capacity value that gives the number of jobs that our workforce can complete, for each day in a planning horizon of fixed length. This is obtained from the number of engineers working on each day, and the number of hours that they will work. By default, we will use all available capacity on each day to complete jobs that are due on that day. Furthermore, workstack jobs can be completed on or before their due date, and completing them early is referred to as *pulling forward*. However, the same does not apply to intake jobs. Since the day that they will arrive in the system is unknown, allowing them to be pulled forward could suggest that they will be completed before they even arrive. Hence, intake jobs cannot be pulled forward.

If any jobs are still incomplete by the end of their due date, then they will not leave the system but incur a cost, and become due on the following day. This is referred to as *rollover*. In this chapter, since capacity is fixed, our model will optimise the pulling forward decision in order to minimise the total rollover cost over the planning horizon. Pulling forward can be utilised to free up capacity on due dates that we expect to have high intake. This helps to reduce rollover and utilise spare capacity.

In the literature on service industry planning models that are closest to ours, demand uncertainty often results in unreasonably large models due to poor scalability. Examples of this come from Ainslie et al. (2015) and Ainslie et al. (2018). In these papers, models had to be solved heuristically due to their size, even though they were deterministic. However, the demand uncertainty is still acknowledged. In fact, in some cases the plan is passed through a predictive model in order to better assess its performance (Ainslie et al., 2017).

The closest model to ours that does model uncertain demand comes from Ross (2016), who used two-stage stochastic programming models for service industry workforce planning. However, this methodology requires the assumption that the demand distribution is known, and this is not an assumption that is reasonable here. The framework that we use to model our problem is *distributionally robust optimisation* (DRO). This framework allows us to include distributional information in our models, without full knowledge of the distributions themselves.

More specifically, we model intakes as binomial random variables where each distribution is ambiguous. Furthermore, we assume that the intake random variables for any two days in the planning horizon are independent of one another. We assume that we have access to a forecasting model or expert knowledge that gives a point estimate of intake and a range of potential values. Hence, for each day, the number of trials is fixed at the maximum intake. Therefore, the success probability is the only unknown parameter for each distribution. This parameter can be estimated through maximum likelihood estimation, with access to past intake data. Our decision to use

the binomial distribution can be justified by the following three reasons:

1. The number of intake jobs due on each day is a discrete quantity and any two jobs arriving on the same day arrive independently of one another.

2. There is a fixed and finite set of values that each intake random variable can take. Other discrete distributions such as the Poisson distribution are unbounded, and hence not fitting for these random variables.

3. Apart from naturality, it gives a concise way of modelling the uncertainty. We can represent each distribution uniquely by one choice of $\boldsymbol{p}$, which is a vector of dimension equal to the number of periods in the planning horizon. Using a non-parametric approach would mean having to analyse the entire distribution, which is a larger vector that has one entry for every realisation of intake.

We emphasize that the binomial assumption is in contrast with much of the DRO literature, in which distributions are usually non-parametric. The reason for this is that parametric distributions often lead to models that do not have convenient reformulations. However, in the context of our problem, we show that it is possible to derive algorithms which are both fast and near-optimal. Binomial and negative binomial distributions have often been used for demand modelling, particularly in inventory management. Examples of this include Collins (2004) for a risk-minimising newsvendor, Gallego et al. (2007) for inventory planning under highly uncertain demand, Dolgui and Pashkevich (2008) for forecasting demand in slow-moving inventory systems, and Rossi et al. (2014) for confidence-based newsvendor problems.

In this chapter, we will use the fact that every distribution in the ambiguity set is binomial in order to find the worst-case expected cost for a fixed pulling forward decision. In general, our methodology consists of three key steps. Firstly, we construct a discrete ambiguity set for the parameters of the true distribution. Secondly, we create a MILP reformulation of the model by replacing the inner objective with a finite number of constraints. In particular, there is one constraint for each distribution in the ambiguity set. Thirdly, we study the objective function as a function of the distribution's parameters in order to construct a set of extreme parameters.

For discrete distributions, the constraints representing the inner objective will always be linear. For continuous distributions, this is not necessarily the case. In such situations, for the second step, one would have to use a linear or quadratic approximation of the objective function. For example, this could be done using piecewise linear approximations or sample average approximations. Doing so would then allow our methodology to be applied.

2.1.2 Our Contributions

We consider a DRO model for a resource planning problem with an unknown number of intake jobs on each day. Using the problem structure, we model intakes as binomial random variables and study the resulting DRO model. Due to the use of the binomial distribution, the problem is considerably harder from a computational point of view. Our contributions in this chapter include the following:

1. A new framework for solving DRO problems with ambiguity sets containing

only distributions in the same parametric family as the nominal distribution. A comparison of this framework with a common, non-parametric framework based on the use of ϕ-divergences.

2. Three solver-based algorithms for the parametric model: an optimal and a heuristic cutting surface algorithm (named CS_opt and CS, respectively), and another heuristic algorithm named Approximate Objective (AO). These algorithms are described in Section 2.3.6. CS, while not exact, considerably simplifies the main bottleneck step of CS_opt: finding the worst-case distribution for a fixed pulling forward decision (referred to as *the distribution separation problem*). This makes it much more scalable with the size of the ambiguity set.

3. Extensive computational experiments on a variety of constructed instances which show the efficacy of our methods. See Section 2.5 for these results.

2.2 Literature Review

In this section, we review relevant literature relating to our problem and problems of a similar nature. In Section 2.2.1, we review the workforce planning literature and highlight the methodologies used there. In Section 2.2.2, we summarise the recent DRO literature and discuss how our research differs from it.

2.2.1 Workforce and Resource Planning

Workforce planning models of various forms have been studied in the OR literature since the mid 1950's, with early papers focusing on creating tractable deterministic

models (Holt et al., 1955; Hanssmann and Hess, 1960). Demand uncertainty has always been discussed in these early papers, with some authors extending previous models to minimise expected cost rather than cost (Fetter, 1961). In more recent literature, the modelling of uncertain demand has been developed further. The most common method in the literature has been *two-stage stochastic programming*. This methodology was applied to nurse scheduling (J. Abernathy et al., 1973) and recruitment for a military organisation (Martel and Price, 1978) in the early literature.

More recent examples of stochastic programming in workforce planning include planning a cyber branch of the US army (Bastian et al., 2020), and service industry workforce planning (Zhu and Sherali, 2009; Ross, 2016). These authors use stochastic programming due to their assumption that the distribution of the uncertain parameters is known. When this is not the case, or if the planner is risk-averse, *robust optimisation* (RO) can be used to represent demand uncertainty. This methodology has been used, for example, in healthcare (Holte and Mannino, 2013) and air traffic control (Hulst et al., 2017).

Recently, there have also been some applications of DRO to workforce and resource planning. Liao et al. (2013) used DRO for staffing a workforce to take calls arriving at a call centre at an uncertain rate. The reason for using DRO was cited as being that the true arrival rates of calls are usually subject to fluctuations, meaning that the typical stochastic model with a fixed Poisson distribution was not appropriate. They simulated the DRO solution and the stochastic programming solution and found that the two had similar costs. However, the stochastic programming solution violated

more model constraints. Chen et al. (2015) also used DRO for workforce planning in a hospital environment. In particular, they used DRO to determine bed requirements in order to appropriately manage admissions to the hospital. They use DRO due to the difficulty in specifying a distribution to describe patient movements in the hospital, and find that it performs better than a deterministic approach.

Our resource planning problem deals with the management of both planned and unplanned jobs. Similar problems exist in other settings, such as scheduling for gas pipeline maintenance (Angalakudati et al., 2014), and operating room scheduling in hospitals (Samudra et al., 2016). Particularly, in operating room planning, the workstack and intake jobs as defined in our model are similar to elective (inpatient and outpatient) and non-elective (emergency) surgeries. Similarly, in gas pipeline maintenance the workstack and intake jobs correspond to planned maintenance jobs and emergency gas leak repairs, respectively.

The main difference between our research and these papers is the choice of performance measure. For example, Angalakudati et al. (2014) use overtime hours as a performance measure under the assumption that jobs have individual completion times. However, since our model is for tactical and not operational planning, jobs and capacity are aggregated. The duration of each job is not modelled directly. Hence, in our case, the amount of overtime would be inferred by the number of jobs that could not be completed, i.e. the number that rolled over. As discussed by Samudra et al. (2016), metrics chosen for optimisation differ based on the underlying context and the stakeholders involved. They emphasize that traditional metrics such as makespan

do not work in presence of both planned and emergency demands. In our application, the time taken to complete jobs is not of particular concern. However, leaving jobs incomplete is very costly due to its effects on customer satisfaction. In industries like telecommunications, customer satisfaction is of great importance, and hence rollover may be the most appropriate performance measure.

The literature reviewed here shows that the modelling of uncertain demand in resource and workforce planning has been the subject of a breadth of research in the past. It suggests that the most common approach is to employ two-stage stochastic programming models. However, the assumption that the distribution of demand is known is not reasonable in our setting. We do assume, however, that we can take samples of intake in order to estimate the parameters of its distribution. In addition, there are no recourse actions in our problem. In such settings, RO and DRO are the only potential solution approaches. For our problem, a robust model will be shown to lead to more conservative decisions and large costs. We show this in Appendix A.2.2.

Hence, we present a DRO model for our problem, which will extend the previous stochastic programming approaches to the case where the distribution is not known exactly. We find that the model is large and complex, due to the size of the sets of intakes and distributions. Hence, we develop heuristics that apply dimension reduction to these sets in order to reduce solution times. One algorithm considers only a small subset of distributions, and the other operates on a small subset of the set of potential intake realisations. While these algorithms perform well on average, they do sacrifice optimality for speed in some large instances.

2.2.2 Distributionally Robust Optimisation

DRO combines concepts from robust optimisation and stochastic programming in order to protect the decision maker from distributional ambiguity. DRO models are constructed using only limited information on the true distribution of the uncertain parameters. This information is encoded in an *ambiguity set*: a set of distributions in which the true distribution should lie. The earliest type of ambiguity set in the literature is the *moment-based* ambiguity set. This set contains all distributions whose moments satisfy a given set of constraints. The simplest moment-based sets consider moments to be fixed and known. The moments concerned have often been the mean and variance. This case was studied by Scarf (1957) for a newsvendor model. Other papers included models where the first m moments were known (Shapiro and Kleywegt, 2002).

Authors have also developed models that did not assume that these values were fixed but that they were known to lie in an interval or that ordinal relationships between probabilities were known (Breton and Hachem, 1995). Other examples of this come from Ghaoui et al. (2003) and Lotfi and Zenios (2018), who study a CVaR model where the first two moments are only known to belong to polytopic or interval sets. Methodologies for solving moment-based ambiguity set models include reformulation via Cauchy-Schwarz bounds on the objective function (Scarf, 1957), reformulating as a second order conic program (Ghaoui et al., 2003; Lotfi and Zenios, 2018), sample average approximations (Shapiro and Kleywegt, 2002) and sub-gradient decomposition (Breton and Hachem, 1995).

The second common methodology for constructing ambiguity sets is using distance measures. A *distance-based* ambiguity set contains all distributions that lie within some pre-prescribed distance of a nominal one. In the literature, many ways to measure this distance have been studied. For example, many papers have used the Wasserstein distance. This distance can lead to reformulations as convex programs (Mohajerin Esfahani and Kuhn, 2018). Due to this, it has been used in a number of contexts, such as portfolio selection (Pflug and Wozabal, 2007), least squares problems (Mehrotra and Zhang, 2013) and statistical learning (Lee and Mehrotra, 2015; Lee and Raginsky, 2018).

Another common family of distance measures in DRO has been ϕ-*divergences*. This family contains a number of distance measures, such as the χ^2 distance, variation distance and Kullback-Leibler divergence. Such measures typically lead to second-order conic programming or even linear programming relaxations via taking the Lagrangian dual of the inner problem (Ben-Tal et al., 2013; Bayraksan and Love, 2015). Due to this, ϕ-divergences have been popular in the DRO literature. There have been numerous examples of ϕ-divergences being used to reformulate distributionally robust (DR) chance-constrained programs as chance-constrained programs (Hu et al., 2013; Yanıkoğlu and den Hertog, 2013; Jiang and Guan, 2016).

Another benefit of ϕ-divergences is that they can be used to create confidence sets and enforce probabilistic guarantees. Ben-Tal et al. (2013) show how to create confidence sets for the true distribution based on ϕ-divergences. This is done by taking a *maximum likelihood estimate* (MLE) of its parameters and using the resulting distribution

as the nominal distribution. Duchi et al. (2021) use DRO models with ϕ-divergence ambiguity sets to construct confidence intervals for the optimal values of a stochastic program with an ambiguous distribution. Their intervals asymptotically achieve exact coverage. By studying ϕ-divergence balls centred around the empirical distribution, Lam (2019) shows that DRO problems can recover the same standard of statistical guarantees as the central limit theorem.

In addition to these papers that consider general ϕ-divergence functions, the fact that ϕ-divergences cover a range of distance measures allows authors to select those that are most appropriate for their models. For example, Hanasusanto and Kuhn (2013) used χ^2-divergence ambiguity sets for a distributionally robust dynamic programming problem. They used the χ^2 divergence, in particular, because it allows the min-max problems in the dynamic programming recursion to be reformulated as second-order cone programs. They also chose this divergence because it does not *suppress* scenarios. In other words, it does not give scenarios zero probability in the worst-case if they have non-zero probability under the nominal distribution. The Kullback-Leibler divergence was also extensively studied by Hu and Hong (2013), who used it for DR chance-constrained problems. They showed that, under this divergence, if the nominal distribution was a member of the exponential family then so was the worst-case.

The literature we have reviewed so far concerns models that can be reformulated and solved exactly, due to their ambiguity sets being constructed using distance measures or moment constraints. However, there has also been significant literature studying

general DRO models that are not formulated in this way. In general, DRO models are *semi-infinite convex programs* (SCPs). They have a potentially infinite number of constraints induced by those defining the inner objective value. Typically, iterative algorithms are used to solve SCP models. For example, Kortanek and No (1993) developed a *cutting surface* (CS) algorithm for linear SCP problems with differentiable constraints. This algorithm approximates the infinite set of constraints with a sequence of finite sets of constraints. Constraints are iteratively added to the current set considered until stopping criteria are met. The constraint that is most violated by the current solution is added at each iteration.

In the context of DRO, adding a constraint corresponds to finding a distribution to add to the current ambiguity set. This is referred to as solving the *distribution separation problem*. Pflug and Wozabal (2007) later applied this algorithm to DRO models for portfolio selection under general ambiguity sets. In an extension of Kortanek and No (1993)'s algorithm, Mehrotra and Papp (2014) developed a CS algorithm for SCP problems that allowed for non-linear cuts, and did not require differentiable constraints. CS algorithms have since become a common approach to solving DRO problems that are computationally expensive and do not have convenient reformulations. For example, Rahimian et al. (2019) applied a CS algorithm to a DRO model using the total variation distance. They state that the model becomes expensive to solve to optimality when there are a large number of scenarios. Another example of its use in the literature is given by Bansal et al. (2018), who used a CS algorithm to solve DR knapsack and server location problems. Luo and Mehrotra (2019) also used

a CS algorithm to solve DRO models under the Wasserstein distance.

Our work differs from the cited literature in two key ways. Firstly, we consider demand distributions belonging to some parametric family, and enforce that the worst-case distribution also belongs to this family. We show that the resulting model can be reformulated as a large MILP. This model becomes slow to solve for large ambiguity and uncertainty sets. This is due to the large amounts of computation required and the large number of constraints. Hence, secondly, we present algorithms that make use of the additional distributional information in order to solve the parametric model. Among these algorithms is an optimal CS algorithm, that we will show to be fast for small problems, but to scale poorly with the size of the ambiguity set.

We also contribute a heuristic version of this CS algorithm, that solves the distribution separation problem at each iteration over a subset containing only the most extreme parameters. We will show that this allows us to greatly reduce the time taken to solve the distribution separation problem. We also show how to construct a confidence set for the worst-case parameter without the use of ϕ-divergences. In addition, we develop the non-parametric model and show how to reformulate it as a second-order conic program. We also compare the results from the parametric and non-parametric models to assess the benefit of incorporating the binomial information.

2.3 Planning Model

In this section, we introduce our planning model and discuss the different types of ambiguity sets that we will consider. In Section 2.3.1 we provide a summary of the

notation that will be used. Following this, in Section 2.3.2, we provide the DRO model itself under a general ambiguity set. Finally, we then detail the parametric and non-parametric versions of the model that will be studied in this chapter.

2.3.1 Notation and Definitions

We consider a planning horizon of T periods, which are days in our setting. The days in the plan are denoted by $t \in \{1, \ldots, T\}$. The inputs for the model are defined as follows. For each day t we have *capacity* c_t, which gives the number of jobs that we can complete on day t. The *workstack* for day t is the number of jobs that are currently due on day t, and is denoted D_t. The workstacks are known at the time of planning. The *intake* for day t is denoted I_t. This quantity is the number of jobs that will arrive between the time of planning and the due date t and will be due on day t. Each I_t is a random variable, and its value is not realised until the end of day t. In other words, workstack and intake jobs represent planned and unplanned/emergency jobs in the terminology used in other problems.

The *rollover* for day t is the number of jobs that are due on day t but are left incomplete at the end of day t. This quantity is denoted by R_t, which is a random variable due to its dependence on I_t. Each unit of rollover on day t incurs a cost a_t. The set of all potential realisations of the random variable I_t is denoted by $\mathcal{I}_t = \{0, \ldots, i_t^{\max}\}$, and a realisation of I_t is denoted by i_t. We use bold letters to represent the vectors of intakes, workstacks and so on. For example, the vector of workstacks is denoted by $\boldsymbol{D} = (D_1, \ldots, D_T)$. The set of all realisations of the vector $\boldsymbol{I}$ is denoted

by $\mathcal{I}$. We assume that the set $\mathcal{I}$ is the cartesian product of the marginal sets, i.e. $\mathcal{I} = \mathcal{I}_1 \times \ldots \times \mathcal{I}_T$. In the language of robust optimisation, $\mathcal{I}$ is referred to as an *uncertainty set* for $\boldsymbol{I}$. For a realisation $\boldsymbol{i}$ of the vector of intakes $\boldsymbol{I}$, the corresponding realisation of rollover is denoted by $\boldsymbol{R^i} = (R_1^{\boldsymbol{i}}, \ldots, R_T^{\boldsymbol{i}})$.

The objective of our problem is to minimise the worst-case expected rollover cost by *pulling forward* jobs. Hence, the decision vector in our problem is the pulling forward variable, which we denote by $\boldsymbol{y}$. Jobs can be completed no earlier than K periods prior to their due date. Therefore, we use y_{t_1,t_2} to denote the number of jobs pulled forward from period $t_1 \in \{2, \ldots, T\}$ to period $t_2 \in \{t_1 - K, \ldots, t_1 - 1\}$. This corresponds to completing y_{t_1,t_2} additional jobs on t_2 that are due on t_1.

2.3.2 General Distributionally Robust Model

We now consider the distributionally robust planning model, which is defined as follows. Denote by $\mathcal{P}$ a general *ambiguity set* of intake distributions, such that every distribution $\boldsymbol{P} \in \mathcal{P}$ assigns a probability to every possible intake $\boldsymbol{i} \in \mathcal{I}$. Our model aims to minimise the worst-case expected rollover cost by selecting the value of $\boldsymbol{y}$. The model is shown in (2.3.1)-(2.3.8).

$$\min_{\boldsymbol{y},\boldsymbol{R}} \max_{\boldsymbol{P}\in\mathcal{P}} \sum_{t=1}^{T} a_t \mathbb{E}_{\boldsymbol{P}}(R_t) \tag{2.3.1}$$

$$\text{s.t.} \sum_{t_2=\max\{t_1-K,1\}}^{t_1-1} y_{t_1,t_2} \leq D_{t_1} \;\forall\; t_1 = 2, \ldots, T, \tag{2.3.2}$$

$$\sum_{t_1=t_2+1}^{\min\{t_2+K,T\}} y_{t_1,t_2} \leq \max\{c_{t_2} - D_{t_2}, 0\} \;\forall\; t_2 = 1, \ldots, T-1, \tag{2.3.3}$$

$$R_1^{\boldsymbol{i}} \geq i_1 + \sum_{t_1=2}^{\min\{1+K,T\}} y_{t_1,1} - (c_1 - D_1) \;\forall\, \boldsymbol{i} \in \mathcal{I}, \tag{2.3.4}$$

$$R_t^{\boldsymbol{i}} \geq R_{t-1}^{\boldsymbol{i}} + i_t + \sum_{t_1=t+1}^{\min\{t+K,T\}} y_{t_1,t} - \left(c_t - D_t + \sum_{t_2=\max\{t-K,1\}}^{t-1} y_{t,t_2} \right)$$

$$\forall\, t = 2, \ldots, T-1 \;\forall\, \boldsymbol{i} \in \mathcal{I}, \tag{2.3.5}$$

$$R_T^{\boldsymbol{i}} \geq R_{T-1}^{\boldsymbol{i}} + i_T - \left(c_T - D_T + \sum_{t_2=\max\{T-K,1\}}^{T-1} y_{t,t_2} \right) \;\forall\, \boldsymbol{i} \in \mathcal{I}, \tag{2.3.6}$$

$$y_{t_1,t_2} \in \mathbb{N}_0 \;\forall\, t_1 = 2, \ldots, T \;\forall\, t_2 = \max\{t_1 - K, 1\}, \ldots, t_1 - 1 \tag{2.3.7}$$

$$R_t^{\boldsymbol{i}} \geq 0 \;\forall\, t = 1, \ldots, T \;\forall\, \boldsymbol{i} \in \mathcal{I}. \tag{2.3.8}$$

The general idea in calculating rollover in the T-day model is as follows. For a given day t, we first compute the number of jobs to be completed on day t. To compute this, we take the rollover from day $t-1$ and day t's intake as a baseline number of jobs. Then we add the number of jobs pulled forward to day t, i.e. $\sum_{t_1=t+1}^{\min\{t+K,T\}} y_{t_1,t}$. We then compute the capacity that can be used to complete these jobs. This is done by taking the capacity c_t and subtracting the capacity required to complete those workstack jobs that are not pulled forward from day t, i.e. $D_t - \sum_{t_2=\max\{t-K,1\}}^{t-1} y_{t,t_2}$. If the remaining capacity is enough to complete all jobs on t, then the rollover is zero. Otherwise, the rollover is the number of jobs left incomplete.

Constraints (2.3.2) and (2.3.3) provide upper bounds on the pulling forward totals. Constraint (2.3.2) ensures that no jobs are pulled forward if they cannot be completed on the day to which they are moved. Constraint (2.3.3) ensures that only workstack jobs can be pulled forward, and that a job cannot be pulled forward multiple times in order to be pulled forward more than K days. Constraint (2.3.4) reflects that

jobs cannot be pulled forward from day 1 and hence we only subtract those jobs pulled forward to day 1 from its remaining capacity. We do not reduce rollover by pulling forward from it. Similarly, constraint (2.3.6) reflects that jobs cannot be pulled forward to the final day of the plan. Hence, we only pull forward from this day and not to this day. For every other day, constraint (2.3.5) captures that we can pull forward to *and* from said day. We therefore add *and* subtract jobs from its capacity to calculate the rollover.

2.3.3 Non-parametric DRO Model

The non-parametric model is defined by ambiguity sets $\mathcal{P}$ containing distributions $\boldsymbol{P}$ that are not necessarily parametric. To be specific, $\mathcal{P}$ can be any subset of the set of all distributions over the set of intakes, i.e. $\mathcal{P} \subseteq \left\{ \boldsymbol{P} \in [0,1]^{|\mathcal{I}|} : \sum_{j=1}^{|\mathcal{I}|} P_j = 1 \right\}$.

Phi-divergence Based Ambiguity Sets

As discussed earlier in the chapter, it is common to define $\mathcal{P}$ using ϕ-divergences. Adopting similar notation to that of Bayraksan and Love (2015), suppose that $\boldsymbol{P}$ and $\boldsymbol{Q}$ are two probability distributions. We define a ϕ-divergence d_ϕ for ϕ-divergence function ϕ as:

$$d_\phi(\boldsymbol{P}, \boldsymbol{Q}) = \sum_{j=1}^{n} Q_j \phi\left(\frac{P_j}{Q_j}\right),$$

where ϕ is a convex function on the non-negative reals. This function measures the distance between $\boldsymbol{P}$ and $\boldsymbol{Q}$. In what follows, $\boldsymbol{Q}$ will be treated as a nominal distribution. Furthermore, we denote by ϕ^* the *conjugate* of ϕ, which can be found

via (2.3.9).

$$\phi^*(s) = \sup_{\tau \geq 0} \{s\tau - \phi(\tau)\} \tag{2.3.9}$$

The conjugate will be useful when finding reformulations later in the chapter. Given a nominal distribution $\boldsymbol{Q}$, we can define $\mathcal{P}$ as the set of all distributions $\boldsymbol{P}$ that lie within some pre-prescribed distance from $\boldsymbol{Q}$ as measured by the ϕ-divergence. In other words, we can use:

$$\mathcal{P}_\kappa = \left\{ \boldsymbol{P} \in [0,1]^{|\mathcal{I}|} : \sum_{j=1}^{|\mathcal{I}|} P_j = 1, d_\phi(\boldsymbol{P}, \boldsymbol{Q}) \leq \kappa \right\}. \tag{2.3.10}$$

As described by Ben-Tal et al. (2013), this formulation of the ambiguity set allows us to choose κ such that $\mathcal{P}$ is a confidence set for the true distribution. Suppose that the *true distribution* $\boldsymbol{P}^0$ lies in a parameterised set $\{\boldsymbol{P}^{\boldsymbol{\theta}} \mid \boldsymbol{\theta} \in \Theta\}$, such that the true value of $\boldsymbol{\theta}$ is $\boldsymbol{\theta}^0$. Also suppose that we take N samples of intake from $\boldsymbol{P}^0$ and take an MLE $\hat{\boldsymbol{\theta}}$ of $\boldsymbol{\theta}^0$. Then, if we choose κ using (2.3.11), the set $\mathcal{P}_\kappa$ is an approximate $100(1-\alpha)\%$ confidence set for $\boldsymbol{P}^0$ around $\hat{\boldsymbol{P}} = \boldsymbol{P}^{\hat{\boldsymbol{\theta}}}$.

$$\kappa = \frac{\phi''(1)}{2N} \chi^2_{o,1-\alpha}. \tag{2.3.11}$$

In (2.3.11), o is the dimension of Θ and $\chi^2_{o,1-\alpha}$ is the $100(1-\alpha)^{\text{th}}$ percentile of the χ^2 distribution with o degrees of freedom.

Reformulation with Modified χ^2-divergence

There are many choices for the choice of ϕ-divergence function, and some examples can be found in the paper by Ben-Tal et al. (2013). In our model, we will use the *modified χ^2-divergence* as our ϕ-divergence. This uses the ϕ-divergence function $\phi_{m\chi^2}(\tau) =$

$(\tau - 1)^2$ and is defined in (2.3.12).

$$d_{\phi_{m\chi^2}}(\boldsymbol{P}, \boldsymbol{Q}) = \sum_{j=1}^{n} \frac{(P_j - Q_j)^2}{Q_j}. \tag{2.3.12}$$

Here, n is the number of potential values of the uncertain parameters. In our problem, we have $n = |\mathcal{I}|$. We choose this function for the following reasons. Firstly, it leads to a *convex quadratic programming* (CQP) reformulation. Secondly, squared deviations from the nominal distribution are represented as a proportion of the nominal distribution's value. This means that small deviations from the nominal distribution can still lead to a large term in the sum in (2.3.12). When n is large, most values of Q_j will be small, and this will help identify significant deviations from small nominal values. Other choices of ϕ-divergences that lead to CQP reformulations, such as the χ^2-divergence, Hellinger distance and the Cressie-Read distance, do not have the normalisation effect given by dividing each term by Q_j.

Following Ben-Tal et al. (2013), we find the following CQP reformulation of our full model with $\mathcal{P} = \mathcal{P}_\kappa$:

$$\min_{\boldsymbol{y}, \boldsymbol{R}, \eta, \nu, \boldsymbol{\zeta}, \boldsymbol{u}} \left\{ \eta(\kappa - 1) + \nu + \frac{1}{4} \sum_{j=1}^{n} Q_j u_j \right\}$$

$$\text{s.t. } (2.3.2) - (2.3.8),$$

$$\sqrt{4\zeta_j^2 + (\eta - u_j)^2} \leq (\eta + u_j) \ \forall \ j = 1, \ldots, n$$

$$\zeta_j \geq \sum_{t=1}^{T} a_t R_t^{\boldsymbol{i}^j} - \nu + 2\eta \ \forall \ j = 1, \ldots, n$$

$$\zeta_j \geq 0 \ \forall \ j = 1, \ldots, n.$$

$$\eta \geq 0.$$

Note that this model is a second-order cone program. In this formulation, ζ_j and u_j for $j = 1, \ldots, n$ are dummy variables defined to ensure that the model is a CQP model. A full derivation of this reformulation can be found in Appendix A.1, along with how to extract the worst-case distribution from its solution.

2.3.4 Parametric DRO Model

In this section, we detail a parametric version of the DRO planning model. This is a new modelling framework for DRO problems that allows the ambiguity set to contain only distributions that are members of the same parametric family as the true distribution. This is useful in cases where we know beforehand which family the true distribution lies in, because it ensures that the worst-case distribution implied by the model is also in this family.

Implications of Parametric Ambiguity Sets

Recall from Section 2.3.3 that we can use ϕ-divergences to create confidence sets when we know that the true distribution lies in some parametric family $\mathcal{P}_\Theta = \{\boldsymbol{P}^{\boldsymbol{\theta}} \mid \boldsymbol{\theta} \in \Theta\}$. The resulting confidence set (2.3.10), however, does not only contain distributions in this family. Therefore, there is no guarantee that the worst-case distribution will lie in this family and hence no guarantee that it is even a distribution that could be equal to $\boldsymbol{P}^0$. Our methodology involves explicitly using the set $\mathcal{P}_\Theta$ in our DRO model instead, which eliminates potential worst-case distributions that are not in the same family as the true distribution. Suppose that we take the ambiguity set given by $\mathcal{P} = \mathcal{P}_\Theta$.

The methodology in Section 2.3.3 relies on being able to represent the requirement that $\boldsymbol{P} \in \mathcal{P}$ in the constraints of the model. However, representing $\boldsymbol{P} \in \mathcal{P}_\Theta$ in the constraints is more challenging. In the case where $\mathcal{P}_\Theta$ represents a set of discrete parametric distributions, e.g. binomial or Poisson, the requirement might be represented by:

$$P_j = f_{\boldsymbol{I}}(\boldsymbol{i}^j \mid \boldsymbol{\theta}) \text{ for some } \boldsymbol{\theta} \in \Theta,$$

where $f_{\boldsymbol{I}}$ is the probability mass function (PMF) of $\boldsymbol{I}$ and $\boldsymbol{i}^j$ is the j^{th} realisation of intake. The only reasonable way that one might attempt to include this in the model is to treat $\boldsymbol{\theta}$ as a dummy variable, and replace P_j in the objective with $f_{\boldsymbol{I}}(\boldsymbol{i}^j \mid \boldsymbol{\theta})$. However, most PMFs as functions of their parameters are either high order polynomials (such as binomial) or include exponential functions (such as Poisson). Including them in the model through the objective function will hence make the model un-solvable. As an example, consider our model with independent intakes and $I_t \sim \text{Bin}(i_t^{\max}, p_t^0)$ for unknown p_t^0 for $t = 1, \ldots, T$. The objective of the inner problem becomes:

$$\max_{\boldsymbol{p} \in \Theta} \sum_{t=1}^{T} \sum_{\boldsymbol{i} \in \mathcal{I}} a_t R_t^{\boldsymbol{i}} \prod_{l=1}^{T} \binom{i_l^{\max}}{i_l} p_l^{i_l} (1 - p_l)^{i_l^{\max} - i_l}. \tag{2.3.13}$$

Treating this as a non-linear program, we might consider solving using the Karush-Kuhn-Tucker (KKT) conditions. The derivative of the objective function in (2.3.13) with respect to $p_{t'}$ is:

$$\sum_{t,i} a_t R_t^{\boldsymbol{i}} \binom{i_{t'}^{\max}}{i_{t'}} \left(i_{t'} p_{t'}^{i_{t'}-1} (1 - p_{t'})^{i_{t'}^{\max} - i_{t'}} - p_{t'}^{i_{t'}} (i_{t'}^{\max} - i_{t'}) (1 - p_{t'})^{i_{t'}^{\max} - i_{t'} - 1} \right) \prod_{l \neq t'} f_{I_l}(i_l),$$

for each $t' \in \{1, \ldots, T\}$, where f_{I_l} is the PMF of I_l. Choosing a vector $\boldsymbol{p}$ such that $p_t < 1$ for all t and all derivatives are equal to zero is a challenging task. This would

need to be done numerically, and hence would not result in a convenient objective function for our outer model. Furthermore, using a ϕ-divergence to define Θ would not result in a convenient reformulation. This would involve using an ambiguity set for $\boldsymbol{p}^0$ of the form:

$$\Theta = \{\boldsymbol{p} \in [0,1]^T : d_\phi(\boldsymbol{p}, \boldsymbol{q}) \leq \kappa\},$$

where $\boldsymbol{q}$ is the success probability vector corresponding to the nominal distribution $\boldsymbol{Q}$. Now consider the methodology in Section 2.3.3. This methodology relies on the objective function being separable over j (see Appendix A.1). Following the same steps but with the objective in (2.3.13), we arrive at the following dual objective:

$$\min_{\boldsymbol{\xi} \geq 0} \left\{ \xi_0 \kappa + \max_{\boldsymbol{p} \geq 0} \sum_{t=1}^{T} \left(\sum_{\boldsymbol{i} \in \mathcal{I}} a_t R_t^i \prod_{l=1}^{T} f_{I_l}(i_l) - \xi_0 q_t \phi\left(\frac{p_t}{q_t}\right) + \xi_t (1 - p_t) \right) \right\},$$

with $\boldsymbol{\xi} = (\xi_0, \xi_1, \ldots, \xi_T)$. Due to the product over l inside the $\max_{\boldsymbol{p} \geq 0}$ operator (which contains each success probability), we see that this objective is not separable over t. Thus, the remaining steps in creating a convenient reformulation cannot be carried out. This holds not only for independent distributions, but for any distribution where the PMF of $\boldsymbol{I}$ depends on more than one p_t.

Hence, our methodology is as follows. Instead of treating the parameter $\boldsymbol{\theta}$ as a vector of decision variables, we represent it using a discrete, finite set of potential values. In other words, we assume that Θ is a discrete and finite set. This allows us to represent the distributional ambiguity via a finite set of constraints that are linear in the rollover variables. The resulting model has one additional constraint for every $\boldsymbol{\theta} \in \Theta$, but remains a mixed integer linear program (MILP). We detail the MILP reformulation of the parametric model in Section 2.3.4.

Mixed Integer Linear Programming Reformulation

To solve this model, we can reformulate it as an MILP as follows. Firstly, we replace the set $\mathcal{P}_\Theta$ with Θ and optimise over the parameters $\boldsymbol{\theta}$ directly. Since there is a one-to-one mapping between $\boldsymbol{\theta}$ and $\boldsymbol{P}^{\boldsymbol{\theta}}$, the objective becomes:

$$\min_{\boldsymbol{y},\boldsymbol{R}} \max_{\boldsymbol{\theta}\in\Theta} \sum_{t=1}^{T} a_t \mathbb{E}_{\boldsymbol{\theta}}\left(R_t\right).$$

Next, we define a dummy variable ϑ to represent the worst-case expected cost for a given $\boldsymbol{y}$. Since the set Θ is a discrete set, we can enforce the requirement that $\vartheta = \max_{\boldsymbol{\theta}\in\Theta} \sum_{t=1}^{T} a_t \mathbb{E}_{\boldsymbol{\theta}}(R_t)$ using a set of linear constraints. Hence, the MILP reformulation of the DRO model is given by:

$$\begin{aligned} \min_{\boldsymbol{y},\boldsymbol{R},\vartheta}\ & \vartheta \\ \text{s.t. } & (2.3.2) - (2.3.8), \\ & \vartheta \geq \sum_{t=1}^{T} a_t \mathbb{E}_{\boldsymbol{\theta}}(R_t)\ \forall\ \boldsymbol{\theta} \in \Theta. \end{aligned}$$

This model can be very slow to build and solve. This is mostly due to the amount of computation required to build the model and its constraints. The constraint for ϑ requires us to compute the distribution $\boldsymbol{P}^{\boldsymbol{\theta}}$ for every $\boldsymbol{\theta} \in \Theta$. Due to the sizes of Θ and $\mathcal{I}$, this can be very slow. To see this, consider an example with $|\Theta| = 3883$ distributions and $|\mathcal{I}| = 20000$ potential intakes. Suppose also that the intakes are independent. Then, for each of 3883 distributions we would need to compute a product of T PMF values, for each of 20000 intakes. This means computing $T \times 3883 \times 20000 = T \times (77.66 \times 10^6)$ PMF values. Furthermore, the model has $T|\mathcal{I}|$ rollover variables and constraints, and $|\Theta|$ expected value constraints. This also makes the model slow

to build and solve for large instances. For this instance with $T = 5$, this corresponds to 103,878 additional constraints, when compared with the deterministic model. Our heuristics therefore employ dimension reduction techniques to make them faster to solve.

2.3.5 Binomial Intakes and Ambiguity Sets

As previously discussed, we will assume that the intakes in our problem are independent binomial random variables. In other words, we assume that $I_t \sim \text{Bin}(i_t^{\max}, p_t^0)$ for unknown p_t^0, for $t = 1, \ldots, T$. We assume that $\mathcal{I}$ is provided to us prior to model building, either by a prediction model or expert knowledge. The true set in which we know that $\boldsymbol{p}^0$ must lie is $\Theta_{\text{base}} = [0, 1]^T$. As detailed in Section 2.3.4, we will however use a finite, discrete subset of Θ_{base} as an ambiguity set for our model. We consider a discretisation Θ'_{base} of Θ_{base} of the form given in (2.3.14), where M is chosen by the planner, and details the fineness of the discretisation.

$$\Theta'_{\text{base}} = \left\{ \frac{m}{M} \;\middle|\; m = 0, \ldots, M \right\}^T \tag{2.3.14}$$

We assume that we have access to N samples of past intake data, from which we can take an MLE $\hat{\boldsymbol{p}}$ of $\boldsymbol{p}^0$. The corresponding distribution is given by $\hat{\boldsymbol{P}}$, which has mean vector $\hat{\boldsymbol{i}} = \hat{\boldsymbol{P}}\boldsymbol{i}^{\max}$. Given the MLE $\hat{\boldsymbol{p}}$, we consider only $\boldsymbol{p} \in \Theta'_{\text{base}}$ that can be considered *close* to $\hat{\boldsymbol{p}}$. It is common in the non-parametric DRO literature to use ϕ-divergences to measure the distance between two distributions. The main reason for this is that it results in convenient reformulations via dualising the inner problem. However, since our approach does not entail dualising the inner problem, this benefit

does not apply to us. Another reason for using ϕ-divergences is that they allow us to create confidence sets for the true distribution. However, this is based on applying the ϕ-divergence to the distributions themselves, not to the parameters.

We could construct a confidence set for $\boldsymbol{p}^0$ by first constructing a confidence set for $\boldsymbol{P}^0$ and then creating Θ by extracting the parameters of each distribution in the confidence set. However, this would entail computing the corresponding distribution for every $\boldsymbol{p} \in \Theta'_{\text{base}}$, which is a large computational task. Hence, we do not use ϕ-divergences for the parametric model. We can, however, construct a confidence set for $\boldsymbol{p}^0$ without using ϕ-divergences and without needing to compute each distribution $\boldsymbol{P}^{\boldsymbol{p}}$. Since $\hat{\boldsymbol{p}}$ is an MLE of $\boldsymbol{p}^0$ based on N samples from the true intake distribution, by Millar (2011), for large N we have:

$$(\hat{p}_t - p_t^0) \sim \mathcal{N}\left(0, \frac{\hat{p}_t(1-\hat{p}_t)}{N i_t^{\max}}\right),$$

approximately. Therefore, by independence of the T different MLE's, we have that:

$$\sum_{t=1}^{T} \frac{N i_t^{\max}}{\hat{p}_t(1-\hat{p}_t)}(\hat{p}_t - p_t^0)^2 \sim \chi^2_T,$$

approximately. Hence, we have the following approximate $100(1-\alpha)\%$ confidence set for $\boldsymbol{p}^0$ around $\hat{\boldsymbol{p}}$:

$$\Theta_\alpha = \left\{ \boldsymbol{p} \in \Theta_{\text{base}} : \sum_{t=1}^{T} N i_t^{\max} \frac{(\hat{p}_t - p_t)^2}{\hat{p}_t(1-\hat{p}_t)} \leq \chi^2_{T,1-\alpha} \right\}. \tag{2.3.15}$$

We then create a discretisation of this set for use in our model as follows:

$$\Theta'_\alpha = \Theta_\alpha \cap \Theta'_{\text{base}}. \tag{2.3.16}$$

Note that (2.3.15) may yield a different approximate confidence set to the one obtained

using the ϕ-divergence method. This is because they are not the same set, but two different approximations of the same set.

2.3.6 Solver-based Solution Algorithms

Suppose that Θ is a discrete set of parameters. As described in Section 2.3.4, the model can be solved to optimality by reformulating it as an MILP. However, when Θ and $\mathcal{I}$ are large, this model has a large number of constraints and decision variables. This can make it very slow to solve. Hence, we develop three dimension reduction algorithms in order to reduce the effects of the sizes of these sets on solution times. In this section, we present 3 algorithms for solving the parameteric model. Firstly, we discuss two CS algorithms. The first is an optimal CS algorithm that also scales poorly with the size of Θ. The second is a heuristic CS algorithm that applies dimension reduction to Θ. Following this, we describe our final algorithm, Approximate Objective (AO), that applies dimension reduction to $\mathcal{I}$.

Cutting Surface Algorithms

In this section, we describe our adaptation of the CS algorithm detailed in the literature review, which has been commonly used in the DRO literature. The algorithm has a number of different forms, but the one that we base our adaptation on is that from the review paper by Rahimian and Mehrotra (2019). The general idea of the algorithm is as follows. In order to deal with the large number of constraints implied by the ambiguity set, the algorithm uses the following steps. We start with a singleton set containing one distribution, and solve the problem over this ambiguity set.

Then, for the generated pulling forward solution, we find the worst-case distribution over the entire ambiguity set. We then add this distribution to the current subset of the ambiguity set and then repeat the above steps. This procedure repeats until optimality criteria are met.

In more detail, suppose that we have some initial subset $\Theta^1 = \{\boldsymbol{p}^{\text{init}}\}$ of our set of distributions Θ and we solve the full model with ambiguity set Θ^1, to get an optimal decision $\boldsymbol{y}^1$. Then, we find the worst-case parameter, $\boldsymbol{p}^1 \in \Theta$, for the fixed solution $\boldsymbol{y}^1$, and add it to our set to create a new subset $\Theta^2 = \Theta^1 \cup \{\boldsymbol{p}^1\}$ of Θ. We then solve the model with ambiguity set Θ^2, and repeat. We stop the algorithm when we have reached ε-optimality, i.e. if the solution from the full problem at iteration k, $\boldsymbol{y}^k$, gives a worst-case expected cost over Θ^k that is within $\varepsilon/2$ of the worst-case expected cost for $\boldsymbol{y}^k$ over Θ. The algorithm returns an ε-optimal solution to the DRO model in a finite number of iterations. The issue with this version of CS is that, even if $\boldsymbol{y}$ is fixed at $\boldsymbol{y}^k$, finding the true worst-case distribution $\boldsymbol{p}^k$ can be a cumbersome task. In our case, we can simply enumerate all distributions in Θ. Even though this is not a difficult task, it requires a significant amount of computation due to the necessity of calculating the distributions themselves.

From now on, we refer to the optimal CS algorithm described above as CS_opt. We will show that this algorithm suffers from poor scaling with respect to the size of Θ. In order to reduce the computational burden, we apply dimension reduction to Θ. Particularly, we use the simple observation that $\mathbb{E}_{\boldsymbol{p}}(I_t) = i_t^{\max} p_t$ is increasing in p_t to construct a set of *extreme* parameters. Intuitively, this result suggests that a higher

success probability also leads to no-less expected rollover, due to the fact that $R_t^{\boldsymbol{i}}$ is increasing in i_t. Hence, we construct a set of probability vectors such that at least one value is maximised. If this is not the case, then one value can be increased and this would cause higher expected rollover for that day. Furthermore, we also assume that the total success probability is maximised given that one value is maximised. This is to ensure that we take the most extreme probability vectors over all those such that one success probability is maximised. Mathematically, we define the set of extreme parameters as follows. Define $p_t^{\max} = \max_{\boldsymbol{p}\in\Theta} p_t$ for $t = 1, \ldots, T$ and find the set of parameters such that one value is maximised:

$$\Theta_t^{\max} = \{\boldsymbol{p} \in \Theta : p_t = p_t^{\max}\} \text{ for } t = 1, \ldots, T.$$

For each t, construct a set of the most extreme parameters in $\Theta_t^{\max}$ and take the union of these sets to form Θ^{ext}:

$$\Theta_t^{\text{ext}} = \underset{\boldsymbol{p}\in\Theta_t^{\max}}{\operatorname{argmax}} \left\{ \sum_{l=1}^{T} p_l \right\}, \quad \Theta^{\text{ext}} = \bigcup_{t=1}^{T} \Theta_t^{\text{ext}}.$$

In order to reduce the computation required, our heuristic CS algorithm (referred to as CS) solves the distribution separation problem over Θ^{ext}, rather than the entire ambiguity set Θ. The general framework for both of our CS algorithms is given below, where CS_opt uses $\tilde{\Theta} = \Theta$ in step 2(b) and CS uses $\tilde{\Theta} = \Theta^{\text{ext}}$.

1. Initialise $\Theta^1 = \{\boldsymbol{p}^{\text{init}}\}$, where $\boldsymbol{p}^{\text{init}} = \hat{\boldsymbol{p}}$ for example.

2. For $k = 1, \ldots, k^{\max}$:

   (a) Solve the model to optimality using ambiguity set Θ^k to generate solution $(\boldsymbol{y}^k, \vartheta^k)$ where ϑ^k is worst-case expected cost of $\boldsymbol{y}^k$ over the set Θ^k passed

to the model.

(b) Solve distribution separation problem $\max_{\boldsymbol{p}\in\tilde{\Theta}} \sum_{t=1}^{T} a_t \mathbb{E}_{\boldsymbol{p}}(R_t \mid \boldsymbol{y} = \boldsymbol{y}^k)$ to get solution $\boldsymbol{p}^k$:

i. For $\boldsymbol{p} \in \tilde{\Theta}$, calculate $C_{\boldsymbol{p}} = \sum_{t=1}^{T} a_t \mathbb{E}_{\boldsymbol{p}}(R_t \mid \boldsymbol{y} = \boldsymbol{y}^k)$.

ii. Choose $\boldsymbol{p}^k$ such that $C_{\boldsymbol{p}^k} = \max_{\boldsymbol{p}\in\tilde{\Theta}}(C_{\boldsymbol{p}})$.

(c) If $C_{\boldsymbol{p}^k} \leq \vartheta^k + \frac{\varepsilon}{2}$ or $\boldsymbol{p}^k \in \Theta^k$ then stop and return solution $(\boldsymbol{y}^k, \boldsymbol{p}^k)$.

(d) Else, set $\Theta^{k+1} = \Theta^k \cup \{\boldsymbol{p}^k\}$ and $k = k + 1$.

The logic behind 2(c), where we check if $\boldsymbol{p}^k \in \Theta^k$, is that calculation differences might cause ϑ^k and $C_{\boldsymbol{p}^k}$ to differ by more than $\frac{\varepsilon}{2}$ when they should be equal. Solvers use some dimension reduction techniques when building and solving their models. This can lead to objective values that are not the same as the ones given by the function used in 2(b), even for the same arguments. This stopping criterion is also used in the CS algorithms by Pflug and Wozabal (2007) and Bansal et al. (2018).

We now explain why this condition cannot cause early stopping. Firstly, assume that $\hat{\boldsymbol{p}}$ is not a worst-case parameter for $\boldsymbol{y}^k$ in Θ^k, i.e. it did not give a cost of ϑ^k. Since $\boldsymbol{p}^k$ is generated by the distribution separation problem, it is a worst-case parameter for $\boldsymbol{y}^k$ over the entire set $\tilde{\Theta}$. If we also have $\boldsymbol{p}^k \in \Theta^k$ then we have the following two facts. Firstly, we have $\Theta^k \setminus \{\hat{\boldsymbol{p}}\} \subseteq \tilde{\Theta}$ and so $\boldsymbol{p}^k$ is necessarily worse than every $\boldsymbol{p} \in \Theta^k \setminus \{\hat{\boldsymbol{p}}\}$. Secondly, $\boldsymbol{p}^k$ must be worse than $\hat{\boldsymbol{p}}$, because otherwise $\hat{\boldsymbol{p}}$ would be a worst-case parameter in Θ^k. Hence, $\boldsymbol{p}^k$ is a worst-case parameter in Θ^k, i.e. $C_{\boldsymbol{p}^k} = \vartheta^k < \vartheta^k + \frac{\varepsilon}{2}$. Now suppose that $\hat{\boldsymbol{p}}$ is a worst-case parameter in Θ^k. If $\boldsymbol{p}^k \in \Theta^k$

then we must have $C_{\boldsymbol{p}^k} \leq \vartheta^k < \vartheta^k + \frac{\varepsilon}{2}$ since $\hat{\boldsymbol{p}}$ is worse than $\boldsymbol{p}^k$. Hence, whenever $\boldsymbol{p}^k \in \Theta^k$ occurs, the first stopping criterion should also be met.

Approximate Objective Algorithm

The final algorithm that we describe is named *approximate objective* (AO). When solving the model to optimality, we are required to compute the distribution $\boldsymbol{P}^{\boldsymbol{p}}$ for each $\boldsymbol{p} \in \Theta$. For each intake $\boldsymbol{i}^j \in \mathcal{I}$ we can easily compute:

$$\max_{\boldsymbol{p} \in \Theta} P_j^{\boldsymbol{p}} = \max_{\boldsymbol{p} \in \Theta} \mathbb{P}(\boldsymbol{I} = \boldsymbol{i}^j \mid \boldsymbol{p}),$$

and then we can consider a new set of intakes in the model defined by:

$$\tilde{\mathcal{I}} = \left\{ \boldsymbol{i} \in \mathcal{I} : \max_{\boldsymbol{p} \in \Theta} P_j^{\boldsymbol{p}} > \beta \right\}$$

where β is our minimum intake probability. By tuning β, we are removing intakes from our set that are very unlikely. When solving the model, we are approximating the expected value by removing some small terms. Since the intakes removed have low probability, this approximation should be strong. To generate a solution to the full model, we simply solve the MILP reformulation with the full set Θ of parameters but over the reduced set $\tilde{\mathcal{I}}$ of intakes. For this chapter, we use $\beta = 10^{-3}$ as our initial testing showed that this value led to good improvements in computation time.

2.3.7 Example: A Two-day Problem

In order to illustrate the logic behind our algorithms, we now give an example of their use for a two-day version of our model. Since there is only one feasible pair of days that we can pull forward jobs between, i.e. $(2, 1)$, there is now only one decision

variable. We refer to this decision variable as $y = y_{2,1}$. The two-day model is given by (2.3.1)-(2.3.8) with $T = 2$ and $K = 1$.

Suppose that we have $\boldsymbol{c} = (30, 10)$, $\boldsymbol{D} = (5, 20)$, $\boldsymbol{i}^{\max} = (20, 20)$ and $a = (1, 1)$. This gives $|\mathcal{I}| = 21^2 = 441$. We construct a discretisation of a 99.5% confidence set for $\boldsymbol{p}^0$ using $\alpha = 0.005$, $N = 10$ and $M = 100$. This gives $|\Theta'_\alpha| = 305$, and we find that the maximum values of p_1 and p_2 are both 0.84. This suggests that the above model has $2 \times 441 = 882$ rollover constraints and variables, 81 expected value constraints and 2 pulling forward constraints. Hence, it has 1189 constraints and 884 decision variables. We solve this model to optimality in 2.6 seconds, to find the optimal y to be $y^{\mathrm{P}} = 9$ and the worst-case $\boldsymbol{p}$ to be $\boldsymbol{p}^{\mathrm{P}} = (0.82, 0.82)$ with an expected cost of $z^{\mathrm{P}} = 19.2$.

When we solve this model with CS, we find that $\Theta^{\mathrm{ext}} = \{(0.84, 0.79), (0.79, 0.84)\}$ and so CS only has to compute 2 PMFs as opposed to P and AO which have to compute 81. We initialise with $\Theta^1 = \{\hat{\boldsymbol{p}}\} = \{(0.75, 0.75)\}$. In iteration 1, CS solves the MILP reformulation over Θ^1 and finds $y^1 = 10$. It then evaluates the expected costs under each $\boldsymbol{p} \in \Theta^{\mathrm{ext}}$ and finds the worst-case to be given by $\boldsymbol{p}^1 = (0.84, 0.79)$. Hence, we have $\Theta^2 = \{(0.75, 0.75), (0.84, 0.79)\}$. In iteration 2, CS solves the model over Θ^2 and finds $y^2 = 8$. It finds the worst-case cost to be given by $\boldsymbol{p}^2 = (0.79, 0.84)$, and hence takes $\Theta^3 = \Theta^2 \cup \{(0.79, 0.84)\}$. In iteration 3, CS finds $y^3 = 9$ and $\boldsymbol{p}^3 = (0.84, 0.79)$. Since $(0.84, 0.79) \in \Theta^3$, the algorithm ends and returns $y^{\mathrm{CS}} = 9$ and $\boldsymbol{p}^{\mathrm{CS}} = (0.84, 0.79)$ with an expected cost of $z^{\mathrm{CS}} = 19.07$. Hence, CS returned the optimal y but slightly underestimated its worst-case cost. This is an example of where CS will be suboptimal because $\boldsymbol{p}^{\mathrm{P}} \notin \Theta^{\mathrm{ext}}$. However, CS returned its solution in 0.17 seconds, as opposed to

P's 2.6 seconds. Note that CS terminated in 2 iterations because $|\Theta^{\text{ext}}| = 2 = T$.

To solve this model with AO, we construct the reduced set of intakes $\tilde{\mathcal{I}}$. In order to do so, we compute the PMFs, which takes 2 seconds. Using $\beta = 0.001$, we find the new set of intakes to have $|\tilde{\mathcal{I}}| = 150$, which is a 67% cardinality reduction. Then, we solve the MILP model over $\tilde{\mathcal{I}}$ and find the solution $y^{\text{AO}} = 9$, $\boldsymbol{p}^{\text{AO}} = (0.82, 0.82)$, meaning that AO was optimal with respect to both y and $\boldsymbol{p}$ in this instance. However, it took 0.71 seconds in total, as opposed to CS's 0.17 seconds.

We can also run this instance with CS_opt. Doing so, CS_opt's first two iterations are the same as CS's. In its third iteration it finds $y^3 = 9$ and $\boldsymbol{p}^3 = (0.82, 0.82)$, whereas CS found $\boldsymbol{p}^3 = (0.84, 0.79)$. Following this, in iteration $k = 4$ it finds $\boldsymbol{p}^4 = (0.82, 0.82)$ and breaks since $\boldsymbol{p}^4 \in \Theta^4$, returning $y^{\text{CS\_opt}} = 9$ and $\boldsymbol{p}^{\text{CS\_opt}} = (0.82, 0.82)$. This is the same solution as P gave. This took CS_opt a total of 0.43 seconds. It finished in twice as many iterations as CS.

2.4 Design of Computational Experiments

This section details our experiments evaluating the performance of the algorithms described in Section 2.3.6 in comparison with the solution from the parametric DRO model. These experiments will also allow us to compare the solutions resulting from the *parametric model* (P) and the *non-parametric model* (NP) and the times taken to reach optimality by each model. In this section, we discuss how the parameters for the experiments will be chosen to ensure that they are representative of typical real-life scenarios. To discuss experimental design, we need to define which parameters of the

model will be varied and the values that they will take. The vector of inputs to the model for a fixed set $\mathcal{I}$ of intakes and $\mathcal{P}$ of distributions is $\boldsymbol{S} = (\boldsymbol{c}, \boldsymbol{D}, \boldsymbol{a}, T, K)$.

2.4.1 Parameter Hierarchy

It is helpful to consider a hierarchy of parameter choices, which is defined by:

1. (T, K) defines the difficulty of the problem in terms of the MILP itself.

2. $\boldsymbol{c}$ and $\boldsymbol{D}$ define the set of solutions that are possible for a given model with fixed T and K. They need to be constructed for each combination of T and K to ensure that we have a varied range of instances when it comes to pulling forward opportunities. We create this variety by varying the number of days that have spare capacity and are hence able to receive additional jobs. The values of $\boldsymbol{c}$ and $\boldsymbol{D}$ used are discussed in Section 2.4.2.

3. (a) For the parametric model, $\mathcal{I}$ and Θ define how the uncertainty is encoded in the model, depending on the planner's attitude to risk. If $|\mathcal{I}|$ or $|\Theta|$ is large, solving to optimality will be very slow, and we would like to use a heuristic that is not significantly affected by these sizes. $|\mathcal{I}|$ is defined by $\boldsymbol{i}^{\max}$, and $|\Theta|$ is defined by two parameters. The initial discretisation of the interval $[0, 1]$ in which each p_t lies is defined by M.

   The maximum distance from the nominal distribution that $\boldsymbol{p} \in \Theta$ can lie is defined by the second parameter, N. This is the number of samples that we take from the distribution of $\boldsymbol{I}$ in order to calculate $\hat{\boldsymbol{p}}$. Larger N results in smaller distances from $\hat{\boldsymbol{p}}$ being allowed, and hence corresponds to a less

risk-averse planner. For these experiments, we use 95% confidence sets, i.e. Θ'_α from (2.3.16) with $\alpha = 0.05$. From now on, we use Θ to represent $\Theta'_{0.05}$.

(b) For the non-parametric model, we also use 95% confidence sets. However, for this model we use the ϕ-divergence based set, $\mathcal{P}_\kappa$, given in (2.3.10) with κ defined by (2.3.11) and $\alpha = 0.05$. This set is only affected by N, which affects the maximum distance from $\hat{\boldsymbol{P}}$ that a distribution can lie under the non-parametric model.

4. $\boldsymbol{a}$ will be left as the ones vector for these experiments as it has not been seen to have an effect on solutions.

We choose $T = 5$ due to it being the number of days in a typical working week. We take the maximum pulling forward window length to be $K = 2$. This is because pulling forward is not enacted until the operational planning phase, where the planning horizon is very short. These choices are partly motivated by usual practices, and also partly by the following fact. We aim to test our heuristics against optimal solutions, and for larger T or K the model becomes very difficult to solve to optimality. Note that the optimality tolerance for CS/CS_opt, ε, will be set to 0.01 and it will be run for a maximum of $k^{\max} = 10$ iterations. Initial testing suggested that these parameters are not so important, as CS and CS_opt always terminated due to a repeat parameter (i.e. $\boldsymbol{p}^k \in \Theta^k$) after less than 10 iterations.

2.4.2 Capacity and Workstacks

The factors affecting the potential solutions of a model the most are $\boldsymbol{c}$ and $\boldsymbol{D}$, due to the fact that they define the rollover and pulling forward opportunities. In this section, we detail the capacities $\boldsymbol{c}$ and workstacks $\boldsymbol{D}$ used in our experiments. These are constructed with the aim of ensuring that a variety of combinations of pulling forward opportunities are represented by at least one $(\boldsymbol{c}, \boldsymbol{D})$ pair. We assume for this section that the previous parameters in the hierarchy, i.e. T and K, are given. We now define how $\boldsymbol{c}$ and $\boldsymbol{D}$ define pulling forward opportunities mathematically. Firstly, we define the set of pairs of days under consideration for pulling forward as:

$$\mathcal{F} = \{(t_1, t_2) \mid t_1 \in \{2, \dots, T\}, t_2 \in \{t_1 - K, \dots, t_1 - 1\}\}$$

and the set of pairs such that the corresponding y can feasibly be positive given $\boldsymbol{c}$ and $\boldsymbol{D}$ as:

$$\mathcal{F}^+(\boldsymbol{c}, \boldsymbol{D}) = \{(t_1, t_2) \in \mathcal{F} \mid c_{t_2} > D_{t_2}, D_{t_1} > 0\}.$$

This is the set of all pairs of days (t_1, t_2) such that t_2 is within pulling forward range of t_1, t_2 has spare capacity and t_1 has workstack jobs to be completed early. For our experiments, we consider instances where $D_t > 0$ for all $t \in \{1, \dots, T\}$. This is because for a short horizon of $T = 5$ days, it is very unlikely that any day will have a workstack of zero. Hence, we can control $|\mathcal{F}^+(\boldsymbol{c}, \boldsymbol{D})|$ by controlling which days have spare capacity. For example, we can set $|\mathcal{F}^+(\boldsymbol{c}, \boldsymbol{D})| = 3$ by setting $D_1 < c_1$ and $D_4 < c_4$ and then $D_t > c_t$ for $t \in \{2, 3, 5\}$. This results in $\mathcal{F}^+(\boldsymbol{c}, \boldsymbol{D}) = \{(2, 1), (3, 1), (5, 4)\}$.

We do this similarly for other values of $|\mathcal{F}^+(\boldsymbol{c}, \boldsymbol{D})|$. The main effect that $\boldsymbol{c}$ and $\boldsymbol{D}$ have on decision making is that they define the constraints on y, meaning their only

important quality is how much pulling forward they do or do not allow. Using this set of values for $\boldsymbol{c}$ and $\boldsymbol{D}$ we will be able to see how well our algorithms detect and make use of opportunities for pulling forward.

2.4.3 Uncertainty and Ambiguity Sets

As a reminder, the term "uncertainty set" refers to $\mathcal{I}$ and "ambiguity set" refers to Θ. We now detail the parameters used to construct these sets in our instances.

Uncertainty Sets

We assumed in Section 2.1.1 that we would be given a set $\mathcal{I}$, either by expert knowledge or by a prediction model. We could then extract $\boldsymbol{i}^{\max}$ from this set. However, in these experiments, we do not have access to real intake data or expert knowledge. Thus, it is more convenient to define $\boldsymbol{i}^{\max}$ and then use this to construct $\mathcal{I}$. Since there is a one-to-one mapping between the two, both methods achieve the same result. We consider $\boldsymbol{i}^{\max}$ satisfying:

$$\sum_{t=1}^{T} i_t^{\max} \leq \sum_{t=1}^{T} \max\{c_t - D_t, 0\}. \tag{2.4.1}$$

This is reasonable because if the total number of jobs arriving in the system exceeds the RHS of (2.4.1) then some intake jobs will always remain incomplete at the end of day T, regardless of our pulling forward decision. Furthermore, we can vary the number of high-intake days, through the quantity $n(\boldsymbol{i}^{\max}) = |\{t \in [T] : i_t^{\max} > c_t - D_t\}|$.

This corresponds to the number of days with the potential for spikes in demand. Depending on $\boldsymbol{c}$ and $\boldsymbol{D}$, $n(\boldsymbol{i}^{\max})$ can range between 0 and $T-1$. However, for these

experiments we consider $n(\boldsymbol{i}^{\max}) \in \left\{1, \left\lfloor \frac{T}{2} \right\rfloor, T-1\right\}$ for sufficient coverage of cases. The case of $n(\boldsymbol{i}^{\max}) = 1$ corresponds to a one-day spike caused by an event such as a major weather event. The case of $n(\boldsymbol{i}^{\max}) = \left\lfloor \frac{T}{2} \right\rfloor$ could correspond to an extended spike lasting for multiple consecutive days, for example, a network problem causing lots of service devices to break. The final case of $n(\boldsymbol{i}^{\max}) = T-1$ corresponds to $T-1$ small spikes in intake, marking a period of consistently high intake.

Ambiguity Sets

The choice of parametric ambiguity set depends on the choice of discretisation of $[0,1]^T$ and also the way we in which we then reduce its size. The choice of discretisation is defined by the parameter M, and increasing this value increases the size of the ambiguity set. For these experiments, we consider $M \in \{5, 10, 15\}$. In our preliminary testing we found that any value larger than 15 can lead to intractability when solving the parametric model to optimality.

Both ambiguity sets are also defined by the sampling parameter N. For the purpose of testing our models, we consider $N \in \{10, 50, 100\}$. Clearly, higher N leads to better convergence to the true distribution of the MLE/ϕ-divergence, but it also leads to much smaller ambiguity sets and typically less conservative decisions. Even for $N = 50$, we obtained some singleton ambiguity sets. Typically, N would be chosen by the planner who is in control of the sampling process. However, the results of our testing can be used to understand the tradeoff between the accuracy of the approximation and the conservativeness of the resulting decisions. Hence, they may influence the value of N used by the planner. In these experiments, we will assume

$\hat{\boldsymbol{i}} = (0.75i_1^{\max}, \ldots, 0.75i_T^{\max})$. Hence, we will obtain $\hat{\boldsymbol{p}} = (0.75, \ldots, 0.75)$. In practice, $\hat{\boldsymbol{p}}$ would be obtained from sampling the true intake distribution. However, without access to true intake data, we set the value somewhat arbitrarily, since it is only used for testing purposes. If these models were used by a real planner, we would suggest that they calculate their own MLE.

2.5 Results

We now detail the results of our experiments that we used to test the algorithms on 279 problem instances with $T = 5$ and $K = 2$. We report the results from all 5 algorithms in terms of times taken, pulling forward decisions and worst-case distributions. Due to space considerations, we present some additional results in the Appendices. We discuss the effects of workstacks on solutions in Appendix A.2.1. We give a brief comparison of our results with those from the RO version of the model in Appendix A.2.2. In addition, we present and test a Benders decomposition algorithm for this problem in Appendix A.3. These experiments were run in parallel on a computing cluster (STORM) which has 486 CPU cores. The solver used in all instances was the Gurobi Python package, gurobipy (Gurobi Optimization, LLC, 2022). The version of gurobipy used was 9.0.1. The node used on STORM was the Dantzig node, which runs the Linux Ubuntu 16.04.6 operating system, Python version 2.7.12, and 48 AMD Opteron 638 CPUs.

2.5.1 Summary of Instances and Their Sizes

In Table 2.5.1, we summarise the sizes of the sets $\mathcal{I}$ and Θ, that formed the basis for the constraints and variables in the model. As a reminder, we use Θ to mean Θ'_α as defined in (2.3.16). A summary of the sizes of $\mathcal{I}$ is given in Table 2.5.1a. The table shows 7 of the 31 $\boldsymbol{i}^{\max}$ values considered and the size of the resulting set $\mathcal{I}$. The other $\boldsymbol{i}^{\max}$ values considered were permutations of the values shown in the table, and hence led to $|\mathcal{I}|$ values that are already listed in the table. Table 2.5.1b shows the values of N and M used and the average size of the resulting ambiguity sets. The sizes vary as the construction of the set also depends on $\boldsymbol{i}^{\max}$. The instances where $|\Theta| = 1$ correspond to instances where $\kappa = \frac{\chi^2_{T,1-\alpha}}{N}$ was too small to allow any $\boldsymbol{p}$ other than $\hat{\boldsymbol{p}}$ to be in the ambiguity set defined by (2.3.16).

We can see here that our choices of $\boldsymbol{i}^{\max}$ gave instances with as many distinct intakes (and rollover vectors) as 20000, and as few as 392. The sizes of the ambiguity sets varied between 1 and 8854, where the largest sets resulted from the smallest $\boldsymbol{i}^{\max}$ and N values, and the largest M values. This is because the criteria for $\boldsymbol{p}$ being included in Θ was $\sum_{t=1}^{T} N i_t^{\max} \frac{(\hat{p}_t - p_t)^2}{\hat{p}_t(1-\hat{p}_t)} \leq \chi^2_{T,1-\alpha}$. Clearly the LHS is increasing in N and $i_t^{\max}$. Hence, larger values lead to a higher distance from the nominal distribution. Large M leads to larger Θ because it results in a finer discretisation of $[0,1]^T$, and hence more candidate $\boldsymbol{p}$ values.

$\boldsymbol{i}^{\max}$	$\lvert\mathcal{I}\rvert$
(1, 6, 6, 1, 1)	392
(1, 3, 3, 3, 3)	512
(2, 2, 2, 6, 2)	567
(2, 2, 8, 8, 2)	2187
(5, 5, 1, 5, 5)	2592
(1, 7, 7, 7, 7)	8192
(9, 9, 1, 9, 9)	20000

(a) Example $\boldsymbol{i}^{\max}$ values and sizes of the associated uncertainty sets $\mathcal{I}$ considered.

N	M	Average $\lvert\Theta\rvert$
100	5	1.000
100	10	1.000
100	15	16.871
50	5	1.419
50	10	14.419
50	15	93.129
10	5	14.742
10	10	504.226
10	15	4301.645

(b) Parameters defining ambiguity sets and average size of corresponding sets.

Table 2.5.1: Summary of input parameters and corresponding set sizes

2.5.2 Optimality of Algorithms and Times Taken

Comparing results for DRO problems is not as simple as comparing final objective values. Our optimal objective value can be written as $z^* = \min_{\boldsymbol{y}} \max_{\boldsymbol{p}} f(\boldsymbol{y}, \boldsymbol{p})$. Here, $f(\boldsymbol{y}, \boldsymbol{p})$ is the total expected rollover cost, i.e. $\sum_{t=1}^{T} a_t \mathbb{E}_p(R_t \mid \boldsymbol{y})$. Suppose we have an instance where $\boldsymbol{y}^{\mathrm{CS}} = \boldsymbol{y}^{\mathrm{P}}$ but $\boldsymbol{p}^{\mathrm{CS}} \neq \boldsymbol{p}^{\mathrm{P}}$. Then, if CS gives a lower objective value than P, it may appear to have given a better solution to the minimisation problem. However, this means that CS did not successfully choose the worst-case $\boldsymbol{p}$ for its chosen $\boldsymbol{y}$. This leads to a lower objective function value but a suboptimal solution with respect to $\boldsymbol{p}$. Similarly, we can say that CS is suboptimal if $\boldsymbol{p}^{\mathrm{P}} = \boldsymbol{p}^{\mathrm{CS}}$

but $\boldsymbol{y}^{\mathrm{CS}} \neq \boldsymbol{y}^{\mathrm{P}}$ and CS gave a higher objective value. Hence, both a higher and a lower objective value can suggest suboptimality for a DRO model. Given this, we summarise the results using 3 optimality criteria. Algorithm $x \in \{\mathrm{CS}, \mathrm{CS\_opt}, \mathrm{AO}\}$ is said to be:

1. ***y*-optimal** if $\max_{\boldsymbol{p}\in\Theta} f(\boldsymbol{y}^x, \boldsymbol{p}) = z^*$.
2. ***p*-optimal** for a given $\boldsymbol{y}^x$ if $f(\boldsymbol{y}^x, \boldsymbol{p}^x) = \max_{\boldsymbol{p}\in\Theta} f(\boldsymbol{y}^x, \boldsymbol{p})$.
3. ***Optimal*** if $f(\boldsymbol{y}^x, \boldsymbol{p}^x) = z^*$. Note that this is met is the algorithm is both $\boldsymbol{y}$-optimal and $\boldsymbol{p}$-optimal.

	No. (%) Optimal Sol	No. (%) $\boldsymbol{p}$-Optimal Sol	No. (%) $\boldsymbol{y}$-Optimal Sol
CS	257 (92.11%)	257 (92.11%)	272 (97.49%)
CS_opt	279 (100.0%)	279 (100.0%)	279 (100.0%)
AO	223 (79.93%)	263 (94.27%)	239 (85.66%)

Table 2.5.2: Summary of optimality of heuristics

We display the number of times each algorithm was optimal, $\boldsymbol{p}$-optimal and $\boldsymbol{y}$-optimal in Table 2.5.2. Table 2.5.2 shows that CS was optimal in 92% of instances, and $\boldsymbol{y}$-optimal in 97%. As can be expected, CS_opt was optimal in every instance. AO was only optimal in 80% of instances and $\boldsymbol{y}$-optimal in 86% of instances. In fact, both CS and AO were optimal in selecting $\boldsymbol{p}$ in more than 92% of instances. Unsurprisingly, AO performs the best in this regard. This is because it solves the problem over the full set of distributions, unlike CS. However, CS was still $\boldsymbol{p}$-optimal in around 92% of

instances. Closeness to optimality of the algorithms is discussed in Section 2.5.3.

A summary of the computation times of each algorithm is given in Table 2.5.3. Firstly, the table shows average and maximum times taken over all instances. CS took around 17 seconds on average. To find the optimal solution, it took approximately 1 minute and 50 seconds on average when using P, which is a large difference. CS_opt found the optimal solution in an average of 20 seconds, which is faster than P. This is only 3 seconds slower than CS on average. However, there are many instances with small ambiguity sets. AO took similar times to CS; it also took around 17 seconds on average. NP solved faster than P, but slower than CS, CS_opt and AO. The fact that NP was slower than CS_opt suggests that the parametric model can be solved to optimality faster than the non-parametric model.

AS reports the times taken to compute Θ for the parametric algorithms. This was not included in the solution time for each algorithm, as it is a pre-computation step. It is worth noting that the average of 6 seconds is significantly faster than extracting Θ from the non-parametric confidence set, which can take hours. Please note that, while the differences between the algorithms' times may seem small, these instances are small compared to real planning instances. We would expect the time differences to be more pronounced when the problems are large. Furthermore, CS_opt requires significantly more memory and computing power than CS. For instances with large ambiguity sets, it stores thousands of distributions, each of which comprises thousands of values. CS only stores around T distributions, regardless of the size of Θ.

Since there were a large number of small instances that affected the overall averages,

	Avg. t.t. (Overall)	Max t.t. (Overall)	Avg. t.t. (Large)	Max t.t. (Large)
P	0:01:22.85	0:19:23.5	0:07:57.99	0:19:23.5
CS	0:00:17.48	0:01:50.37	0:00:06.1	0:00:33.82
CS_opt	0:00:20.17	0:02:12.09	0:00:24.74	0:01:07.84
AO	0:00:17.29	0:03:52.97	0:02:08.08	0:03:52.97
NP	0:00:25.35	0:03:26.88	0:00:07.25	0:00:44.4
AS	0:00:05.95	0:00:19.38	0:00:14.18	0:00:16.59

Table 2.5.3: Summary of times taken

Table 2.5.3 also shows average and maximum times for instances with the largest ambiguity sets. This corresponds to the largest 10% of instances with respect to Θ or equivalently $|\Theta| \geq 1000$. From these two columns, we see that CS_opt took more than 4 times longer than CS on average, when Θ was large. We also see that CS_opt took 34 seconds longer to solve its slowest instance than CS took for its slowest instance. The largest time difference was 46 seconds, and this occurred when $|\Theta| = 831$ and $|\mathcal{I}| = 20000$. This time difference was due to two main reasons. Firstly, CS never spent more than 0.5 seconds computing PMFs, whereas CS_opt took up to 22 seconds. Hence, CS significantly reduced the amount of computation required. Secondly, CS typically completed in many fewer iterations than CS_opt. This is because its use of Θ^{ext} meant it identified a repeat parameter in fewer iterations. Based on the optimality counts and time taken, CS is the strongest heuristic. It selected the optimal $\boldsymbol{y}$ is 97% of instances, and did so in less time than CS_opt. CS_opt can be used when Θ is small, but it will begin to solve slowly in comparison with CS when Θ is large.

2.5.3 Performance of Algorithms in Detail

To illustrate further how well the algorithms performed, we define the following two metrics. Note that a positive value for *either* of these metrics suggests suboptimality.

1. **Quality of $\boldsymbol{p}$ choices.** For a solution $\boldsymbol{y}^x$ that was selected by an algorithm x, where $x \in \{\text{CS}, \text{CS\_opt}, \text{AO}\}$, we calculate the worst-case expected cost over all $\boldsymbol{p} \in \Theta$ using brute force. We can then compare this cost with the expected cost obtained by the algorithm, i.e. from $\boldsymbol{p}^x$, the $\boldsymbol{p}$ that the algorithm selected. This allows us to establish how close to worst-case the choices of $\boldsymbol{p}$ were. We refer to this difference as the $\boldsymbol{p}$-*gap*, and it is defined as:

$$g_{\boldsymbol{p}}(\boldsymbol{y}^x, \boldsymbol{p}^x) = \max_{\boldsymbol{p} \in \Theta} f(\boldsymbol{y}^x, \boldsymbol{p}) - f(\boldsymbol{y}^x, \boldsymbol{p}^x).$$

2. **Quality of $\boldsymbol{y}$ choices.** For a given solution $\boldsymbol{y}^x$ from algorithm x, we compute the worst-case expected cost using brute force, as we did when finding $g_{\boldsymbol{p}}(\boldsymbol{y}^x, \boldsymbol{p}^x)$. We can then compare this worst-case cost with that of the optimal $\boldsymbol{y}$, to assess how close $\boldsymbol{y}^x$ is to optimal. This is referred to as the $\boldsymbol{y}$-*gap*, and is defined as:

$$g_{\boldsymbol{y}}(\boldsymbol{y}^x) = \max_{\boldsymbol{p} \in \Theta} f(\boldsymbol{y}^x, \boldsymbol{p}) - z^*.$$

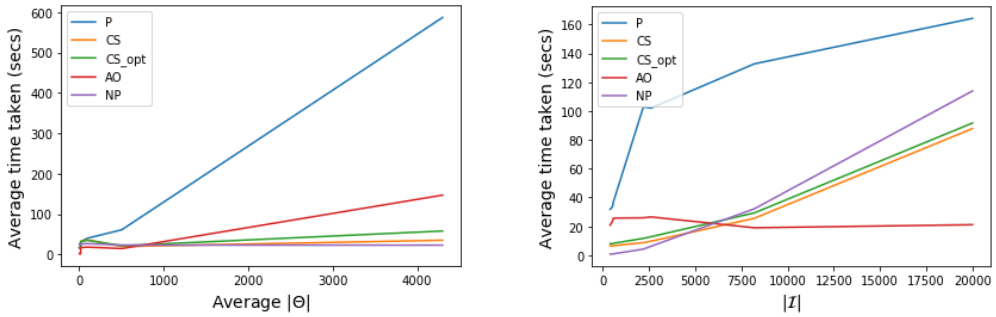
In Table 2.5.4, we summarise the average $\boldsymbol{p}$-gaps and $\boldsymbol{y}$-gaps of the three heuristics, along with the average *absolute percentage gaps* (APGs). The $\boldsymbol{p}$-APG was obtained by taking the $\boldsymbol{p}$-gap as an absolute percentage of the worst-case expected cost for the chosen solution $\boldsymbol{y}^x$. The $\boldsymbol{y}$-APG was obtained by taking the $\boldsymbol{y}$-gap as an absolute percentage of the optimal objective value.

	Avg. $\boldsymbol{p}$-gap	Avg. $\boldsymbol{p}$-APG	Avg. $\boldsymbol{y}$-gap	Avg. $\boldsymbol{y}$-APG
CS	0.0561	0.084%	0.0058	0.0101%
CS_opt	0.0000	0.0%	0.0000	0.0%
AO	0.0064	0.0065%	0.0233	0.1369%

Table 2.5.4: Summary of gaps and APGs of the heuristics

This suggests that all algorithms perform very well at choosing the worst-case $\boldsymbol{p}$ for a fixed $\boldsymbol{y}^x$, since all had an average $\boldsymbol{p}$-APG of less than 0.09%. AO performed the best at selecting $\boldsymbol{p}$, which supports the observation made from the optimality counts. CS and AO are very good at selecting the optimal $\boldsymbol{y}$, since they both use a solver to do so. Of CS and AO, CS performed the best in this regard, with an average $\boldsymbol{y}$-APG of 0.01%. The $\boldsymbol{y}$ solution CS chose had, on average, a worst-case expected cost that was 0.0058 away from the optimal objective value. AO also performed well in selecting $\boldsymbol{y}$, but its average $\boldsymbol{y}$-APG was a factor of 20 larger than that of CS. Due to its optimality in every instance, CS_opt had average gaps and APGs of 0.

We also study the results broken down by the size of the set of distributions. In order to reduce the size of the table, we present results averaged over the categories for $|\Theta|$ given in Table 2.5.1b. We present these results in Table A.4.1, which is in Appendix A.4.1 due to space considerations. In summary, the table suggests that CS did not return suboptimal $\boldsymbol{p}$s for its chosen $\boldsymbol{y}$ until the set reached the average size of 93. CS was consistent in its $\boldsymbol{y}$-gaps across all values of $|\Theta|$. CS's $\boldsymbol{y}$-APG stayed very close to 0 in all instances. AO had larger $\boldsymbol{p}$-gaps for larger $|\Theta|$.



(a) Average times taken by $|\Theta|$ (b) Average times taken by $|\mathcal{I}|$

Figure 2.5.1: Average times taken by sizes of sets

Interestingly, AO's performance in selecting $\boldsymbol{y}$ improved as $|\Theta|$ grew larger. CS_opt had zero gaps and APGs for all values of $|\Theta|$, but its times taken did not scale as well as CS's and AO's with large $|\Theta|$. For small ambiguity sets, CS_opt took similar times to CS, but it took twice as long for the largest ambiguity sets (average size of 4301). We also plot the average times by $|\Theta|$ in Figure 2.5.1a. This plot suggests that the algorithms that use Gurobi on the full set of distributions, i.e. P and AO, do not scale well with $|\Theta|$ in terms of time. CS, CS_opt and NP all scale much better with $|\Theta|$ than AO and P. For CS and CS_opt, this is because they only ever solve an MILP reformulation over a small subset of Θ. For NP, this is because increasing the size of the ambiguity set for the non-parametric model does not result in a more complex model, it only increases κ. This plot supports our conclusion that CS_opt solves in similar times to CS when Θ is small, but takes noticeably longer for large Θ.

Finally, we can look at the performance of the algorithms by the size of the set of intakes $\mathcal{I}$. These results are shown in Table A.4.2 in Appendix A.4.2. The $\boldsymbol{p}$-APGs for the heuristics were not significantly affected by $|\mathcal{I}|$, apart from a drop in perfor-

mance for CS when $|\mathcal{I}| = 8192$. This was likely due to other model parameters, since there is no reason for $|\mathcal{I}|$ to affect the $\boldsymbol{p}$-APG. AO also began to lose $\boldsymbol{y}$-performance when $|\mathcal{I}| = 8192$. This is an intuitive result, because as this set gets larger AO will remove more and more intakes. This reduces the accuracy of its approximation of the objective function. CS does not remove intakes, which explains why its performance was consistent. In fact, CS's $\boldsymbol{y}$-APG was lower than AO's when $|\mathcal{I}| = 20000$. Again, CS_opt had all zero gaps and APGs. The difference between CS and CS_opt in terms of times taken is less noticeable here. CS_opt consistently took 3-5 seconds longer than CS for all values of $|\mathcal{I}|$. This indicates that $|\Theta|$ was the main factor causing CS_opt to solve slowly. We also plot the average times by $|\mathcal{I}|$ in Figure 2.5.1b. This plot suggests that P does not scale well with $|\mathcal{I}|$, and that AO scales very well with $|\mathcal{I}|$. CS, CS_opt and NP scale better than P, but not nearly as well as AO, due to the fact that they do not apply dimension reduction to $\mathcal{I}$.

2.5.4 CS's Suboptimal Distributions

In this section, we compare the solutions and distributions from CS with those from P. Since CS is only limited by its performance in selecting $\boldsymbol{p}$, we study CS's worst-case $\boldsymbol{p}$s in order to find ways to improve its performance. We do not study CS's performance with respect to $\boldsymbol{y}$, since if Θ^{ext} contains $\boldsymbol{p}^{\text{P}}$ then CS will return the same $\boldsymbol{y}$ as P, as evidenced by CS_opt. Hence, improving Θ^{ext} is sufficient to improve CS with respect to $\boldsymbol{y}$ and $\boldsymbol{p}$. We do not analyse AO's solutions, since improving its performance can only come from tuning β.

As shown in Table 2.5.2, CS chose the optimal $\boldsymbol{p}$ for its selected $\boldsymbol{y}$ in 92% of instances, leaving 22 instances where it did not. This indicates that our set Θ^{ext} did not in fact contain the worst-case $\boldsymbol{p}$ in those 22 instances. To compare CS with P, we study only instances where CS selected the same $\boldsymbol{y}$ as P, which occurred in 15 of these 22 instances. Firstly, for these 15 instances, we can confirm that $\boldsymbol{p}^{\text{P}}$ was not contained in the set Θ^{ext} used by CS. This either occurred because no probability was at its maximum, or because the sum of the probability vector was not maximised. We find that one value of p_t^{P} was maximised in 13 out of 15 instances. However, in every one of these 13 instances, the sum over the vector was not maximised. This indicates that the main reason why CS did not return the worst-case $\boldsymbol{p}$ in every instance was because the worst-case does not need to satisfy this condition. In general, we find that CS both allocated a higher maximum success probability and more success probability in total than P.

In order to see why the worst-case $\boldsymbol{p}$ does not need to satisfy the sum-maximisation criterion, we study some examples more closely. For example, in one instance we had $\boldsymbol{p}^{\text{P}} = (0.933, 0.867, 0.867, 0.867, 0.733)$ and $\boldsymbol{p}^{\text{CS}} = (0.933, 0.933, 0.867, 0.8, 0.867)$. We see that P and CS both gave maximal probability to day 1. However, P reduced days 2 and 5's probabilities in order to allocate more to day 4. The resulting rollover vectors were $(0.87, 10.6, 27.34, 24.54, 41.01)$ for P and $(0.87, 10.74, 27.47, 24.27, 41.0)$ for CS. In this instance, allocating higher probability to day 4 resulted in higher day-4 and also day-5 rollover, and more rollover in total, despite the fact that the total probability was not maximised. Another way that CS can be suboptimal is choosing

the wrong day to set to its maximum.

For example, in one of the 15 instances P gave $\boldsymbol{p}^{\mathrm{P}} = (0.933, 0.867, 0.8, 0.8, 0.8)$ and CS gave $\boldsymbol{p}^{\mathrm{CS}} = (0.8, 0.867, 0.8, 0.867, 0.867)$. Here, CS set p_2 to its maximum, while P set p_1 at its maximum. For this instance, the closest values of $\boldsymbol{p}$ to $\boldsymbol{p}^{\mathrm{P}}$ that were in Θ^{ext} were $(0.933, 0.8, 0.8, 0.867, 0.933)$ and $(0.933, 0.8, 0.8, 0.933, 0.867)$. These two solutions give less expected cost than $\boldsymbol{p}^{\mathrm{CS}}$, and so CS did not allocate maximal probability to day 1. Clearly the maximal cost came from allocating high probability to day 2 as well as day 1, but no such probability vectors were contained in Θ^{ext}. In addition, in two instances no value of $\boldsymbol{p}^{\mathrm{P}}$ was at its maximum. One example of this occurred when $\boldsymbol{p}^{\mathrm{P}} = (0.867, 0.867, 0.867, 0.733, 0.733)$ and $\boldsymbol{p}^{\mathrm{CS}} = (0.8, 0.933, 0.8, 0.8, 0.8)$. CS has allocated day 2 its maximum probability. However, the worst-case parameter spread the success probability more evenly over the first 3 days.

These observations explain why CS did not always return the true worst-case $\boldsymbol{p}$. Clearly, the issue lies in the construction of Θ^{ext}. In particular, the assumption that the sum over the success probability vector should be maximised is not always required. In fact, sometimes it is worse to reduce the sum in order to give high priority days a higher success probability. In order to assess whether or not this is the case, CS would need to compare the maximum intakes for each day in order to see where the most rollover could be caused.

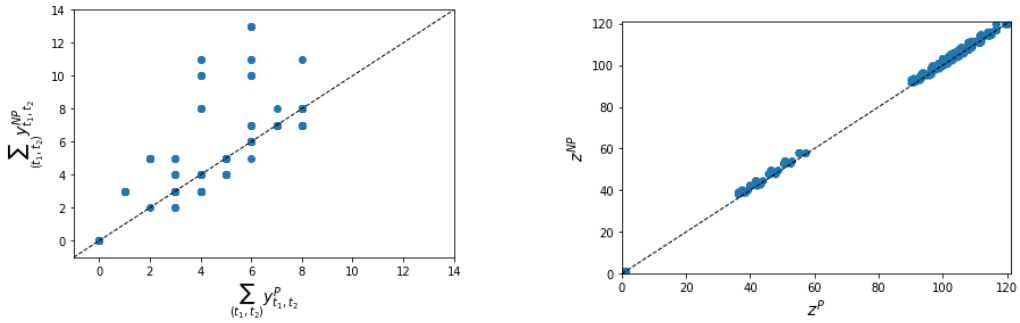
2.5.5 Parametric vs. Non-parametric Decisions and Distributions

In this section, we compare NP's solutions and distributions with those from P. This will allow us to assess the benefits and costs of including the parametric information in the model. As we have seen, incorporating this information creates a model that is larger and computationally more difficult to solve. However, it retains the information on the family of distributions that $\boldsymbol{P}^0$ lies in and ensures that the worst-case distribution from the model is also in this family. This is something that is not guaranteed by the non-parametric model.

Pulling Forward Decisions and Objective Values

We first study the differences in pulling forward decisions between the two models along with their worst-case objective values. We find that the two models gave the same pulling forward decision in 199 of the 279 instances solved. This can be stated as NP being $\boldsymbol{y}$-optimal with respect to the parametric model in 71% of instances. In every one of these instances, it was only optimal to pull forward between either days 2 and 1 or not at all. The worst-case expected cost from NP was 1.21 higher than that from P in these instances, on average. This suggests that the worst-case distribution from NP for a fixed $\boldsymbol{y}$ is typically worse than that from P.

Figure 2.5.2a shows a scatter plot of the total amount pulled forward under each model in each of the 279 instances. Figure 2.5.2b shows the corresponding worst-case expected costs. The dashed line corresponds to instances where both models pulled



(a) Amount pulled forward under P vs. NP

(b) Worst-case expected costs under P vs. NP

Figure 2.5.2: Scatter plots comparing P and NP's pulling forward decisions

forward the same amount or had the same worst-case cost. The points in Figure 2.5.2a where the decisions were different suggests that there is no definitive answer to which model's decision is more conservative. In 42 instances NP pulled forward more, and in 38 instances it pulled forward less. However, when NP pulled forward more than P, it pulled forward up to 7 jobs more. When P pulled forward more, it only pulled forward 1 job more. On average over the instances where the two solutions were different, NP pulled forward 1.24 more jobs. The overall average difference was 0.32.

This suggests that NP is generally slightly less conservative than P. However, as shown in Figure 2.5.2b, rarely did NP attain a lower worst-case expected cost than P. The overall average difference between P and NP's worst-case expected costs was -1.21. This suggests that NP's worst-case distribution typically suggests that there will be 1.21 more jobs being expected to roll over in the worst case. This is surprising since NP typically pulled forward more. Hence, this result indicates that NP's less conservative nature led to more expected rollover in the majority of these instances.

Since NP results from relaxing the requirement that the worst-case distribution is binomial, we can view NP as a heuristic for solving the parametric model. Hence, it may be beneficial to study the expected cost resulting from $\boldsymbol{y}^{\mathrm{NP}}$ under the binomial worst-case distribution that would be given by P, instead of the distribution given by NP. Therefore, for each value of $\boldsymbol{y}^{\mathrm{NP}}$, we compute the worst-case binomial distribution given by a $\boldsymbol{p} \in \Theta$, and the associated expected cost. This allows us to compute the objective value that $\boldsymbol{y}^{\mathrm{NP}}$ would attain under the parametric model. Hence, it allows us to assess the quality of $\boldsymbol{y}^{\mathrm{NP}}$ in comparison with $\boldsymbol{y}^{\mathrm{P}}$, as we did for our heuristics.

We can also study the difference between $\boldsymbol{y}^{\mathrm{NP}}$'s worst-case cost under P and NP, via the $\boldsymbol{p}$-gap. This allows us to assess how the two objective functions differ for the same $\boldsymbol{y}$. As a reminder, for an algorithm x the $\boldsymbol{p}$-gap is defined as $g_p(\boldsymbol{y}^x, \boldsymbol{p}^x) = \max_{\boldsymbol{p}\in\Theta} f(\boldsymbol{y}^x, \boldsymbol{p}) - f(\boldsymbol{y}^x, \boldsymbol{p}^x)$, and the $\boldsymbol{y}$-gap is given by $g_{\boldsymbol{y}}(\boldsymbol{y}^x) = \max_{\boldsymbol{p}\in\Theta} f(\boldsymbol{y}^x, \boldsymbol{p}) - z^*$.

	Avg. $\boldsymbol{p}$-gap	Avg. $\boldsymbol{p}$-APG	Avg. $\boldsymbol{y}$-gap	Avg. $\boldsymbol{y}$-APG	$\boldsymbol{y}$-opt. %
NP	-1.1764	13.0095%	0.0234	0.0429%	87.1%
CS	0.0561	0.084%	0.0058	0.0101%	97.1%

Table 2.5.5: Summary of NP and CS's gaps

The gaps for NP are summarised in Table 2.5.5, along with those from CS for comparison. The $\boldsymbol{p}$-gaps show that the worst-case cost for $\boldsymbol{y}^{\mathrm{NP}}$ from NP was 1.18 higher than that from P, on average. This indicates that the NP model typically overestimated the worst-case cost associated with $\boldsymbol{y}^{\mathrm{NP}}$. This is consistent with our previous

observation that NP's worst-case objective values were higher for a fixed $\boldsymbol{y}$. In fact, NP overestimated the worst-case cost of $\boldsymbol{y}^{\text{NP}}$ in 248 of 279 instances (89%). The most that NP overestimated this cost by was 3.4. These values may seem small, but relative to the true worst-case cost they can be quite large. The largest $\boldsymbol{p}$-APG was 165%, indicating that the worst-case cost from NP was 2.65 times that from P. These results indicate that NP will typically give an objective value that makes a decision look worse than it would be in reality. The $\boldsymbol{y}$-APGs suggest that the $\boldsymbol{y}$ decisions from NP performed similarly to that of P, under P's objective. However, they did result in a slight cost increase on average.

Based on the results here, we believe that CS is the strongest performing algorithm. CS ran in less time than NP and gave solutions closer to those from P. In fact, we can say that the NP solutions had gaps that were 4 times higher than CS's on average. Both average gaps were small, but CS was optimal in 92% of instances, as opposed to 71% for NP. Furthermore, if one were to use the NP model, then they would likely overestimate the rollover cost from their decision by approximately 13%, whereas CS would underestimate this cost by approximately 0.084%.

Worst-case Distributions

In order to explain the differences in decisions and costs, we now study the worst-case distributions from P and NP. There are a number of ways in which these distributions can be different. The most obvious one is that P's worst-case distribution is always binomial, whereas NP's is not. As well as this, the two approximations of the 95% confidence set for $\boldsymbol{P}$ can be different, allowing different distances from $\hat{\boldsymbol{P}}$. In fact,

typically the confidence sets for P were larger. This indicates that the parametric sets had better coverage.

We first study the maximum distances from $\hat{\boldsymbol{P}}$ allowed by each ambiguity set and the distances attained by the parametric and non-parametric worst-case distributions, as measured by d_ϕ. We find that the maximum distance allowed by P could be almost twice that allowed by NP. The maximum distance allowed by NP was 1.22, whereas this value was 2.32 for P. This suggests that the parametric ambiguity set can be significantly larger than the non-parametric set. We also find that NP's worst-case distribution always achieved the maximum distance from $\hat{\boldsymbol{P}}$. Interestingly, the same does not apply for P. The maximum distance that $\boldsymbol{P}^{\mathrm{P}}$ had from $\hat{\boldsymbol{P}}$ was 2.00, showing that the worst-case binomial distribution was not always as far from $\hat{\boldsymbol{P}}$ as it was allowed to be. In fact, there were 106 instances where P did not reach its maximum distance. As a result, even though the parametric ambiguity sets allowed $\boldsymbol{P}^{\mathrm{P}}$ to be further from $\hat{\boldsymbol{P}}$, we still find that $\boldsymbol{P}^{\mathrm{NP}}$ was further from $\hat{\boldsymbol{P}}$ on average. The fact that NP's solution was always on the boundary may indicate that the true worst-case distribution was further from $\hat{\boldsymbol{P}}$ than was allowed by NP's ambiguity set.

In order to compare the worst-case distributions directly, we compute a number of summary statistics for each distribution and present their average values in Table 2.5.6. This table also shows the percentage difference between the summary values for the two distributions, which is calculated as $100 \times \frac{\mathrm{NP–P}}{\mathrm{P}}$. The first two results we show are the average distances from $\hat{\boldsymbol{P}}$ as measured by d_ϕ and by the *Kullback-Leibler Divergence* (KLD). The KLD value, $KLD(\boldsymbol{P}^x, \hat{\boldsymbol{P}})$, can be loosely interpreted as the

amount of *surprise* that would result in simulating from $\boldsymbol{P}^x$ if the true distribution were $\hat{\boldsymbol{P}}$. These two rows indicate that NP was further from $\hat{\boldsymbol{P}}$, on average, than P with respect to both distance measures. The values of d_ϕ are quite close, but proportionally the difference in KLD values is much larger. In fact, NP had 52% more surprise than P, on average. This is likely due to the fact that $\boldsymbol{P}^{\text{NP}}$ is not binomial, unlike $\boldsymbol{P}^{\text{P}}$. *Entropy* also measures surprise, but with respect to the values given by the distribution. We see that both distributions had a similar total entropy, but P had slightly more.

	P	NP	% Gap
$d_\phi(\boldsymbol{P}^x, \hat{\boldsymbol{P}})$	0.435	0.480	10.345%
$KLD(\boldsymbol{P}^x, \hat{\boldsymbol{P}})$	0.167	0.254	52.096%
Entropy	5.379	5.227	-2.826%
Total EV	16.701	17.048	2.078%
Total Variance	3.670	3.590	-2.18%
Total Skewness	-4.274	-4.431	3.673%
No. Suppressed	215.556	568.178	163.587%

Table 2.5.6: Summary statistics comparing $\boldsymbol{P}^{\text{P}}$ with $\boldsymbol{P}^{\text{NP}}$

We also present summaries of the total mean, variance and skewness of each distribution. We see that NP had a higher total expected intake than P on average, but less variance. This can be expected since NP can control the mean and variance separately. P, on the other hand, fixes the variance by fixing the mean. P can therefore

have a smaller variance than P, even when the two means are the same. However, P was typically less negatively skewed than NP. These results may explain why NP's worst-case costs were higher. If NP is more negatively skewed with a higher mean and lower variance, then this suggests that more mass is allocated to the higher intakes and less to the lower ones. Hence, expected costs will necessarily be higher.

Finally, we look at the number of intakes that were *popped* and *suppressed* by each worst-case distribution. A distribution $\boldsymbol{P}^x$ popping an intake $\boldsymbol{i}$ is defined as $\mathbb{P}(\boldsymbol{I} = \boldsymbol{i} \mid \boldsymbol{P}^x) > 0$ when $\mathbb{P}(\boldsymbol{I} = \boldsymbol{i} \mid \hat{\boldsymbol{P}}) = 0$. The distribution $\boldsymbol{P}^x$ suppressing $\boldsymbol{i}$ is defined as $\mathbb{P}(\boldsymbol{I} = \boldsymbol{i} \mid \boldsymbol{P}^x) = 0$ when $\mathbb{P}(\boldsymbol{I} = \boldsymbol{i} \mid \hat{\boldsymbol{P}}) > 0$. Since $\hat{\boldsymbol{P}}$ is a binomial distribution, technically we will never have popping as $\mathbb{P}(\boldsymbol{I} = \boldsymbol{i} \mid \hat{\boldsymbol{P}}) > 0 \ \forall \ \boldsymbol{i} \in \mathcal{I}$. We will also never have suppressing under P, for the same reason. In addition, by Bayraksan and Love (2015), the modified χ^2-divergence cannot pop scenarios. Hence, we consider the distributions when rounded to 6 d.p. instead. The table shows that NP popped 55% more intakes than P on average. Both popped only a few intakes, which is consistent with our observation that neither method can technically pop scenarios.

This is not the main cause for the difference in the distributions, however. The main difference is due to suppressing. We see that NP suppressed 163% more intakes than P on average. This indicates that NP's worst-case distribution set a large number of $\hat{\boldsymbol{P}}$'s positive values to zero. P is much more restricted in this sense, due to the fact that $\boldsymbol{P}^{\mathrm{P}}$ is also binomial. This means that P cannot set any values to be exactly zero. This difference may also explain the increased values of KLD given by NP; some intakes that would be generated by $\hat{\boldsymbol{P}}$ would not be generated by $\boldsymbol{P}^{\mathrm{NP}}$.

2.6 Conclusions and Further Research

In this chapter, we presented parametric and non-parametric DRO models for a workforce planning problem under a mixture of known and uncertain demand. We developed heuristics to solve the parametric model, due to its poor scalability. The general conclusions that we can make from our results are as follows. The full model can be slow to solve to optimality using the MILP reformulation, i.e. using P. CS_opt solves this model to optimality in a short time on average, but begins to solve slowly when the ambiguity set is large.

Our heuristics, AO and CS, employ dimension reduction to the sets of intakes and distributions respectively in order to solve the problem in significantly less time than P. The main conclusion we make about these algorithms is that CS performs very well, and takes a fraction of the time that P takes. However, we found that CS can fail to select the worst-case success probability for its chosen pulling forward decision due to its assumption that the total success probability should be maximised. We compared the parametric and non-parametric solutions, and made a number of conclusions. Namely, NP typically pulls forward more than P but it overestimates the worst-case cost of a decision. Our results also suggest that the NP distributions have higher means, lower variance and more negative skewness. They also suppress many more intakes than P's distributions.

The main contribution that we have made to the existing DRO literature is the new modelling framework of parametric DRO. In real-world planning problems, incorporating distributional ambiguity often results in unreasonably slow models. Instead,

data-driven estimates of the demand distribution or its parameters are commonly used. However, this can lead to poor solutions when the estimates are poor. Our methodology provides a way that parameter estimates can be utilised while also hedging against cases when they are inaccurate. It allows planners to build confidence sets around their estimates, that can be adjusted to fit their level of risk aversion. For example, if the planner does not have trust in their estimates then they can choose a large confidence level in order to generate a larger and hence more risk-averse ambiguity set. In addition, our use of parametric distributions instead of non-parametric ones means that the worst-case distribution from our model is more explainable, since it can be summarised by a small number of parameters. This distribution is also less extreme, and less surprising given the estimated distribution. Furthermore, using parametric distributions has allowed us to create fast algorithms for solving the planning problem. This means that planners can incorporate additional uncertainty without having to wait long periods of time for solutions.

There are a number of natural extensions to our work which would be of further interest from a practical viewpoint. Firstly, we have considered a simplified problem in which each job requires one unit of capacity to complete. This is not typically the case in real life workforce planning. Adding more varied completion times would be a clear next step in improving this model. Secondly, the model considers the case where there is only one skill, and is equivalent to assuming all workers can complete any job. In some scenarios this is not the case, and the model could account for this by considering separate demand values and decision variables for each skill. Thirdly, we

have treated the capacity as fixed and aimed to optimise its use. In some cases, if not all, however, capacity can be manipulated in the tactical planning phase. For example, one can order extra units of existing resources (overtime) or hire outside resources for a cost (contractors). These ways to manipulate capacity (planning levers) will form the basis of some of our future research. Finally, we have assumed in this chapter that the intakes are independent. Extending our model to account for correlated intakes is a promising area for future work.

Chapter 3

Parametric Distributionally Robust Optimisation Models for Budgeted Multi-period Newsvendor Problems

In this chapter, we consider a static, multi-period newsvendor problem under a budget constraint. In the case where the true demand distribution is known, we develop a heuristic algorithm to solve the problem. By comparing this algorithm with off-the-shelf solvers, we show that it generates near-optimal solutions in a short time. We then consider a scenario in which limited information on the demand distribution is available. It is assumed, however, that the true demand distribution lies within some given family of distributions and that samples can be obtained from it.

We consider the cases of normal and Poisson demands. For each case, we show that using maximum likelihood estimates in place of the true parameters can lead to poor estimates of the true cost associated with an order quantity. Hence, we make use of likelihood inference to develop confidence sets for the true parameters. These are used as ambiguity sets in a distributionally robust model, where we enforce that the worst-case distribution lies in the same family as the true distribution.

We solve these models by discretising the ambiguity set and reformulating them as piecewise linear models. We show that these models quickly become large as the ambiguity set grows, resulting in long computation times. To overcome this, we propose a heuristic cutting surface algorithm that exploits theoretical properties of the objective function to reduce the size of the ambiguity set. We illustrate that our cutting surface algorithm solves orders of magnitude faster than the piecewise linear model, while generating very near-optimal solutions.

3.1 Introduction

The *newsvendor problem* (Arrow et al., 1951) is a classical problem in operational research and operations management. In this problem, we consider a retailer facing the decision of how much stock to order in order to try to meet uncertain demand as closely as possible. Based on the fact that newspapers are no longer saleable after the period is over, in the newsvendor problem any unsold stock is either lost or sold at a lower price. Furthermore, the newsvendor is typically penalised for missing demand. Even in its earliest form, the newsvendor problem models a situation in which demand

is not known exactly. However, in early papers such as that of Arrow et al. (1951), it was typically assumed that the demand distribution was known exactly. The resulting problem is therefore stochastic, and usually nonlinear.

In the years after its introduction, the newsvendor problem has been extended in many ways. Two such extensions are relevant to this chapter. The first is the *static multi-period newsvendor problem* (Chen et al., 2017). In this version of the problem, the retailer must decide prior to the selling horizon on how much stock to have delivered at the start of each period. This corresponds to cases where the retailer and the supplier must agree on their stocking quantities beforehand, and applies to many industries. As discussed by Chen et al. (2017), this type of ordering structure is relevant to fresh produce industries. In these industries, due to the short shelf-lives of products, orders cannot simply be delivered prior to the selling period and stored for its entirety. Some products can only be stored for a short time and hence must be delivered within the selling horizon to ensure that they do not perish before they are sold.

Furthermore, ordering in advance allows the retailer to use one period's order for multiple subsequent periods, if this yields a higher profit. Depending on the costs associated with holding stock, this can be cheaper than making multiple separate orders. Our model makes an additional extension to the model of Chen et al. (2017), through a budget constraint. This is not common in multi-period models, due to the added complexity this constraint brings, but it is more common in multi-product models (Alfares and Elmorra, 2005). This constraint adds an extra level of realism, since it is very unlikely that any retailer can spend infinite amounts of money. In this

chapter, we develop an iterative algorithm for our model.

Another key extension of the newsvendor problem is the *distributionally robust* (DR) problem. In this version of the problem, we do not assume that the distribution of demand is known. We assume that the true distribution lies in some set, referred to as an *ambiguity set*. Then the problem is to find the ordering quantity with the highest worst-case expected profit over all distributions in this ambiguity set. Early DR problems assumed that only some moments of the distribution, such as mean and variance, are known and fixed. The ambiguity set then contains all distributions whose moments are equal to these values. We refer to problems with moment-based ambiguity sets as *distribution free* (DF) problems (Scarf, 1957).

The assumption that these moments are known is not always realistic, and typically they must be estimated from sample data (Lee et al., 2021). It has been found that poor estimation of these parameters can lead to poor estimation of costs (Rossi et al., 2014) and bias in solutions (Siegel and Wagner, 2021). Furthermore, DF models can lead to overly conservative solutions (Wang et al., 2016). Due to these issues, many recent DR models considered ambiguity sets containing distributions that lie within some pre-prescribed distance of a nominal distribution (Zhao and Guan, 2015).

Our model is different from the standard DR model in two key ways. Firstly, we only consider parametric distributions. To be more specific, we assume that the demand distribution lies in some parametric family, and then construct ambiguity sets that only contain distributions that also lie in this family. Parametric distributions have often been used in the newsvendor problem due to their ability to allow statistical

estimation and analysis to be integrated with optimisation (Liyanage and Shanthikumar, 2005; Rossi et al., 2014). Secondly, due to the pitfalls of poor cost estimation, we do not simply assume that the estimates of these parameters are the truth. Instead, we use maximum likelihood theory to develop confidence sets for the true parameters, and use these as ambiguity sets. The concept of parametric *distributionally robust optimisation* (DRO) was introduced in Chapter 2, in the context of a resource planning problem under binomial demand distributions. In this chapter, we further develop this framework by studying a multi-period newsvendor problem under normal and Poisson demands. This allows us to show how the methods of Chapter 2 can be applied to both continuous and discrete demand distributions with infinite support.

The model resulting from the parametric ambiguity set does not have any convenient reformulation. Hence, we discretise the ambiguity set and represent the inner objective using a set of constraints. Due to the nonlinearity of the objective function, these constraints are non-linear. For discrete distributions, these constraints are piecewise linear, and for continuous distributions we use piecewise linear approximations of them. One piecewise linear constraint is required for each distribution in the ambiguity set and each period considered by the model. Hence, the model can grow very large and become slow to solve. Therefore, inspired by Chapter 2, we develop a heuristic *cutting surface* (CS) algorithm in order to solve the model in a fast time.

CS algorithms have often been used for solving non-parametric DRO problems to ε-optimality (Mehrotra and Papp, 2014). Such CS algorithms were shown to become slow for large parametric problems in Chapter 2. Hence, our algorithm is a heuristic

CS algorithm that exploits knowledge of the distributional family to improve solution times. In general, a CS algorithm iteratively solves the model over subsets of the ambiguity set, each time adding a new distribution to the subset before solving again. This new distribution is chosen by finding the worst-case distribution for the most recent solution. To speed up this stage, we use theoretical properties of the objective function to develop a set of extreme distributions from which to select the worst-case. We will show that our heuristic CS algorithm performs very well and generates solutions in a short time. In summary, the key contributions of this chapter are:

1. We extend the multi-period model of Chen et al. (2017) by including a budget constraint.

2. We present an iterative solution algorithm for the multi-period model under a budget constraint. As far as we know, there is no existing algorithm for this problem.

3. We show that the MLE approach provides poor estimates of the cost of a given order quantity, which can lead to predicting a profit for an order that would result in a loss.

4. We extend the concept of parametric DRO from Chapter 2 to the newsvendor literature. We consider both continuous and discrete random variables with infinite support. Only binomial random variables have been studied before. In addition, this chapter shows how the work of Chapter 2 can be extended to models with nonlinear objective functions.

5. We develop a fast heuristic version of the CS algorithm from the DRO literature. The general CS algorithm is ε-optimal, but can solve slowly for large ambiguity sets. Hence, our heuristic CS algorithm exploits properties of the distributional family to reduce the size of the ambiguity set considered, at little cost to solution quality. We perform extensive computational experiments to test the efficacy of our CS algorithm.

3.2 Literature Review

Our research is based on three areas of the newsvendor literature: DR newsvendor problems, multi-period and capacitated/budgeted newsvendor problems, and parametric newsvendor problems. We now review the literature corresponding to these three variations of the problem.

3.2.1 Distributionally Robust Newsvendor Problems

The first example of the DR newsvendor problem comes from Scarf (1957). Scarf formulated the first DF model under the assumption that only its mean and variance are known. He optimises the worst-case cost over all distributions with this mean and variance, proving a simple ordering rule to be optimal. The DF model has been the subject of many papers since the work of Scarf (1957). For example, Gallego and Moon (1993) extended this work to the multi-product and random yield models. Moon and Choi (1995) also extended the single-period DF model to the case where customers can balk, i.e. decide not to buy an item, after observing stock levels.

Further extensions to the DF model come from Alfares and Elmorra (2005), who extended the work of Gallego and Moon (1993) to the case where the model includes a shortage cost for missing demand. Ouyang and Chang (2002) extended the DF model to incorporate uncertainty in parameters such as lost-sales and backorder rates. Later extensions of the model incorporate addition elements such as the cost of advertising (Lee and Hsu, 2011), risk- and ambiguity-aversion (Han et al., 2014), and carbon emissions (Liu et al., 2015; Bai and Chen, 2016). These authors were able to find closed-form solutions using derivatives and/or KKT conditions.

DF models are commonly used due to their tractability. However, the assumptions are sometimes unrealistic and can lead to issues with the resulting solutions. Lee et al. (2021) discuss these issues, citing that the main reason for not using a DF approach is that it is unrealistic to assume that any parameters of the distribution are known exactly. In addition, the moment-based ambiguity sets used in DF models can lead to overly conservative decisions (Wang et al., 2016). In practice, typically these parameters must be estimated from historical data, and Lee et al. (2021) state that maximum likelihood estimation approaches can lead to suboptimal solutions.

An example of this comes from Rossi et al. (2014), who used confidence interval analysis for the newsvendor problem under parametric distributions. Instead of singular solutions, their method provides a discrete set of order quantities that contains the true optimal order quantity with a given probability. They found that, even for large samples, the costs given by the maximum likelihood estimates were inaccurate with respect to the true cost. As far as we are aware, this is the only method so far that

allows decision makers to hedge against uncertainty in parameters. However, it may not be practical, since the decision maker then faces the dilemma of which solution to choose. Since it provides a single order quantity to hedge against the worst costs, a DR solution is therefore more convenient for a risk-averse decision maker.

In more recent literature, DR models with more complex ambiguity sets have been developed. Where the DF model's ambiguity set typically contains all distributions with some of their parameters fixed at nominal values, many models have used *distance-based* ambiguity sets. These sets contain all distributions that lie within some pre-prescribed distance from the nominal, according to some distance measure. These are common in the DRO literature, with an example distance measure being *ϕ-divergences* (Bayraksan and Love, 2015). These ambiguity sets lead to problems requiring different treatment than DF problems, and are usually solved via tractable reformulations using Lagrangian dualisation.

One of the first examples of distance-based ambiguity sets for the newsvendor problem was given by Zhao and Guan (2015). They used *ζ-structure probability metrics*, a class of metrics containing the Wasserstein distance, to form their ambiguity sets. Their model was reformulated as a stochastic program, and solved by an iterative sampling algorithm. Gao and Kleywegt (2017a) studied a single-period newsvendor problem with Wasserstein and ϕ-divergence ambiguity sets, which was solved using a convex programming reformulation. The Wasserstein distance was also studied by Mohajerin Esfahani and Kuhn (2018) and Lee et al. (2021), who also found that it led to convex and linear programming reformulations.

Our research differs from the above literature in three key ways. Firstly, we do not assume that any parameters of the distribution are known. In order to avoid the issues resulting from poor estimation, we instead assume that the demand distribution lies in a known parametric family, but that its parameters are completely unknown.

Secondly, we enforce that the worst-case distribution selected by the DRO model also lies within this family, in order to ensure that it is a reasonable candidate for the true distribution. We enforce this via utilising the distribution's probability mass or density function in the objective function directly, and identifying the worst-case parameters instead of the worst-case distribution itself. In particular, we consider normal and Poisson demands. Hence, our model considers nonlinear objective functions, continuous distributions and discrete distributions with infinite support.

This extends the work of Chapter 2, where only binomial distributions and linear objective functions were studied. As was true for the resource planning problem of Chapter 2, parametric ambiguity sets do not lead to convenient reformulations. Therefore, we solve using discretised ambiguity sets. This means that the inner objective can be represented by a finite set of constraints. Since the number of constraints required can be very large, we use theoretical properties of the cost function to develop a fast heuristic CS algorithm for our problem. We show that it performs very well, and that it generates solutions in a much shorter time than solving the full model.

The final way in which our research differs is that we do not use distance measures to build our ambiguity sets. One of the key benefits of doing this, as stated by Lee et al. (2021), is that they allow the ambiguity sets to provide asymptotic probabilis-

tic guarantees. In our methodology, we use maximum likelihood theory to develop confidence sets for the true parameters of the demand distribution. This allows us to provide ambiguity sets with the same probabilistic guarantees, without requiring the use of distance measures.

3.2.2 Multi-period and Capacitated/Budgeted Newsvendor Problems

A natural extension of the classical newsvendor problem of Arrow et al. (1951) is the *multi-period* newsvendor problem. This incorporates either the option to make orders at multiple periods in a planning horizon (*dynamic* problem), or to make orders for multiple periods in a selling horizon prior to the horizon concerned (*static* problem). The earliest multi-period problems are dynamic, where orders are made and delivered in each period. These problems are typically treated as dynamic programs, and can often be solved by *base-stock policies* (Bellman et al., 1955; Bouakiz and Sobel, 1992). Under a base-stock policy, the order quantity for each period is chosen to bring the stock level up to a predefined level.

Much of the literature on multi-period newsvendor problems has considered the dynamic formulation. Some examples include Ahmed et al. (2007), who studied the case where the objective function is a coherent risk measure, Levi et al. (2007), who considered the case where the demand distribution is unknown, and Altintas et al. (2008), who considered setting discounts for a multi-period newsvendor. Later examples considered service-dependent demand (Deng et al., 2014), non-stationary (i.e.

time-dependent) demand (Kim et al., 2015), and two-product models where the total demand is fixed (Zhang and Yang, 2016).

Rather than dynamic, our model is static. Static multi-period models are less common in the literature. Matsuyama (2006) considered a static problem with ordering cycles, but mainly focused on deriving theoretical properties of the optimal profit. Other papers on static multi-period problems typically do not include any constraints on the order quantities, because adding these can result in much more complex KKT conditions. Chen et al. (2017) consider a situation in which a seller sets their price for selling to a multi-period, static newsvendor (buyer). They formulate the buyer's problem as a nonlinear program and solve using the KKT conditions. This solution would be complicated by additional constraints, as we will show later. Ullah et al. (2019) also consider a price-setting scenario, this time through a DF approach under price-dependent demand. This model is unconstrained, and solved using first-order conditions.

Our model extends that of Chen et al. (2017) to the case where the buyer has a monetary budget. This adds a constraint to the model, meaning that standard KKT solutions are not easily applicable. To the best of our knowledge, there are no optimal algorithms in the literature for this problem. In fact, typically, multi-period models do not have capacity or budget constraints. However, such constraints are commonly found in the multi-product literature. In this literature, iterative algorithms are usually used (Lau and Lau, 1996; Abdel-Malek and Montanari, 2005; Alfares and Elmorra, 2005). This is mainly due to the presence of non-negativity constraints in

addition to the budget or capacity constraint. Adding Lagrange multipliers for each non-negativity constraint can make the KKT conditions difficult to solve.

Furthermore, ignoring them can lead to negative order quantities Lau and Lau (1996). Motivated by this, Abdel-Malek and Montanari (2005) developed an iterative algorithm for the multi-product capacitated problem, based on Lagrangian relaxation. Such algorithms are now common ways to solve capacitated problems. To solve our multi-period problem, we adapt another iterative algorithm that was presented by Alfares and Elmorra (2005). This algorithm is designed to solve a multi-product problem with a capacity constraint. It consists of first solving the KKT stationarity condition under no constraints, and then iteratively increasing the Lagrange multiplier from zero until the solution either becomes feasible or negative. If the solution becomes negative, then the corresponding order quantity is set to zero and it is no longer considered. The process then begins again. This allows us to solve the problem without ever explicitly incorporating the non-negativity constraints.

In summary, our research differs from the multi-period literature due to the presence of the budget constraint. Due to the additional complexity caused by this constraint, we use the KKT conditions of the problem to develop an iterative solution algorithm. Similar to the algorithm of Alfares and Elmorra (2005) for the multi-product capacitated problem, our algorithm iteratively increases the Lagrange multiplier for the budget constraint until the order quantities either become feasible or negative. If the latter case occurs first, one order is set to zero and removed from consideration.

However, compared to the multi-product model, the multi-period model has the ad-

ditional complexity that the optimal unconstrained order quantities depend on one another. This makes it unclear which period's order should be set to zero when one becomes negative. Due to this, the proposed algorithm is not optimal in general for our problem. However, we will show that it performs comparably with algorithms from off-the-shelf solvers, and is even faster than these algorithms. More details can be found in Section 3.3.2, where we present and test the algorithm.

3.2.3 Parametric Newsvendor Problems

The final area of the literature that we will review concerns parametric newsvendor problems. Parametric demand distributions allow for two main methodologies for the newsvendor problem. The first approach is the *Bayesian* approach. This consists of first assuming some prior distribution of demand, and then updating the prior distribution to obtain the posterior after some demand is realised. For example, Hill (1997) and Bensoussan et al. (2009) used Bayesian approaches for estimating newsvendor demand under uniform and exponential priors, respectively, in the case of fully observable demand.

The case of censored demand has also been addressed in a Bayesian fashion. For example, Chen (2010) considered a dynamic inventory problem with unobserved data, focussing on developing heuristics and bounds based on Bayesian updating. Mersereau (2015) uses Bayesian estimation to study the effect of inventory record inaccuracy under censored demand. As discussed by Rossi et al. (2014), the Bayesian approach can be inappropriate due to difficulty in selecting a prior and the infinite number

iterations that are required to prove convergence.

The second approach is the *frequentist* approach. This corresponds to authors estimating the parameters of the demand distribution from sample data. Early papers on this approach did not consider integrating the estimates into the model. For example, Nahmias (1994) studied different estimation methods for the parameters of a normal demand distribution in the situation where lost sales are not observed. Agrawal and Smith (1996) studied estimating negative binomial demand from sample data. Both of these studies assumed the order quantity was given. Later papers considered incorporating frequentist estimates into the optimisation models. Liyanage and Shanthikumar (2005) showed that using parameter estimates in place of true parameters can lead to suboptimal solutions. They then introduced the "operational statistics" framework, which integrates estimation and optimisation to reduce bias.

There has also been research on which distributions should be used for newsvendor demand. Gallego et al. (2007) compared the profits obtained from normal, lognormal, gamma and negative binomial distributions in the case where the coefficient of variation was high. They found that the normal distribution can lead to large negative profits, and suggested using non-negative distributions instead. Rossi et al. (2014) proposed a methodology that combines statistical confidence interval analysis with the newsvendor problem to reduce the effect of poor parameter estimation. They provided confidence intervals for the true parameters of the distributions and use this to provide confidence intervals for the true optimal cost. They argue that this avoids the pitfalls of using maximum likelihood estimates, namely poor estimation of optimal

costs. However, it provides multiple solutions, of which one must be selected. Finally, Siegel and Wagner (2021) also presented a methodology to avoid said pitfalls, which is based on an asymptotic adjustment to the estimates in order to reduce bias.

Our methodology utilises maximum likelihood theory for parametric distributions in the newsvendor problem. Motivated by the problems that can occur from assuming that they are the true parameters, we study the effects of assuming that parameter estimates are the true values. We will show that this method performs similarly to how it did for Rossi et al. (2014), namely that it leads to suboptimality and poor cost estimates. In addition, we use multivariate confidence theory to develop confidence sets for the true parameters. These are then used as ambiguity sets in our models. Although our confidence sets are based on asymptotic theory, they are large for small samples and small for large samples. This means that the model actively accounts for small sample sizes through constructing a more conservative set.

3.3 Fixed Distribution Model and Solution

In Section 3.3.1, we present the model in the case where the demand distribution is known. Then, in Section 3.3.2, we present and test our algorithm for solving this problem. This algorithm will be used to benchmark the DRO methods that are the main focus of this chapter.

3.3.1 Model

Adapting the model by Chen et al. (2017), we assume that there are T periods for which an order needs to be made. We denote the periods by $t \in \mathcal{T} = \{1, \ldots, T\}$. The *order quantity* vector is denoted $\boldsymbol{q} = (q_1, \ldots, q_T)$, and gives the amount of stock to order for each period $t \in \mathcal{T}$. The demand for period t is denoted by X_t. We also assume that the newsvendor has a total *budget* for expenditure of W, that limits the amount that can be ordered in total. In practice, we do not have access to infinite resources and hence this is a realistic addition to the model. We denote by F the joint cumulative distribution function (CDF) of the demand vector $\boldsymbol{X}$.

Furthermore, let I_t be the *inventory* available during period $t \in \{1, \ldots, T\}$ and assume $I_0 = 0$ w.l.o.g. Finally, let us define the costs for the model. Let w_t be the cost of ordering a unit of stock to sell on period t (*ordering cost*). We will assume that $w_t \geq w_{t+1} \; \forall \, t = 1, \ldots, T$. This is reasonable because, as shown by Chen et al. (2017), any optimal set of prices for the supplier satisfies this condition. Let h be the cost of holding one unit of stock for one period (i.e. a *holding cost*), and let b be the cost of one unit of unmet demand (*backorder cost*). Finally, let c be the price a customer pays to purchase stock from the newsvendor (*purchasing cost*). Then, in each period t the following events occur:

1. The order of size q_t arrives and the supplier pays $w_t q_t$ for the shipment.

2. Demand X_t is realised.

3. Inventory at end of period is calculated as $I_t = I_{t-1} + q_t - X_t$.

4. If $t \leq T-1$ then the newsvendor receives profit $cX_t - hI_t^+ - bI_t^-$. Here $I_t^+ = \max\{I_t, 0\}$ is the leftover product that must be held for the following period and $I_t^- = \max\{-I_t, 0\}$ is the unmet demand. This profit is the revenue from sales, minus the total holding costs and backorder costs. It assumes that all demand for period t is met at some point during or after period t. If it cannot be met, this is not known until $t = T$, where the profit is reduced accordingly.

   If $t = T$ then the profit is $c\min\{X_T, I_{T-1} + q_T\} - hI_T^+ - bI_T^-$. The first term in this expression represents that demand in period T can only be met in period T, since there are no subsequent periods. Hence, it can only be met if the newsvendor has enough stock to meet it in period T.

Given this, the expected cost for an ordering decision $\boldsymbol{q}$ is given by:

$$C_F(\boldsymbol{q}) = \sum_{t=1}^{T} \left(h\mathbb{E}_F[I_t^+] + b\mathbb{E}_F[I_t^-] + w_t q_t \right) - c\left(\mathbb{E}_F\left[\sum_{t=1}^{T} X_t \right] - \mathbb{E}_F[I_T^-] \right),$$

which represents the total expected costs incurred from purchasing, holding and backordering, minus any revenue made. The final term is the expected revenue made from sales, where $\mathbb{E}_F\left[\sum_{t=1}^{T} X_t\right]$ is the expected total demand and $\mathbb{E}_F[I_T^-]$ is the expected number of missed sales, which corresponds to the unmet demand in period T. Then, the model for a fixed distribution F, referred to as *MPNVP*, can be written as:

$$\min_{\boldsymbol{q}} C_F(\boldsymbol{q}) = c\mathbb{E}_F[I_T^-] + \sum_{t=1}^{T} \left(h\mathbb{E}_F[I_t^+] + b\mathbb{E}_F[I_t^-] + w_t q_t - c\mathbb{E}_F[X_t] \right) \tag{3.3.1}$$

$$\text{s.t.} \quad \sum_{t=1}^{T} w_t q_t \leq W, \tag{3.3.2}$$

$$I_t = I_{t-1} + q_t - X_t \;\forall\; t = 1, \ldots, T, \tag{3.3.3}$$

$$q_t \geq 0 \;\forall\; t = 1, \ldots, T. \tag{3.3.4}$$

Constraint (3.3.2) is the budget constraint and (3.3.3) is the inventory constraint.

3.3.2 Solution Procedure

A Solution of the KKT Stationarity Condition

In order to develop a solution algorithm for this problem, we will now assume that F represents a continuous distribution. This allows us to use the KKT conditions to derive an algorithm. The resulting algorithm iteratively increases the Lagrange multiplier for the budget constraint, each time generating a new solution by evaluating the inverse CDF of the demand distribution. Since the inverse CDF of a discrete distribution will always provide integer order quantities, the algorithm we develop can also be applied to discrete distributions. We will show that it performs well for both the Poisson and normal distributions.

Define a Lagrange multiplier ν for the budget constraint. Then, the Lagrangian of MPNVP without considering the non-negativity constraints is defined as:

$$L(\boldsymbol{q}, \nu) = C_F(\boldsymbol{q}) + \nu \left(\sum_{t=1}^{T} w_t q_t - W \right).$$

The KKT conditions for a solution $\boldsymbol{q}^*$ to be optimal are therefore:

$$\frac{\partial}{\partial q_j} C_F(\boldsymbol{q}^*) + \nu w_j = 0 \ \forall \ j = 1, \ldots, T, \tag{3.3.5}$$

$$\sum_{t=1}^{T} w_t q_t^* \leq W, \tag{3.3.6}$$

$$\nu \left(\sum_{t=1}^{T} w_t q_t^* - W \right) = 0, \tag{3.3.7}$$

$$\nu \geq 0. \tag{3.3.8}$$

Our algorithm will make use of Theorem 3.3.1, which finds a solution satisfying the stationarity condition (3.3.5):

Theorem 3.3.1. *For a given Lagrange multiplier ν, suppose that the following two conditions hold:*

1. $\nu \leq \frac{b}{w_t - w_{t+1}} - 1 \ \forall \ t \in \{1, \ldots, T-1\}$,

2. $\nu \leq \frac{b+c}{w_T} - 1$.

and let $\tilde{F}_t$ be the CDF of $\sum_{l=1}^{t} X_l$. Then, the solution:

$$\left.\begin{aligned} q_1^* &= \tilde{F}_1^{-1}\left(\frac{b-(1+\nu)(w_1-w_2)}{h+b}\right) \\ q_t^* &= \tilde{F}_t^{-1}\left(\frac{b-(1+\nu)(w_t-w_{t+1})}{h+b}\right) - \tilde{F}_{t-1}^{-1}\left(\frac{b-(1+\nu)(w_{t-1}-w_t)}{h+b}\right), (2 \leq t \leq T-1) \\ q_T^* &= \tilde{F}_T^{-1}\left(\frac{b-(1+\nu)w_T+c}{h+b+c}\right) - \tilde{F}_{T-1}^{-1}\left(\frac{b-(1+\nu)(w_{T-1}-w_T)}{h+b}\right). \end{aligned}\right\} \tag{3.3.9}$$

satisfies the stationarity KKT condition (3.3.5) *of MPNVP.*

This theorem is proved in Appendix B.1. Even though this solution satisfies the stationarity condition of MPNVP, it may not be feasible. We also require that $q_t \geq 0 \ \forall \ t$ and $\sum_{t=1}^{T} w_t q_t \leq W$. Adding Lagrange multipliers for every non-negativity constraint results in an un-solvable set of KKT conditions, as discussed by Alfares and Elmorra (2005); Abdel-Malek and Montanari (2005) and Lau and Lau (1996), who each studied multi-product newsvendor models with budget constraints. Due to this, along with the fact that relaxing non-negativity constraints can lead to negative order quantities, the common approach among these papers has been to develop iterative algorithms that do not require that the non-negativity constraints are enforced directly.

An Iterative Solution Algorithm: FD

We now present our algorithm for solving MPNVP. We will refer to this algorithm as *fixed distribution* (FD). Details on deriving FD can be found in Appendix B.2.1. In essence, FD starts with $\nu = 0$ and iteratively increases ν until either the stationarity solution in (3.3.9) becomes feasible or an order becomes negative. If an order becomes negative, it selects an order to set to zero and remove from consideration, and then starts again. However, once an order has been set to zero, the solution in (3.3.9) can no longer be used since it does not enforce this.

Let us denote by $\mathcal{T}^0$ the set of days that have had their orders set to zero and have been removed from consideration. When $\mathcal{T}^0$ is non-empty, instead of using (3.3.9), we need to generate a solution that satisfies (3.3.5) and also has $q_t = 0 \ \forall \ t \in \mathcal{T}^0$. In Appendix B.2.1, we show that one solution that satisfies both of these requirements is given by $\tilde{\boldsymbol{q}}^*$, as defined in (3.3.10).

$$\left.\begin{aligned}
\tilde{q}_1^* &= \mathbb{1}\{1 \notin \mathcal{T}^0\}\tilde{F}_1^{-1}\left(\frac{b-(1+\nu)(w_1-w_2)}{h+b}\right) \\
\tilde{q}_t^* &= \mathbb{1}\{t \notin \mathcal{T}^0\}\left(\tilde{F}_t^{-1}\left(\frac{b-(1+\nu)(w_t-w_{t+1})}{h+b}\right) - \sum_{l=1}^{t-1}\tilde{q}_l^*\right) \ \ \forall\ t \in \{2,\ldots,T-1\}. \\
\tilde{q}_T^* &= \mathbb{1}\{T \notin \mathcal{T}^0\}\left(\tilde{F}_T^{-1}\left(\frac{b-(1+\nu)w_T+c}{h+b+c}\right) - \sum_{l=1}^{T-1}\tilde{q}_l^*\right)
\end{aligned}\right\} \tag{3.3.10}$$

Let $\boldsymbol{q}^*(\nu)$ be the solution obtained from evaluating (3.3.9) with Lagrange multiplier ν. Similarly, define $\tilde{\boldsymbol{q}}^*(\nu)$ as the solution obtained from evaluating (3.3.10) with Lagrange multiplier ν. Then, FD can be described as follows:

1. Set $\mathcal{T}^0 = \{\}$.

2. Take $\boldsymbol{q} = (\max\{\tilde{q}_1^*(0), 0\}, \ldots, \max\{\tilde{q}_T^*(0), 0\})$, i.e. $\tilde{\boldsymbol{q}}^*(0)$ but setting negative values to zero.

3. If $\sum_{t=1}^{T} w_t q_t \leq W$ then go to step 7. If $\sum_{t=1}^{T} w_t q_t > W$ then if $\mathcal{T}^0 = \{\}$ go to step 4 and otherwise go to step 5.

4. The budget constraint is binding. Compute upper bounds on ν, i.e.

$$\nu^{\text{UBs}} = \left\{ \frac{b}{w_t - w_{t+1}} - 1 \;\forall\; t \in \{1, \ldots, T-1\} \right\} \cup \left\{ \frac{b+c}{w_T} - 1 \right\}$$

and set $\nu^{\text{UB}} = \min\{\nu^{\text{UBs}}\}$. Then $\nu \leq \nu^{\text{UB}}$ ensures that the inverse CDFs are all defined at ν. Go to step 5.

5. Starting from $\nu = 0$, increase ν until the first occurrence of one of the following:

   (a) The resulting $\tilde{\boldsymbol{q}}^*(\nu)$ has $\tilde{q}_l^*(\nu) < 0$ for some l. In this case define $\nu^* = \nu$, set $\boldsymbol{q} = \tilde{\boldsymbol{q}}^*(\nu)$ and go to step 6.

   (b) The resulting $\tilde{\boldsymbol{q}}^*(\nu)$ has $\sum_{t=1}^{T} w_t \tilde{q}_t^*(\nu) \leq W$. In this case go to step 7.

6. Select which day to add to $\mathcal{T}^0$:

   (a) For each day $t \in \mathcal{T} \setminus \mathcal{T}^0$, calculate a lower bound on the cost resulting from setting $q_t = 0$ and adding t to $\mathcal{T}^0$. Specifically, calculate $\tilde{\boldsymbol{q}}^*(\nu^*)$ with $\mathcal{T}^0 = \mathcal{T}^0 \cup \{t\}$.

   (b) Select l to be the day with the lowest lower bound resulting from adding it to $\mathcal{T}^0$.

   (c) Set $\mathcal{T}^0 = \mathcal{T}^0 \cup \{l\}$ and go to step 2.

7. Return $\tilde{\boldsymbol{q}}^*(\nu)$.

The algorithm uses a line search on $[0, \nu^{\text{UB}}]$ to find the optimal value of ν in Step 5. More details on step 5 can be found in Appendix B.2.2. The reason why FD is only a heuristic is step 6. The idea behind step 6 is that, unlike in the multi-product model, the order quantity that becomes negative may not be the one that should be set to zero. This is because, in the multi-period model, setting a negative order to zero reduces subsequent orders.

In order to select which order quantity should be set to zero, we estimate the effect of each choice by finding a lower bound on the resulting cost. For each $t \in \mathcal{T} \setminus \mathcal{T}^0$, we calculate this lower bound by temporarily adding t to $\mathcal{T}^0$ and evaluating the cost of the resulting $\tilde{\boldsymbol{q}}^*(\nu)$. The reason for evaluating the next decision at ν^* is as follows. Since step 5(a) caused us to proceed to step 6, we know that a negative order occurs before the budget constraint is met. Since period t's order was negative, if this order is set to zero then the budget constraint will still not be met (otherwise 5(b) would have occurred first). Hence, the remaining orders must be reduced further, i.e. we must have $\nu \geq \nu^*$ if $\tilde{\boldsymbol{q}}^*(\nu)$ is to meet the budget constraint. Since ν has to be increased further to enforce the budget constraint, the cost at optimality must be higher than the cost of $\tilde{\boldsymbol{q}}^*(\nu)$. Hence, the cost of this solution gives a lower bound on the cost of the solution where t has been added to $\mathcal{T}^0$. Given all lower bounds, we then select t with the lowest lower bound and add it to $\mathcal{T}^0$.

Testing FD

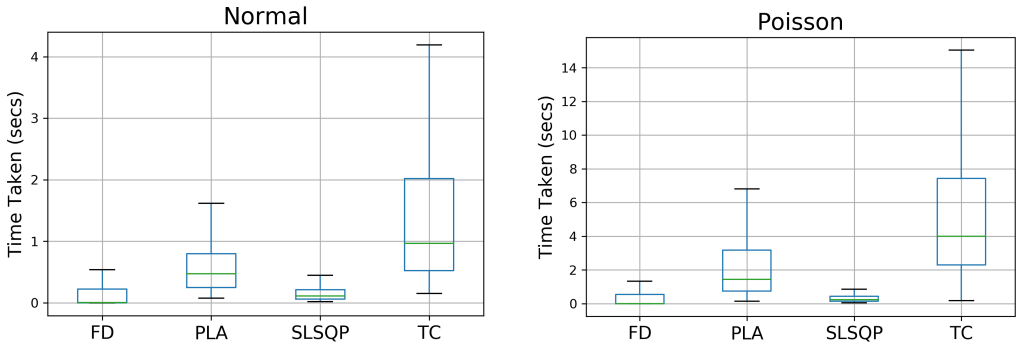
In order to establish the performance of FD, we tested it on 1536 problem instances. We obtained these by considering problems with $T \in \{2, \ldots, 5\}$, $c, h, b \in \{1, 2\}$,

$W \in \{10, 25, 50\}$, $w \in \{(2T, 2(T-1), \ldots, 2), (1, \frac{1}{2}, \ldots, \frac{1}{T})\}$. In addition, we selected 2 mean vectors randomly for each T and set the standard deviation vector to be $\boldsymbol{\sigma} = \frac{\boldsymbol{\mu}}{4}$ in each case. This only affects the normal experiments. Furthermore, the tolerance for the budget constraint was either 0 or 10^{-6}. The tuning parameter τ for the line search used by FD was tested for $\tau \in \{0.25, 0.5\}$. This generated 768 instances for normal and 768 instances for Poisson demands.

Since our algorithm is a heuristic, we compare it with 3 off-the-shelf algorithms that might be applied to solve MPNVP. The first two algorithms are *Sequential Least Squares Quadratic Programming* (SLSQP) (Kraft, 1988) and *Trust Constraint* (TC) Conn et al. (2000) from the Python library Scipy (Virtanen et al., 2020). For more details on these algorithms, see Appendix B.2.3. The final algorithm is *Piecewise Linear Approximation* (PLA), an algorithm that uses Gurobi (Gurobi Optimization, LLC, 2022) to solve a piecewise linear approximation of MPNVP.

We now summarise the results of our experiments. Figure 3.3.1 shows the times taken by each algorithm to solve MPNVP under normal and Poisson demands, with outliers removed. Points that are more than 1.5 times the IQR above the 75th percentile or below the 25th percentile are treated as outliers. Note that this is the case for all boxplots in this chapter, and it is the standard metric for defining outliers in boxplots. The plots show that FD and SLSQP are the fastest algorithms for both normal and Poisson demands. On average, for normal demands, FD took 0.14 seconds and SLSQP took 0.15 seconds. For Poisson demands, FD took 0.23 seconds whereas SLSQP took 0.32 seconds, on average. Hence, our results suggest that FD is the fastest algorithm

on average.



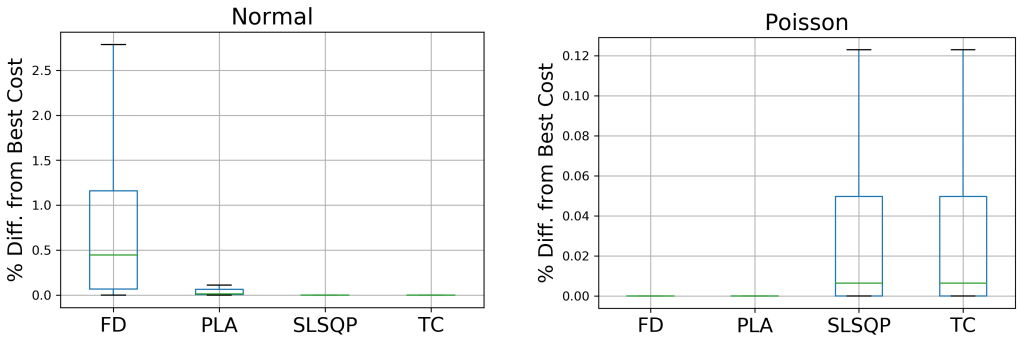
(a) (b)

Figure 3.3.1: Boxplots summarising times taken by the 4 algorithms

To understand the performance of the algorithms, we define a percentage gap from the best solutions as follows:

$$100 \times \frac{C_F(\boldsymbol{q}^a) - \min_{y \in \mathcal{Y}} C_F(\boldsymbol{q}^y)}{\min_{y \in \mathcal{Y}} C_F(\boldsymbol{q}^y)},$$

where $a, y \in \mathcal{Y} = \{\text{FD}, \text{PLA}, \text{SLSQP}, \text{TC}\}$ represents an algorithm, and $\boldsymbol{q}^y$ is the solution returned by algorithm y. Given this definition, we summarise the gaps in Figure 3.3.2. This plot shows that FD is typically within 2.5% of the best solution for normal demands. SLSQP and TC perform the best here. However, for Poisson demands, FD and PLA are the best algorithms in terms of objective value. FD had the best objective value in 84% of Poisson instances. On average over all instances, PLA is the best performing algorithm. It is important to mention that, while SLSQP performs well for Poisson demands, in 678 out of 768 instances with Poisson demands, its solution was not integer. There is no way to enforce integer variables within this algorithm. Hence, FD and PLA have this as an advantage over SLSQP: if the distribution is discrete, their solutions are integer.



(a) (b) s

Figure 3.3.2: Boxplots summarising percentage optimality gaps of the 4 algorithms

The final result we would like to mention is the exceedance of the budget. One key difference between SLSQP and TC is that TC allows us to specify a tolerance for exceeding the budget, where SLSQP does not. We recorded the number of times each algorithm exceeded the budget by more than the tolerance, and by how much. We find that SLSQP exceeded the tolerance in 206 instances, with a maximum exceedance of 6.58×10^{-6}. FD was the only algorithm that never exceeded the tolerance.

From these results, we can conclude four main points. Firstly, FD was the fastest algorithm of those that we tested. Secondly, FD was typically close to the best solution on average, and performed better than SLSQP and TC for Poisson demands. Thirdly, SLSQP and TC do not return integer solutions for the Poisson distribution in general. There is no way to enforce that these algorithms return integer solutions. Despite FD being designed for continuous order quantities, when the demand is discrete, its solution will always be integer. This is due to its solution being found from evaluating the inverse CDF of the demand distribution. This is an important point, as when demand is discrete the order quantity should also be. Finally, FD is the only algorithm

that can be guaranteed not to exceed the budget constraint by more than some user-specified tolerance.

3.4 DRO Model with Normal Demands

In this section, we describe and solve the DRO model under normal demands. We first detail the model and ambiguity sets used in Section 3.4.1. Following this, in Section 3.4.2, we develop a set of extreme distributions based on theoretical results concerning the objective function. In Section 3.4.3, we use these results to develop our CS algorithm. Finally, in Section 3.4.4, we perform extensive computational experiments to test the MLE approach and the CS algorithm.

3.4.1 Formulation and Ambiguity Sets

The first parametric DRO model that we study is the one resulting from independent normal demands. Suppose we have a set Θ of pairs $\boldsymbol{\theta} = (\boldsymbol{\mu}, \boldsymbol{\sigma})$ such that each $\boldsymbol{\theta}$ gives a unique multivariate normal distribution. This means there is a unique CDF F for each $\boldsymbol{\theta}$ and so we can replace F with $\boldsymbol{\theta}$. Then the DRO model is:

$$\min_{\boldsymbol{q}} \max_{\boldsymbol{\theta}\in\Theta} C_{\boldsymbol{\theta}}(\boldsymbol{q}) \tag{3.4.1}$$

$$\text{s.t.} \sum_{t=1}^{T} w_t q_t \leq W \tag{3.4.2}$$

$$\boldsymbol{q} \in \mathbb{R}_+^T \tag{3.4.3}$$

Let $a_t = b + c\mathbb{1}\{t = T\} + h$, and let ϕ and Φ denote the PDF and CDF of the standard normal distribution respectively. Lemma 3.4.1 allows us to rewrite the objective

function in a more convenient form.

Lemma 3.4.1. *When $X_t \sim \mathcal{N}(\mu_t, \sigma_t^2)$ for $t = 1, \ldots, T$ and the X_t's are independent, we can rewrite $C_{\boldsymbol{\theta}}(\boldsymbol{q})$ as:*

$$C_{\boldsymbol{\theta}}(\boldsymbol{q}) = \sum_{t=1}^{T} \left\{ (a_t \Phi(\beta_t) - h) \sum_{l=1}^{t} (\mu_l - q_l) + a_t \phi(\beta_t) \sqrt{\sum_{l=1}^{t} \sigma_l^2} + q_t w_t - c\mu_t \right\}, \quad (3.4.4)$$

with $\beta_t = \frac{\sum_{l=1}^{t}(\mu_l - q_l)}{\sqrt{\sum_{l=1}^{t} \sigma_l^2}}$.

The proof of Lemma 3.4.1 can be found in Appendix B.3.1. Given this formulation, we next decide on the form that our ambiguity set Θ will take. As is common in the DRO literature, we will use maximum likelihood estimation to derive an approximate $100(1-\alpha)\%$ confidence set for the true parameter vector, which we will denote by $\boldsymbol{\theta}^0 = (\boldsymbol{\mu}^0, \boldsymbol{\sigma}^0)$. Suppose for each $t = 1, \ldots, T$ that we have access to a sample $\boldsymbol{x}_t = (x_t^1, \ldots, x_t^N)$ of size N from the true demand distribution $\mathcal{N}(\mu_t^0, (\sigma_t^0)^2)$. We can use these samples to create MLEs of the true parameters, $\hat{\boldsymbol{\mu}} = (\hat{\mu}_1, \ldots, \hat{\mu}_T)$ and $\hat{\boldsymbol{\sigma}} = (\hat{\sigma}_1, \ldots, \hat{\sigma}_T)$. We also define $\hat{\boldsymbol{\theta}}$ as the MLE of the true parameter vector $\boldsymbol{\theta}^0$, i.e. $\hat{\boldsymbol{\theta}} = (\hat{\boldsymbol{\mu}}, \hat{\boldsymbol{\sigma}})$. It is well-known that the MLEs for normal parameters are given by:

$$\hat{\mu}_t = \frac{1}{N} \sum_{n=1}^{N} x_t^n, \quad \hat{\sigma}_t = \sqrt{\frac{1}{N} \sum_{n=1}^{N} (x_t^n - \hat{\mu}_t)^2} \quad (t = 1, \ldots, T).$$

Given these estimates, we can construct an approximate confidence set using Proposition 3.4.2. This result is proved in Appendix B.3.2.

Proposition 3.4.2. *Suppose that $X_t \sim \mathcal{N}(\mu_t^0, (\sigma_t^0)^2)$ for $t = 1, \ldots, T$, where $\boldsymbol{\theta}^0 = (\boldsymbol{\mu}^0, \boldsymbol{\sigma}^0)$ is unknown and the X_t's are independent. Furthermore, suppose that $\hat{\mu}_t$ and $\hat{\sigma}_t$ are MLEs obtained from N samples from the distribution of X_t, for $t = 1, \ldots, T$.*

Then, an approximate $100(1-\alpha)\%$ *confidence set for* $\boldsymbol{\theta}^0$ *is given by:*

$$\Theta_\alpha = \left\{ (\boldsymbol{\mu}, \boldsymbol{\sigma}) \in \mathbb{R}_+^{2T} : \sum_{t=1}^{T} \left(\frac{N}{\hat{\sigma}_t^2} \left(\hat{\mu}_t - \mu_t \right)^2 + \frac{2N}{\hat{\sigma}_t^2} \left(\hat{\sigma}_t - \sigma_t \right)^2 \right) \leq \chi^2_{2T,1-\alpha} \right\}. \quad (3.4.5)$$

In order to solve our model, we assume a discrete ambiguity set. This means that the inner objective can be represented by a finite number of constraints. As described in Chapter 2, parametric ambiguity sets do not yield convenient reformulations in any other way. Hence, in this chapter, we will work with discrete subsets of the set in (3.4.5). Since it is difficult to discretise this set directly, we will first construct a baseline ambiguity set Θ_{base} such that $\Theta_\alpha \subseteq \Theta_{\text{base}}$ and discretise this set. Then, we build a discretisation of Θ_α by taking all elements of the discretisation of Θ_{base} that also lie in Θ_α. A logical way in which to construct Θ_{base} is as follows. The definition of (3.4.5) implies that any $(\boldsymbol{\mu}, \boldsymbol{\sigma}) \in \Theta_\alpha$ satisfies:

$$\mu_t \in \mu_t^{\text{int}} = \left[\hat{\mu}_t - \hat{\sigma}_t \sqrt{\frac{\chi^2_{2T,1-\alpha}}{N}}, \hat{\mu}_t + \hat{\sigma}_t \sqrt{\frac{\chi^2_{2T,1-\alpha}}{N}} \right], \ \forall\, t \in \{1, \ldots, T\}$$

$$\sigma_t \in \sigma_t^{\text{int}} = \left[\hat{\sigma}_t - \hat{\sigma}_t \sqrt{\frac{\chi^2_{2T,1-\alpha}}{2N}}, \hat{\sigma}_t + \hat{\sigma}_t \sqrt{\frac{\chi^2_{2T,1-\alpha}}{2N}} \right] \ \forall\, t \in \{1, \ldots, T\}.$$

Therefore, defining Θ_{base} using (3.4.6), we have $\Theta_\alpha \subseteq \Theta_{\text{base}}$.

$$\Theta_{\text{base}} = \left\{ (\boldsymbol{\mu}, \boldsymbol{\sigma}) : \mu_t \in \mu_t^{\text{int}} \wedge \sigma_t \in \sigma_t^{\text{int}} \ \forall\, t = 1, \ldots, T \right\}. \quad (3.4.6)$$

Let us refer to the upper and lower boundaries of the individual intervals as μ_t^l, μ_t^u and σ_t^l, σ_t^u for each $t = 1, \ldots, T$. Then, we can create discretisations of these intervals containing M points using:

$$\tilde{\mu}_t^{\text{int}} = \left\{ \mu_t^l + m \frac{\mu_t^u - \mu_t^l}{M-1} : m = 0, \ldots, M-1 \right\} \ \forall\, t \in \mathcal{T}$$

$$\tilde{\sigma}_t^{\text{int}} = \left\{ \sigma_t^l + m\frac{\sigma_t^u - \sigma_t^l}{M-1} : m = 0, \ldots, M-1 \right\} \ \forall\ t \in \mathcal{T}.$$

Then we can construct a discretisation of the baseline ambiguity set via:

$$\Theta'_{\text{base}} = \left\{ (\boldsymbol{\mu}, \boldsymbol{\sigma}) : \mu_t \in \tilde{\mu}_t^{\text{int}} \wedge \sigma_t \in \tilde{\sigma}_t^{\text{int}} \ \forall\ t = 1, \ldots, T \right\}.$$

Then, a discretisation of Θ_α is given by $\Theta'_\alpha = \Theta'_{\text{base}} \cap \Theta_\alpha$. This is the set that we will use in our experiments. For the discrete ambiguity set Θ'_α, we can reformulate the normal DRO model (3.4.1)-(3.4.3) as:

$$\min \left\{ \vartheta : \vartheta \leq C_{\boldsymbol{\theta}}(\boldsymbol{q}) \ \forall\ \boldsymbol{\theta} \in \Theta'_\alpha, \sum_{t=1}^{T} w_t q_t \leq W, \boldsymbol{q} \in \mathbb{R}_+^T \right\}. \tag{3.4.7}$$

The first constraint here is implemented in Gurobi using piecewise linear approximations. For details on the piecewise linear reformulation of this model, see Appendix B.3.3.

3.4.2 Extreme and Dominated Distributions

Depending on the size of Θ'_α, the piecewise linear model can become very slow. This is due to the fact that one piecewise linear approximation is required for every $\boldsymbol{\theta} \in \Theta'_\alpha$ and every $t \in \{1, \ldots, T\}$. The resulting model can have a significant number of constraints. In order to generate solutions in a shorter time, we will develop a CS algorithm. The idea of this algorithm is to iteratively solve the model on a small subset of Θ'_α, and after each iteration generate a new distribution to add to the previous subset. Since finding the worst-case parameter for a given $\boldsymbol{q}$ can be cumbersome when Θ'_α is large, we will create a set Θ^{ext} of *extreme* parameters that are close to

the worst case. In order to create this set, we derive the following properties of the objective function. These results are proved in Appendix B.3.4.

Theorem 3.4.3. *When $X_t \sim \mathcal{N}(\mu_t, \sigma_t^2)$ for $t = 1, \ldots, T$ and the X_t's are independent, $C_{\boldsymbol{\theta}}(\boldsymbol{q})$ is:*

1. *Convex in μ_t for all $t \in \{1, \ldots, T\}$,*
2. *Increasing in σ_t for all $t \in \{1, \ldots, T\}$.*

Given these results, the question then arises of how to combine them in order to find the most extreme $\boldsymbol{\theta}$ values. Let us define the sets of all distinct $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ vectors as $\mathcal{M}$ and $\mathcal{S}$:

$$\mathcal{M} = \left\{ \boldsymbol{\mu} \in \mathbb{R}_+^T \mid \exists\ \boldsymbol{\sigma} \in \mathbb{R}_+^T : (\boldsymbol{\mu}, \boldsymbol{\sigma}) \in \Theta'_\alpha \right\}$$

$$\mathcal{S} = \left\{ \boldsymbol{\sigma} \in \mathbb{R}_+^T \mid \exists\ \boldsymbol{\mu} \in \mathbb{R}_+^T : (\boldsymbol{\mu}, \boldsymbol{\sigma}) \in \Theta'_\alpha \right\}.$$

From $\mathcal{M}$ and $\mathcal{S}$ we can extract the most extreme $\boldsymbol{\mu}$'s and the most extreme $\boldsymbol{\sigma}$'s individually using Theorem 3.4.3. We first consider characterising the worst-case $\boldsymbol{\sigma}$. Since the cost function is increasing in each σ_t, it may seem reasonable that the worst-case $\boldsymbol{\theta}$ will have at least one σ_t set at its maximum. However, if one σ_t is at its maximum, then the others may need to be closer to $\hat{\sigma}_t$ in order for $\boldsymbol{\sigma}$ to remain sufficiently close to $\hat{\boldsymbol{\sigma}}$. In other words, maximising one σ_t may mean reducing the others, and this can yield a lower expected cost than if multiple standard deviations were allowed to be moderately large. In order to account for this fact, we consider $\boldsymbol{\sigma}$'s with the largest sum. This allows us to avoid prioritising any one day in particular and instead ensure that there is maximal total variability over all days.

We now discuss the worst-case $\boldsymbol{\mu}$. Since the cost function is convex in each μ_t, we will consider $\boldsymbol{\mu}$ such that at least one μ_t is equal to its maximum or minimum value. This is similar to the idea of maximising one σ_t, and therefore has the same problems. However, it is possible that some μ_t's should be maximised and others minimised. Therefore, maximising or minimising $\sum_{t=1}^{T} \mu_t$ is not logical here, since it implies that all should be maximised or all should be minimised.

In addition, it may be the case that any $(\boldsymbol{\mu}, \boldsymbol{\sigma}) \in \Theta'_\alpha$ that has $\sum_{t=1}^{T} \sigma_t$ maximised does not have any μ_t at its maximum or minimum. Similarly, any $(\boldsymbol{\mu}, \boldsymbol{\sigma})$ such that some μ_t is maximised or minimised may not have $\sum_{t=1}^{T} \sigma_t$ being maximised. Therefore, in order to construct a set of extreme parameters, we use a two-step procedure. For each $\boldsymbol{\mu} \in \mathcal{M}$ we can find the $\boldsymbol{\sigma}$'s with the largest sum over all $\boldsymbol{\theta} \in \Theta'_\alpha$ whose mean is equal to $\boldsymbol{\mu}$. This corresponds to finding the most extreme distributions for this fixed $\boldsymbol{\mu}$. Hence, we define the set:

$$\Theta_1^{\text{ext}} = \left\{ (\boldsymbol{\mu}, \boldsymbol{\sigma}) \in \Theta'_\alpha : \sum_{t=1}^{T} \sigma_t = \max_{(\boldsymbol{\mu}, \tilde{\boldsymbol{\sigma}}) \in \Theta'_\alpha} \sum_{t=1}^{T} \tilde{\sigma}_t \right\}.$$

Given this set, we now eliminate pairs from Θ_1^{ext} where $\boldsymbol{\mu}$ is dominated when the corresponding $\boldsymbol{\sigma}$ is fixed. We can characterise this by finding the maximum and minimum for each μ_t conditional on $\boldsymbol{\sigma}$. Then, we can remove any pairs from Θ_1^{ext} that do not have any μ_t equal to their maximum or minimum. Therefore, we define:

$$\mu_t^{\max}(\boldsymbol{\sigma}) = \max_{(\boldsymbol{\mu}, \boldsymbol{\sigma}) \in \Theta_1^{\text{ext}}} \mu_t, \ \mu_t^{\min}(\boldsymbol{\sigma}) = \min_{(\boldsymbol{\mu}, \boldsymbol{\sigma}) \in \Theta_1^{\text{ext}}} \mu_t,$$

$$\Theta_2^{\text{ext}} = \left\{ (\boldsymbol{\mu}, \boldsymbol{\sigma}) \in \Theta_1^{\text{ext}} \mid \exists t \in \mathcal{T} : \mu_t \in \left\{ \mu_t^{\min}(\boldsymbol{\sigma}), \mu_t^{\max}(\boldsymbol{\sigma}) \right\} \right\}.$$

We can write this in a simpler form as a single set using:

$$\Theta^{\text{ext}} = \left\{ (\boldsymbol{\mu}, \boldsymbol{\sigma}) \in \Theta'_\alpha \;\middle|\; \left(\sum_{t=1}^{T} \sigma_t = \max_{(\boldsymbol{\mu}, \tilde{\boldsymbol{\sigma}}) \in \Theta'_\alpha} \sum_{t=1}^{T} \tilde{\sigma}_t \right) \wedge \left(\exists t \in \mathcal{T} : \mu_t \in \{ \mu_t^{\min}(\boldsymbol{\sigma}), \mu_t^{\max}(\boldsymbol{\sigma}) \} \right) \right\} \tag{3.4.8}$$

Note that this set may not contain the true worst-case $\boldsymbol{\theta} \in \Theta'_\alpha$, since it may be the case that selecting a less extreme mean/standard deviation in favour of a more extreme standard deviation/mean may lead to a worse distribution. However, in Section 3.4.4, we will show that the worst-case in Θ^{ext} is typically very close to the true worst-case over the entire ambiguity set Θ'_α.

The size of Θ'_α can be too large for the model to be solved by PLA in a reasonable time. In addition to constructing Θ^{ext}, Theorem 3.4.3 allows us to remove some dominated distributions that can never be the worst case. Consider two parameter values $\boldsymbol{\theta}^1$ and $\boldsymbol{\theta}^2$ such that $\boldsymbol{\mu}^1 = \boldsymbol{\mu}^2$, there is at least one t such that $(\sigma^1)_t > (\sigma^2)_t$, and $(\sigma^1)_t \geq (\sigma^2)_t \; \forall \; t$. Since, for a fixed μ, the objective function is increasing in each σ_t, this means that $\boldsymbol{\theta}^2$ is *dominated* by $\boldsymbol{\theta}^1$. Hence, $\boldsymbol{\theta}^2$ can never be a worse parameter than $\boldsymbol{\theta}^1$. Since this parameter can never be the worst-case, it can be removed from Θ'_α and the solution to the model will not change. In our experiments, in order to reduce the computational burden for PLA, we will remove all dominated distributions prior to solving. This can greatly reduce the number of model constraints.

3.4.3 Cutting Surface Algorithm

We now present our CS algorithm. The idea of the algorithm is as follows. We start with some initial singleton ambiguity set Θ^1. Then, for $k = 1, \ldots, k^{\max}$, we solve the

piecewise linear approximation of the DRO model in (3.4.7) over Θ^k to get a solution $\boldsymbol{q}^k$. Then, we find the worst-case parameter $\boldsymbol{\theta}^k$ for the solution $\boldsymbol{q}^k$ over the set Θ^{ext} defined in (3.4.8). We then set $\Theta^{k+1} = \Theta^k \cup \{\boldsymbol{\theta}^k\}$ and solve again. This process repeats until stopping criteria are met. The CS algorithm for this problem is as follows.

1. Initialise by choosing initial set of one distribution $\Theta^1 = \{\boldsymbol{\theta}^{\text{init}}\} \subseteq \Theta'_\alpha$, optimality tolerance ϵ, $k^{\max}$, and the gap ε between successive z points for the piecewise linear approximation.

2. Construct set of extreme distributions Θ^{ext} using (3.4.8).

3. Set $k = 1$. While $k \leq k^{\max}$:

   (a) Solve piecewise linear approximation of the DRO model (3.4.7) using the set Θ^k, to get solution $(\boldsymbol{q}^k, \tilde{\boldsymbol{\theta}})$.

   (b) Use true objective function (3.4.4) to find objective value $\vartheta^k = C_{\tilde{\boldsymbol{\theta}}}(\boldsymbol{q}^k)$.

   (c) Using enumeration, find worst-case cost $C^k = \max_{\boldsymbol{\theta} \in \Theta^{\text{ext}}} C_{\boldsymbol{\theta}}(\boldsymbol{q}^k)$ and worst-case parameter $\boldsymbol{\theta}^k \in \operatorname{argmax}_{\boldsymbol{\theta} \in \Theta^{\text{ext}}} C_{\boldsymbol{\theta}}(\boldsymbol{q}^k)$.

   (d) If $C^k \leq \vartheta^k + \frac{\epsilon}{2}$ or $\boldsymbol{\theta}^k \in \Theta^k$ then set $k = k^{\max} + 1$.

   (e) Else set $\Theta^{k+1} = \Theta^k \cup \{\boldsymbol{\theta}^k\}$ and $k = k + 1$.

4. Return solution $(\boldsymbol{q}^k, \boldsymbol{\theta}^k)$.

Note that, in step 4, CS uses the true objective function to evaluate $\boldsymbol{q}^k$ and to select a worst-case distribution for $\boldsymbol{q}^k$. Therefore, CS may give better solutions than solving the full model using a piecewise linear approximation. Furthermore, this algorithm

typically ends in only a few iterations. This means that the DRO model is only ever solved over a small set of distributions, meaning that the piecewise linear models used solve very quickly.

3.4.4 Experiments on Confidence-based Ambiguity Sets

In this section, we describe the design and results of our numerical experiments assessing the performance of our methods. Firstly, we describe the experimental design used. Following this, we study the order quantities and costs resulting from assuming that $\boldsymbol{\theta}^0 = \hat{\boldsymbol{\theta}}$. Finally, we assess CS as a heuristic for solving the full DRO model. These experiments were run in parallel on a computing cluster (STORM) which has 486 CPU cores. The solver used in all cases was the Gurobi Python package, gurobipy (Gurobi Optimization, LLC, 2022). The version of gurobipy used was 9.0.1. The node used on STORM was the Tukey node, which runs the Linux Ubuntu 16.04.7 operating system, Python version 2.7.12, and 46 AMD Opteron 6238 CPUs.

Experimental Design

In this section, we describe the parameter values used in our experiments. As an ambiguity set, we will use the discretisation Θ'_α of the confidence set (3.4.5) as described in Section 3.4.1. The parameters used in our experiments are as follows. The number of periods was $T \in \{2, 3, 4\}$. The gap for the piecewise linear approximation in Appendix B.3.3 was $\varepsilon \in \{0.1, 0.25, 0.50\}$ and the number of points used in the discretisation of univariate confidence intervals $M \in \{3, 5, 10\}$. The number of samples taken from the true distribution was $N \in \{10, 25, 50\}$. The model costs

were $c, b, h \in \{100, 200\}$ and $w_t = 100(T - t + 1)$ for $t = 1, \ldots, T$. The budget was $W = 4000$ when $T = 2, 3$ and $W = 8000$ for $T = 4$. We randomly selected 3 values of $\boldsymbol{\mu}^0$ from the set $\{1, \ldots, 20\}^T$ and 3 values of $\boldsymbol{\sigma}^0$ from the set $\{1, \ldots, 10\}^T$. These values are chosen in this way for two reasons. Firstly, selecting only 3 keeps the number of experiments manageable. Secondly, selecting the values randomly means that we are not biasing the results by selecting them favourably.

Note that we only select means and standard deviations such that $3\sigma_t^0 \leq \mu_t^0 \ \forall \ t$, to ensure that the demand distributions are unlikely to generate negative values. We also do not consider instances where $T = 4$ and $M = 10$. This is because in these cases the base ambiguity sets are too large to be feasible. The reason why we use $W = 8000$ for $T = 4$ is that the optimal solution if $W = 4000$ is always $(0, 0, 0, 40)$. This is not interesting, and increasing the budget gives a wider variety of solutions. The above inputs yield 864 instances, not including those with $T = 4$ and $M = 10$.

Performance of MLE Approaches

In this section, we assess the performance of the MLE approaches as a heuristic for solving the model under the true distribution. In particular, we study how close to optimality the solutions resulting from the MLE distributions are, and how well the MLE distribution's cost function approximates the true cost function. As a reminder, we use the term *MLE cost function* to mean (3.4.4) with $\boldsymbol{\theta} = \hat{\boldsymbol{\theta}}$, i.e. $C_{\hat{\boldsymbol{\theta}}}$. Similarly, we use *true cost function* to mean (3.4.4) with $\boldsymbol{\theta} = \boldsymbol{\theta}^0$, i.e. $C_{\boldsymbol{\theta}^0}$. We ran the MLE approaches on the 864 instances generated by the inputs from Section 3.4.4, solving each instance with both FD and PLA.

We present boxplots summarising the performance of the MLE approach in Figure 3.4.1. Note that, as in Section 3.3.2, boxplots are presented with outliers (points further than 1.5IQR above the 75th percentile or below the 25th percentile) removed. Let $\hat{\boldsymbol{q}}$ be the solution obtained from applying FD assuming $\boldsymbol{\theta} = \hat{\boldsymbol{\theta}}$. Figure 3.4.1a shows the *absolute percentage optimality gap* of $\hat{\boldsymbol{q}}$ when compared with the optimal $\boldsymbol{q}$ under $\boldsymbol{\theta}^0$, given by:

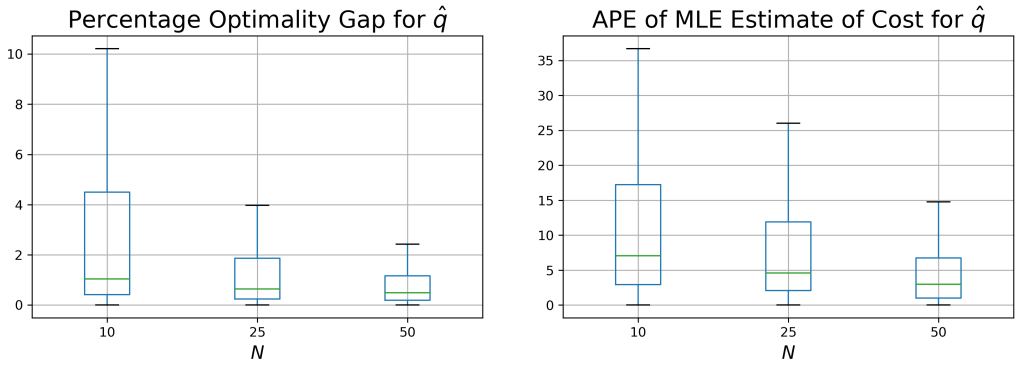
$$100 \times \left| \frac{\min_{\boldsymbol{q}} C_{\boldsymbol{\theta}^0}(\boldsymbol{q}) - C_{\boldsymbol{\theta}^0}(\hat{\boldsymbol{q}})}{\min_{\boldsymbol{q}} C_{\boldsymbol{\theta}^0}(\boldsymbol{q})} \right|.$$

This plot shows that $\hat{\boldsymbol{q}}$ is quite close to optimality under $\boldsymbol{\theta}^0$. For the smallest samples, i.e. $N = 10$, it typically was no further than 10% from optimal. The optimality gap reduces as N increases. We also study the *absolute percentage error* (APE) of $C_{\hat{\boldsymbol{\theta}}}(\hat{\boldsymbol{q}})$ as an estimate of $C_{\boldsymbol{\theta}^0}(\hat{\boldsymbol{q}})$, calculated as:

$$100 \times \left| \frac{C_{\boldsymbol{\theta}^0}(\hat{\boldsymbol{q}}) - C_{\hat{\boldsymbol{\theta}}}(\hat{\boldsymbol{q}})}{C_{\boldsymbol{\theta}^0}(\hat{\boldsymbol{q}})} \right|.$$

Figure 3.4.1b shows that the MLE distribution can result in poor estimates of the cost of $\hat{\boldsymbol{q}}$. The plot shows that for $N = 10$, the APE can reach over 35%. The accuracy of the estimate increases with N, but even for $N = 50$ we see errors of 15%.

In addition, these errors can be even more extreme than shown here, due to outliers being removed from the plot. In one instance, the APE was as high as 15,600%. This occurred in a 2-period instance with $N = 25$, and the corresponding costs were $C_{\hat{\boldsymbol{\theta}}}(\hat{\boldsymbol{q}}) = -386.06$ and $C_{\boldsymbol{\theta}^0}(\hat{\boldsymbol{q}}) = -2.46$. In addition to this, we found that in 21 instances the MLE solution gave an order quantity that was predicted to make a profit but ultimately resulted in a loss. In the most extreme example of this, we found $C_{\hat{\boldsymbol{\theta}}}(\hat{\boldsymbol{q}}) = -479.12$ and $C_{\boldsymbol{\theta}^0}(\hat{\boldsymbol{q}}) = 52.42$. The lowest cost under $\boldsymbol{\theta}^0$ in this instance



(a) (b)

Figure 3.4.1: Boxplots summarising performance of FD in comparison with true optimal solution.

was -5.42, meaning that it was possible to make a profit, but that using the MLE approach would have resulted in a loss. Interestingly, this happened for all values of N and was not restricted to $N = 10$. We found that the MLE approach was likely to underestimate the cost of its chosen $\boldsymbol{q}$. This was the case in 61% of instances.

We can also confirm that these results are not simply due to the performance of FD as a heuristic for MPNVP. We found that FD and PLA's solutions were typically within -2% and 1.5% of one another in terms of cost. We also found that FD sometimes gave better solutions than PLA. In addition, we found similar results about the performance of the MLE approach when the model was solved using PLA. Namely, the MLE cost function predicted a profit for PLA's solution when there would in fact be a cost in 16 instances, and it underestimated the cost of PLA's decision in 59% of instances.

From the results shown here, we can make three key conclusions. Firstly, using the MLE distribution will typically lead to solutions that are suboptimal, but still

reasonably close to optimal under the true distribution. Secondly, we can conclude that the MLE distribution can lead to poor estimates of the costs associated with a given $\boldsymbol{q}$. In particular, solving using the MLEs can lead to the suggestion that the resulting decision will lead to a profit when it will actually lead to a cost. Finally, we can conclude that using the MLE distribution is likely to lead to an underestimation of the cost associated with the selected decision.

We can confirm that the DRO objective in (3.4.1) never predicted a profit when the solution would actually result in a loss. For the aforementioned instance where $\hat{\boldsymbol{q}}$ had a predicted cost of -479.12 and an actual cost of 52.4, the cost that would be associated with this decision under the DRO model was 156.84.

Performance of DRO Algorithms

Each algorithm was given a maximum time of four hours in which to carry out all pre-computation and to build and solve the resulting models. By pre-computation, we mean computing the sets Θ'_{base} and Θ'_{α}. Since this precomputation was required to generate the inputs for both algorithms and is not technically part of either algorithm, we do not include this in the times that we now present. Instead, we present *run times*, i.e. the amount of time the algorithms ran for after all precomputation was complete, until it either finished running or was timed out. We use the term timed out to mean that the algorithm's run time exceeded the time that was remaining after all precomputation was complete. For example, if an algorithm timed out with a run time of 1 minute, this meant that it spent 3 hours and 59 minutes in pre-computation.

Of the 864 instances that we ran, 6 instances resulted in ambiguity sets with only 1 parameter. Hence, they were skipped. This means that we have results for 858 instances. Of these 858 instances, PLA timed out in 1. CS did not time out in any instance. PLA took 7 minutes and 26 seconds on average, whereas CS took just 1.2 seconds. The 25% and 75% quantiles for PLA's time taken are 0.7 seconds and 52.6 seconds. For CS, these values are 0.72 seconds and 3.3 seconds. In addition, PLA ran for up to a maximum of 3 hours and 53 minutes, whereas CS always finished running in less than 8 seconds. Note that PLA never ran for more than 4 hours because the time limit accounts for pre-computation time and run time. In the above example, the pre-computation took 7 minutes and PLA ran for 3 hours and 53 minutes. It is worth noting that PLA's average time is strongly affected by some large run times. PLA took no more than 2 minutes to finish running in 79% of instances. However, it took over 3 hours in approximately 1% of instances.

We can now look at the factors affecting the pre-computation and run times. From our analysis, the main factor affecting PLA's run times is M. For large M, the base ambiguity set Θ'_{base} becomes very large. Therefore, Θ'_{α} is also large and even constructing it can take a very long time. Table 3.4.1 summarises the times taken to complete pre-computation, for each M. This time includes building Θ'_{base}, extracting Θ'_{α} from this set, and reducing Θ'_{α} using our theoretical results. Table 3.4.1 shows that this pre-computation takes a negligible amount of time for $M \in \{3, 5\}$. It also shows a drastic increase in pre-computation times when $M = 10$ compared to $M = 5$. The pre-computation times exceeded 4 hours in 108 instances, all of which had $M = 10$

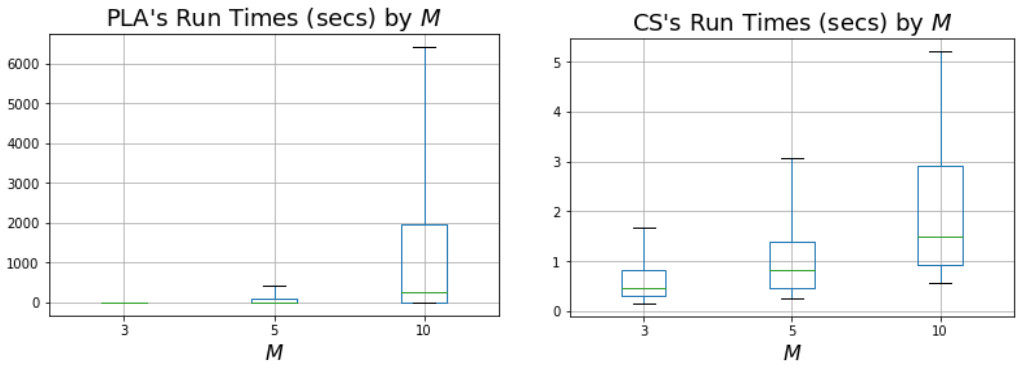
and $T = 4$. The maximum time taken for $M = 5$ was 3.14 seconds.

M	Mean	Std	Median	75th Percentile
3	0.057	0.162	0.004	0.013
5	0.447	0.526	0.130	0.997
10	4883.919	6750.327	235.938	14402.217

Table 3.4.1: Summary of precomputation times (normal)

We now highlight the effect of M on PLA and CS's run times. To do so, we present boxplots of these times grouped by M in Figure 3.4.2. Figure 3.4.2a shows that PLA begins to solve slowly for some instances when $M = 5$. The maximum time that PLA took when $M = 5$ was 15.5 minutes. On average, however, PLA was still relatively fast for $M = 5$, taking an average of 1 minute and 21 seconds over all instances. Figure 3.4.2a also shows that PLA's run times were much larger for $M = 10$. The average time taken by PLA for $M = 10$ was 28 minutes. The effect of M on PLA's run times is due to the fact that larger M means larger Θ'_α and therefore more constraints in the full model. As might have been expected, M does not significantly effect CS's solution times. This is due to the fact that it uses a reduced subset of Θ'_α which has approximately the same size regardless of M.

Since large M leads to significant increases in pre-computation and solution times, it is reasonable to wonder if selecting a larger M results in better solutions under some metric. In order to assess the effect of M on solution quality, we analyse the cost of PLA and CS's solutions under the true distribution. In particular, we calculate the



(a) (b)

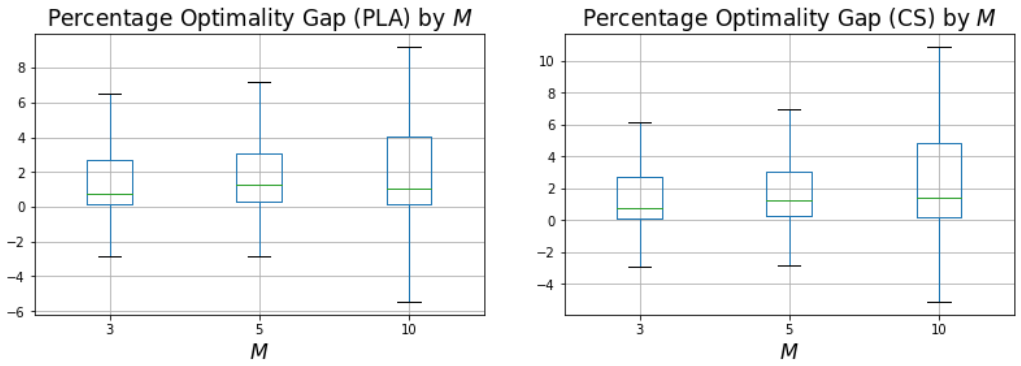
Figure 3.4.2: Boxplots of the run times of (a) PLA and (b) CS by M (normal)

percentage optimality gaps of these solutions as:

$$100 \times \frac{\min_{\boldsymbol{q}} C_{\boldsymbol{\theta}^0}(\boldsymbol{q}) - C_{\boldsymbol{\theta}^0}(\boldsymbol{q}^y)}{|\min_{\boldsymbol{q}} C_{\boldsymbol{\theta}^0}(\boldsymbol{q})|}, \; y \in \{\text{CS}, \text{PLA}\}.$$

Note that, since $\min_{\boldsymbol{q}} C_{\boldsymbol{\theta}^0}(\boldsymbol{q})$ is calculated approximately by applying piecewise linear approximations, it may be the case that PLA or CS's DRO solutions yield lower costs under $\boldsymbol{\theta}^0$. Let $\boldsymbol{q}^0$ be the solution obtained from solving the model under $\boldsymbol{\theta}^0$. We provide boxplots of the percentage optimality gaps in Figure 3.4.3. Figure 3.4.3a shows that, regardless of M, $\boldsymbol{q}^{\text{PLA}}$ attained a cost that was typically within -6% and 9% of the cost attained by $\boldsymbol{q}^0$. Similarly, $\boldsymbol{q}^{\text{CS}}$ typically attained a cost that was between -5% and 11% of the cost of $\boldsymbol{q}^0$.

Figure 3.4.3 suggests that M does not have a particularly strong effect on the quality of the DRO solutions under $\boldsymbol{\theta}^0$. This may indicate that there is no real reason to select $M = 10$ as opposed to $M = 3$ or 5. In order to see this more clearly, we provide some summary statistics in Table 3.4.2. Table 3.4.2 shows that PLA had the highest average percentage optimality gap for $M = 10$, and lowest for $M = 5$.



(a) (b)

Figure 3.4.3: Percentage optimality gaps of (a) PLA and (b) CS by M (normal)

In addition, PLA's optimality gap had the lowest standard deviation when $M = 5$, and it's 75th percentile was lower for $M = 5$ than for $M = 10$. Therefore, taking $M = 10$ drastically increases run times compared to taking $M = 5$ and generally does not result in better solutions under $\boldsymbol{\theta}^0$. These results suggest that $M = 5$ should be preferred over $M = 10$ if optimality under the true distribution is of concern.

M	Mean	Std	Median	75th Percentile
3	21.224	191.404	0.766	2.718
5	15.631	87.505	1.274	3.040
10	21.355	132.053	1.060	4.016

Table 3.4.2: Summary of optimality gaps of PLA by M (normal)

We now assess the performance of CS as a heuristic for solving the full DRO model. This is done via comparing its objective values and returned $\boldsymbol{\theta}$ values with those from PLA. We first present the number of times that CS and PLA obtained the worst-case

parameters for their chosen $\boldsymbol{q}$. We present these values for PLA as well as CS since both algorithms use a reduced ambiguity set. CS uses Θ^{ext} and PLA uses Θ'_α with dominated parameters removed. Hence, studying these results allows us to confirm that the reduced set used by PLA contains the true worst-case parameter for each solution. We present the optimality counts in Table 3.4.3. Note that these counts are given only over the instances where the algorithm concerned finished running. Since PLA timed out in one instance, it finished running in 1 less instance than CS.

CS chose the worst-case parameters for its selected $\boldsymbol{q}$ in 85% of instances. PLA chose the worst-case parameters in every instance. This is expected, since despite the fact that the approximate objective can lead to the wrong worst-case parameter, we use the true objective function to generate the cost of PLA's solution after it finishes running. The main purpose of this is to confirm that our removal of dominated distributions did not remove the worst-case parameter in any instance. CS failed to select the worst-case parameters in 126 instances.

Table 3.4.3 only shows the number of times CS chose the worst-case parameters. In order to gain a better understanding of its performance, we study two quantities:

1. *Percentage $\boldsymbol{\theta}$-gap*: The percentage difference between the worst-case cost for $\boldsymbol{q}^{\text{CS}}$ and the cost obtained under the $\boldsymbol{\theta}$ returned by CS. Defined as:

$$100 \times \frac{\max_{\boldsymbol{\theta}\in\Theta'_\alpha} C_{\boldsymbol{\theta}}(\boldsymbol{q}^{\text{CS}}) - C_{\boldsymbol{\theta}^{\text{CS}}}(\boldsymbol{q}^{\text{CS}})}{\max_{\boldsymbol{\theta}\in\Theta'_\alpha} C_{\boldsymbol{\theta}}(\boldsymbol{q}^{\text{CS}})}.$$

2. *Percentage $\boldsymbol{q}$-gap*: The percentage difference between the worst-case cost at-

tained by $\boldsymbol{q}^{\mathrm{CS}}$ and that attained by $\boldsymbol{q}^{\mathrm{PLA}}$. Defined as:

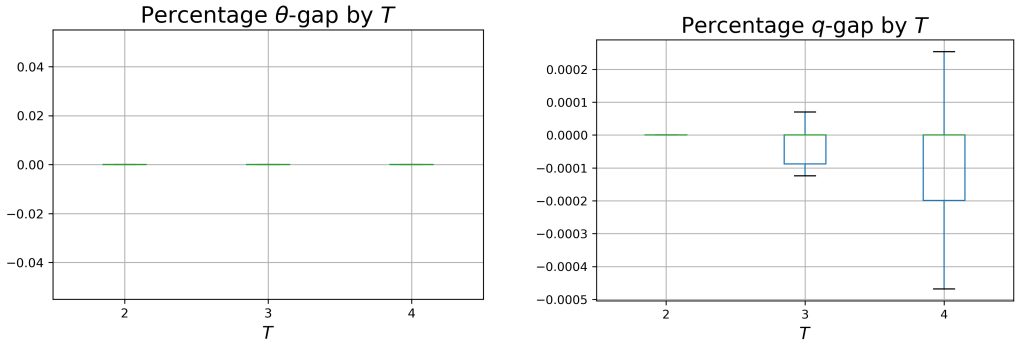
$$100 \times \frac{\max_{\boldsymbol{\theta}\in\Theta'_\alpha} C_{\boldsymbol{\theta}}(\boldsymbol{q}^{\mathrm{PLA}}) - \max_{\boldsymbol{\theta}\in\Theta'_\alpha} C_{\boldsymbol{\theta}}(\boldsymbol{q}^{\mathrm{CS}})}{\max_{\boldsymbol{\theta}\in\Theta'_\alpha} C_{\boldsymbol{\theta}}(\boldsymbol{q}^{\mathrm{PLA}})}.$$

T	CS No.	CS Worst-case Count (%)	PLA No.	PLA Worst-case Count (%)
All	858	732 (85.31%)	857	857 (100.0%)
2	320	270 (84.38%)	320	320 (100.0%)
3	323	257 (79.57%)	322	322 (100.0%)
4	215	205 (95.35%)	215	215 (100.0%)

Table 3.4.3: Number of instances in which algorithms selected true worst case parameters.

Note that positive values (lower expected cost) for the $\boldsymbol{\theta}$-gap implies that CS gave a suboptimal $\boldsymbol{\theta}$. Negative values (higher worst-case expected cost) for the $\boldsymbol{q}$-gap imply that CS gave a suboptimal $\boldsymbol{q}$. Figure 3.4.4 summarises the percentage gaps using boxplots. The first thing to notice is that the $\boldsymbol{\theta}$-gaps were so commonly zero that everything non-zero was considered as an outlier. On average, the percentage $\boldsymbol{\theta}$-gap was 0.045%. We see from Figure 3.4.4b that CS's chosen $\boldsymbol{q}$ had a worst-case cost that was very close to PLA's, in general. The percentage optimality gaps typically do not exceed 0.003%. The overall average percentage $\boldsymbol{q}$-gap was -0.06%. CS even performed better than PLA in 153 instances. Since CS can be viewed as a heuristic for solving the full model that PLA solves, this may be unexpected. This occurs due to one main reason. CS's solution can have a higher worst-case cost than PLA's

solution under the approximate cost function used by PLA, even when the worst-case cost of CS's solution is lower than that of PLA's under the true cost function.



(a) (b)

Figure 3.4.4: Boxplots of (a) percentage $\boldsymbol{\theta}$-gaps and (b) percentage $\boldsymbol{q}$-gaps by T

From these results we can conclude 4 main points. Firstly, CS was much faster than PLA and scales better. It never took more than 8 seconds to finish running, whereas PLA took up to 3 hours and 53 minutes. Secondly, PLA performed better in selecting the worst-case parameters for a fixed $\boldsymbol{q}$. This is expected, as CS applies cardinality reduction to the ambiguity set. CS's returned parameters were very close to the worst-case in terms of cost, however. Thirdly, CS was very close to optimal with respect to its chosen $\boldsymbol{q}$. On average, it returned solutions that are only 0.045% away from optimal. Finally, CS performed better than PLA in some instances, due to the inaccuracy of the piecewise linear approximation.

3.5 DRO Model with Poisson Demands

In this section, we present and solve the DRO model when the demands are Poisson random variables. In Section 3.5.1, we give the model and describe the ambiguity

sets used. Then, in Section 3.5.2, we describe the set of extreme parameters used by CS. Finally, in Section 3.5.3, we test the MLE approach and our CS algorithm using computational experiments.

3.5.1 Formulation and Ambiguity Sets

When the demands are independent Poisson random variables, i.e. $X_t \sim \text{Pois}(\lambda_t^0)$ for unknown $\boldsymbol{\lambda}^0$, then we have that $\sum_{l=1}^t X_l \sim \text{Pois}\left(\sum_{l=1}^t \lambda_l^0\right)$ for $t = 1, \ldots, T$. Since we have $\boldsymbol{\theta} = \boldsymbol{\lambda}$, we use $\boldsymbol{\lambda}$ in place of $\boldsymbol{\theta}$ from hereon out. Also, we will now consider integer decision variables, i.e. $\boldsymbol{q} \in \mathbb{N}_0^T$, since the demand is discrete. Furthermore, we can use Lemma 3.5.1 to rewrite the objective function in a more convenient fashion. This lemma is proved in Appendix B.4.1.

Lemma 3.5.1. *Suppose that the demands are independent Poisson random variables, i.e.* $X_t \sim Pois(\lambda_t)\ \forall\ t = 1, \ldots, T$. *Then, for* $\boldsymbol{q} \in \mathbb{N}_0^T$, *the objective function* $C_{\boldsymbol{\lambda}}(\boldsymbol{q})$ *can be written as:*

$$C_{\boldsymbol{\lambda}}(\boldsymbol{q}) = \sum_{t=1}^{T} \Big(a_t Q_t \tilde{F}_t(Q_t) - \Lambda_t a_t \tilde{F}_t(Q_t - 1) + (b + c\mathbb{1}\{t = T\})(\Lambda_t - Q_t) + w_t q_t - c\lambda_t \Big),$$

with $a_t = h + b + c\mathbb{1}\{t = T\}$ *and* $\Lambda_t = \sum_{l=1}^t \lambda_l$.

Therefore, the DRO model for Poisson demands is given by:

$$\min_{\boldsymbol{q}} \max_{\boldsymbol{\lambda} \in \Theta} C_{\boldsymbol{\lambda}}(\boldsymbol{q}) = \sum_{t=1}^{T} \Big(a_t Q_t \tilde{F}_t(Q_t) - \Lambda_t a_t \tilde{F}_t(Q_t - 1) + (a_t - h)(\Lambda_t - Q_t) + w_t q_t - c\lambda_t \Big) \tag{3.5.1}$$

$$\text{s.t.} \quad \sum_{t=1}^{T} w_t q_t \leq W, \tag{3.5.2}$$

$$q_t \in \mathbb{N}_0 \ \forall \ t \in \{1, \ldots, T\}. \tag{3.5.3}$$

Similarly to the case for normally distributed demands, we will construct confidence sets for use as ambiguity sets. Suppose that we draw N samples $\boldsymbol{x}^n = (x_1^n, \ldots, x_T^n)$ from $\boldsymbol{X}$. Then, the MLEs are given by $\hat{\lambda}_t = \frac{1}{N}\sum_{n=1}^{N} x_t^n$ for $t = 1, \ldots, T$. We can construct an approximate $100(1-\alpha)\%$ confidence set for $\boldsymbol{\lambda}^0$ using Proposition 3.5.2, which is proved in Appendix B.4.2.

Proposition 3.5.2. *Suppose that $X_t \sim Pois(\lambda_t^0)$ for $t \in \{1, \ldots, T\}$ and that the X_t are independent. Then, an approximate $100(1-\alpha)\%$ confidence set for $\boldsymbol{\lambda}^0$ is given by:*

$$\Theta_\alpha = \left\{ \boldsymbol{\lambda} \in \mathbb{R}_+^T : \sum_{t=1}^{T} \frac{N}{\hat{\lambda}_t}(\hat{\lambda}_t - \lambda_t)^2 \leq \chi^2_{T,1-\alpha} \right\}. \tag{3.5.4}$$

As before, we will construct a discretisation of Θ_α. In the Poisson case, we have that $\boldsymbol{\lambda} \in \Theta_\alpha$ implies that:

$$\lambda_t \in \lambda_t^{\text{int}} = \left[\hat{\lambda}_t - \sqrt{\frac{\chi^2_{T,1-\alpha}\hat{\lambda}_t}{N}}, \hat{\lambda}_t + \sqrt{\frac{\chi^2_{T,1-\alpha}\hat{\lambda}_t}{N}} \right] \quad (t = 1, \ldots, T).$$

Therefore, if we define $\Theta_{\text{base}} = \left\{ \boldsymbol{\lambda} \in \mathbb{R}_+^T : \lambda_t \in \lambda_t^{\text{int}} \ \forall \ t = 1, \ldots, T \right\}$, then we have $\Theta_\alpha \subseteq \Theta_{\text{base}}$. As before, we can discretise the intervals λ_t^{int} using:

$$\tilde{\lambda}_t^{\text{int}} = \left\{ \lambda_t^l + m\frac{\lambda_t^u - \lambda_t^l}{M-1}, m = 0, \ldots, M-1 \right\},$$

where λ_t^l, λ_t^u are the lower and upper bounds of λ_t^{int}. Hence, we can calculate a discretisation of Θ_{base} by taking $\Theta'_{\text{base}} = \{\boldsymbol{\lambda} \in \mathbb{R}_+^T : \lambda_t \in \tilde{\lambda}_t^{\text{int}} \ \forall \ t \in \{1, \ldots, T\}\}$. Finally, we can take a discretisation of the confidence set Θ_α by taking $\Theta'_\alpha = \Theta'_{\text{base}} \cap \Theta_\alpha$. This is the set used in our experiments. For the discrete ambiguity set Θ'_α, we can

reformulate (3.5.1)-(3.5.3) as:

$$\min\left\{\vartheta : \vartheta \leq C_{\boldsymbol{\lambda}}(\boldsymbol{q})\ \forall\ \boldsymbol{\lambda} \in \Theta'_\alpha, \sum_{t=1}^{T} w_t q_t \leq W, \boldsymbol{q} \in \mathbb{N}_0^T\right\}.$$

This model is implemented in Gurobi using piecewise linear constraints. For more details on how the model is implemented, see Appendix B.4.3.

3.5.2 Extreme Distributions

As for the case of normally distributed demands, we will develop a CS algorithm that produces solutions in a comparatively short time. In order to construct the set Θ^{ext} of extreme parameters, we will use the result in Theorem 3.5.3.

Theorem 3.5.3. *Suppose that $X_t \sim Pois(\lambda_t^0)$ for $t \in \{1,\ldots,T\}$ and that the X_t are independent. Then $C_{\boldsymbol{\lambda}}(\boldsymbol{q})$ is convex in λ_t for each $t \in \{1,\ldots,T\}$.*

This result is proved in Appendix B.4.4. This result means that the worst-case λ_t will either be the smallest in the ambiguity set or the largest (when all other λ_l are fixed). Therefore, we can define a set of extreme distributions for a given Θ'_α as:

$$\Theta^{\text{ext}} = \left\{\boldsymbol{\lambda} \in \Theta'_\alpha \middle| \exists t \in \{1,\ldots,T\} : \lambda_t \in \left\{\min_{\boldsymbol{\theta}\in\Theta'_\alpha} \theta_t, \max_{\boldsymbol{\theta}\in\Theta'_\alpha} \theta_t\right\}\right\}. \tag{3.5.5}$$

The CS algorithm is therefore the same as the one in Section 3.4.3, but using (3.5.5) as the set of extreme distributions and (3.5.1) as the objective function.

3.5.3 Experimental Design and Results

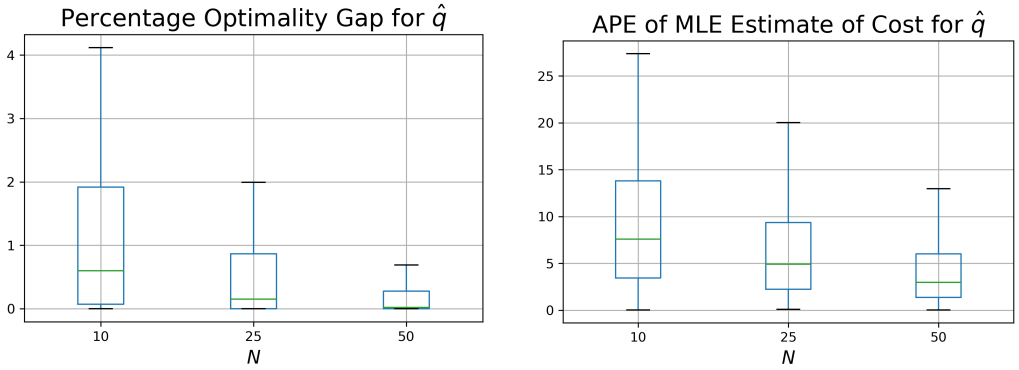
To test the CS algorithm on Poisson demands, we run experiments on the same inputs as for the normal distribution. In every case, we take the value of $\boldsymbol{\lambda}^0$ to be equal to

the sampled value of $\boldsymbol{\mu}^0$ in the normal experiments. This allows to see the effect on our results if the only aspect allowed to change is the distribution used. Note, however, that we now run the tests with $T = 4$ and $M = 10$, since there is now only one parameter per day and so the Poisson ambiguity sets are much smaller. Since the objective function is piecewise constant between integer points, we can simply use an ε of 1 in all instances. Hence, we run our experiments for an additional 6 randomly selected values of $\boldsymbol{\lambda}^0$, in order to generate more instances.

These parameters yield 972 instances (since we now consider $T = 4$ and $M = 10$). Note that, as in the normal case, all boxplots have outliers (points further than 1.5IQR above the 75th percentile or below the 25th percentile) removed. These experiments were also run on STORM, this time on the Wald node. The Wald node runs 70 Intel Xeon E5-2699 CPUs, and also runs Python 2.7.12 and Gurobipy 9.0.1.

Performance of MLE Approaches

Firstly, as for the normal distribution, we assess the effect of solving the newsvendor problem using the MLE distribution. We present the APE of the MLE's cost estimate along with the optimality gap of the MLE solution in Figure 3.5.1. This figure indicates that the MLE approach is close to optimal for Poisson demands, with solutions typically not being further than 4% from optimal. However, the cost estimates are still not accurate. For $N = 10$, we can see some APEs above 25%. Even for $N = 50$, we see APEs of almost 15%. In addition, this plot does not show outliers. There were 19 instances where the APE exceeded 100%, and 2 where it exceeded 10000%.



(a) (b)

Figure 3.5.1: Boxplots summarising performance of MLE approach. Figure 3.5.1(a) shows the optimality gap of $\hat{\boldsymbol{q}}$ as a solution to the model under $\boldsymbol{\lambda}^0$. Figure 3.5.1(b) shows the APE of $C_{\hat{\boldsymbol{\lambda}}}(\hat{\boldsymbol{q}})$ as an estimate of $C_{\boldsymbol{\lambda}^0}(\hat{\boldsymbol{q}})$.

In addition, we again find instances where the MLE predicts a profit for a decision that would actually attain a cost. This happened in 3 instances for the Poisson distribution, as opposed to 24 for the normal distribution. We summarise these 3 instances in Table 3.5.1. This table shows the values of $\boldsymbol{\lambda}^0$ and $\hat{\boldsymbol{\lambda}}$ as well as the different costs associated with the MLE solution. This table shows that, in every instance, the DRO cost was positive. This suggests that the DRO objective would always show the newsvendor that the solution could lead to a cost. We also found that the MLE distribution predicted a cost when it would make a profit, although this only happened in two instances.

Similarly to the normal case, we can confirm that the poor accuracy is not due to FD being a heuristic for the MPNVP model. In the Poisson case, the two algorithms are even closer to one another than in the normal case. We found that FD returned the same solution as PLA in 921 of the 972 instances considered. In addition, the

$\boldsymbol{\lambda}^0$	$\hat{\boldsymbol{\lambda}}$	FD Prediction	True Cost	DRO Cost
(10, 17)	(8.8, 15.72)	-28.62960	118.14383	177.11568
(9, 16)	(8.12, 15.2)	-25.11351	52.26802	130.41717
(15, 16)	(14.24, 16.24)	-16.40814	110.62268	264.08408

Table 3.5.1: Summary of instances where FD predicted profit but incurred a cost

solutions from the two algorithms had the same objective values in the other 51 instances. Therefore, the two algorithms have exactly the same performance for Poisson demands. Hence, the issues presented here are likely only due to the effects of treating the MLE as the true parameter.

In conclusion, the MLE approach performs better for Poisson demands than for normal ones, but it still has the same issues. In particular, it provides poor estimates of the cost associated with an order quantity. It is even capable of predicting a profit when a cost would be incurred, and vice versa. In addition, we confirmed that the DRO solution does not do this. In fact, in these instances the DRO cost was sometimes closer to reality than the MLE cost.

Performance of DRO Algorithms

We now present the results for the 952 instances where we solved the DRO model with Poisson demands. The other 20 instances only had singleton ambiguity sets. We will summarise the run times of PLA and CS, along with CS's performance as a heuristic for the full model. Both algorithms were given a maximum time of 4 hours to build

and solve the models. Firstly, we found that PLA timed out in 27 instances and CS did not time out in any instance. PLA took an average of 11 minutes and 12 seconds to either return a solution or time out, and its maximum time was 4 hours. CS took an average of 0.75 seconds, and its maximum time was 5.9 seconds. The 25th and 75th percentiles of PLA were 0.45 seconds and 16.84 seconds, respectively. For CS, these values were 0.22 and 0.61 seconds. Hence, again it is clear that PLA's average was affected by a small number of very large run times.

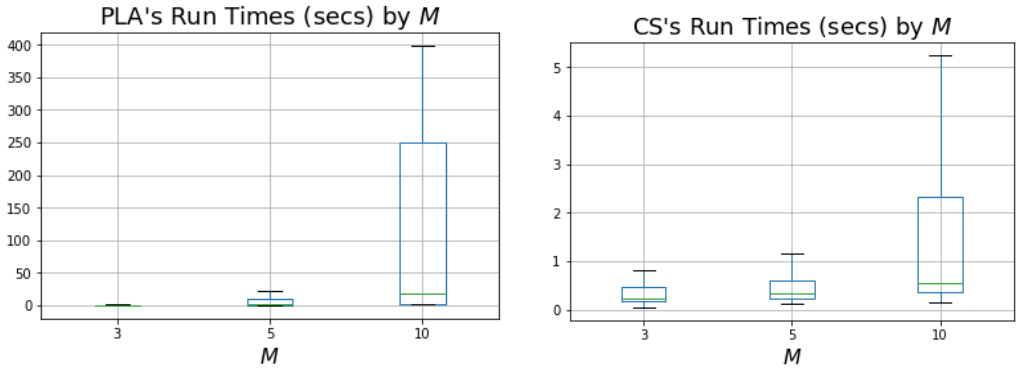
For Poisson demands, the pre-computation stage was much faster. In fact, this stage never took longer than 3 seconds. We summarise the pre-computation stage in Table 3.5.2. This table suggests that, for Poisson demands, M does not have a significant effect on pre-computation times. This is due to a number of reasons. Firstly, regardless of M, the ambiguity sets were much smaller than the normal ambiguity sets. For $M = 10$, the average size of the Poisson ambiguity set was 781 as opposed to 277441 for normal demands (before reducing the set). This size difference is mainly due to the fact that there are 2 parameters per period for normal demands as opposed to 1 for Poisson demands. In addition, the pre-computation step is faster for Poisson demands since we do not perform the cardinality reduction step that we perform for normal demands. This is not necessary for Poisson ambiguity sets since they are already small, and it can take a large amount of time for normal demands.

Despite M not having much of an effect on pre-computation times, it still had a noticeable effect on PLA's run times. We present these times by M in Figure 3.5.2. We can see that PLA is again greatly affected by M; it begins to take as long as

M	Mean	Std	Median	75th Percentile
3	0.014	0.053	0.001	0.002
5	0.023	0.085	0.002	0.003
10	0.024	0.069	0.007	0.022

Table 3.5.2: Summary of precomputation times (Poisson)

7 minutes for $M = 10$. CS was also effected by M, but much less drastically. The difference is most noticeable for $M = 10$, and this is the value at which both algorithms start to solve more slowly. However, CS never took longer than 6 seconds.



(a) (b)

Figure 3.5.2: Boxplots of the run times of (a) PLA and (b) CS by M (Poisson)

In order to further inform which value of M should be used, we again study the percentage optimality gaps of PLA's solutions under $\boldsymbol{\lambda}^0$. Recall that this gap is defined as:

$$100 \times \frac{\min_{\boldsymbol{q}} C_{\boldsymbol{\lambda}^0}(\boldsymbol{q}) - C_{\boldsymbol{\lambda}^0}(\boldsymbol{q}^{\mathrm{PLA}})}{|\min_{\boldsymbol{q}} C_{\boldsymbol{\lambda}^0}(\boldsymbol{q})|}$$

and let $\boldsymbol{q}^0$ be the solution obtained from solving the model under $\boldsymbol{\lambda}^0$ using piecewise linear approximations. We summarise the optimality gaps of PLA in Table 3.5.3. This table suggests that the quality of PLA's solution w.r.t. the true distribution worsens as M increases. This is reflected by every column present in Table 3.5.3. Since PLA's solution time is much larger for $M = 10$, if optimality under the true distribution is of importance to the decision maker then there is no motivation to use $M = 10$.

M	Mean	Std	Median	75th Percentile
3	7.060	30.897	0.856	2.944
5	9.169	34.267	1.634	4.841
10	12.474	48.967	1.778	6.205

Table 3.5.3: Summary of optimality gaps of PLA by M (Poisson)

We now present the optimality of CS with respect to $\boldsymbol{\lambda}$ and $\boldsymbol{q}$. Firstly, we present the number of times each algorithm selected the worst-case parameter for its chosen $\boldsymbol{q}$ in Table 3.5.4. This table shows that CS selected the worst-case $\boldsymbol{\lambda}$ for its $\boldsymbol{q}$ in 94.2% of instances. We will show that its other choices were very close to the worst-case, however. PLA selected the worst-case in every instance.

The percentage $\boldsymbol{\lambda}$- and $\boldsymbol{q}$-gaps are defined as they were for the normal experiments in Section 3.4.4, and we present summaries of these gaps in Figure 3.5.3. Firstly, Figure 3.5.3a reflects the fact that CS's chosen $\boldsymbol{\lambda}$ was very close to the true worst-case parameter in terms of cost. The overall average $\boldsymbol{\lambda}$-gap was 0.09%. The fact that CS did not always select the true worst-case is due to the fact that Θ^{ext} is constructed

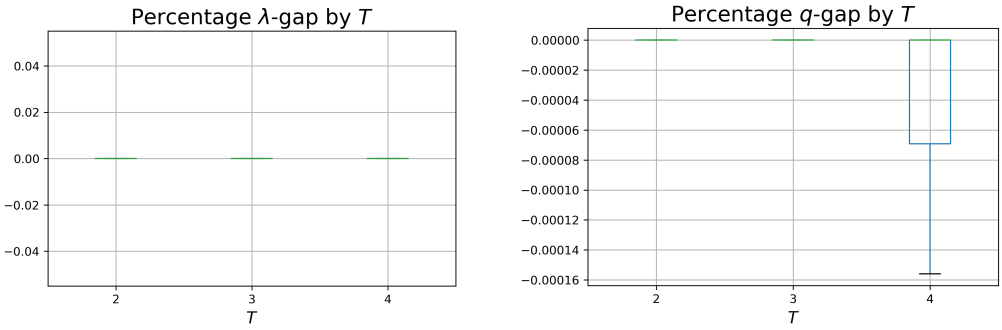
T	CS No.	CS Worst-case Count (%)	PLA No.	PLA Worst-case Count (%)
All	952	880 (92.44%)	925	925 (100.0%)
2	311	311 (100.0%)	311	311 (100.0%)
3	319	311 (97.49%)	319	319 (100.0%)
4	322	258 (80.12%)	295	295 (100.0%)

Table 3.5.4: Number of instances in which algorithms selected true worst case parameters.

by combining univariate results. Despite the fact that the expected cost can always be increased by setting one λ_t to its minimum or maximum, this may not be possible without changing another λ_l if the resulting $\boldsymbol{\lambda}$ is to remain a member of (3.5.4). The cost may improve as a result of these two changes. Hence, the worst-case vectors sometimes did not have any single parameters at their maximum or minimum.

Secondly, Figure 3.5.3b shows that CS typically selected a $\boldsymbol{q}$ with a worst-case cost of less than $1.6 \times 10^{-4}\%$ higher than PLA's. In fact, CS's $\boldsymbol{q}$ had the same worst-case expected cost as PLA's in 796 (86%) of the 925 instances in which PLA did not time out. The overall average $\boldsymbol{q}$-gap was -0.02%, and the minimum value of this gap was -3.14%. CS did not encounter much suboptimality until $T = 4$. In these instances, its suboptimality was likely due to either its selection of the incorrect worst-case parameter, solver-specific issues like Gurobi's pre-solve model reductions, or different starting points for Gurobi's MILP algorithms.

From the results shown here, we can conclude two main points. Firstly, CS again



(a) (b)

Figure 3.5.3: Boxplots of (a) percentage $\boldsymbol{\lambda}$-gaps and (b) percentage $\boldsymbol{q}$-gaps by T.

solved much faster than PLA. Secondly, it was very near optimal if not optimal, in general. It was capable of selecting the worst-case parameter for its chosen $\boldsymbol{q}$ in the large majority of instances, and this $\boldsymbol{q}$ was often optimal.

3.6 Conclusions and Further Work

In this chapter, we considered a static, multi-period newsvendor problem under a budget constraint. We first developed an algorithm to solve the problem under a known demand distribution. This was done by adapting an algorithm from the literature that was used to solve a multi-product budgeted newsvendor problem. It entails first finding a solution to the stationarity KKT condition, and then optimising the Lagrange multiplier until the solution becomes feasible. We tested this algorithm against three benchmarks, and found that it is faster than all 3 of them and provides comparable solutions in terms of expected cost.

Following this, we studied parametric distributionally robust optimisation models for this newsvendor problem. We assumed that the newsvendor knows the family of

distributions in which the true demand distribution lies. We showed how maximum likelihood estimation can be used to construct a confidence-based ambiguity set for the true parameters, and then how to build and solve the resulting distributionally robust model for normal and Poisson demands. Due to the size of the discrete ambiguity sets used, we developed a decomposition algorithm named CS for each family of distributions. We did so by obtaining theoretical results about the behaviour of the expected cost as a function of the distribution's parameters, in order to characterise the worst-case distribution.

We showed that the CS algorithm gave very close to optimal solutions in a matter of seconds, for both families. We also assessed the performance of the method that solves the fixed distribution problem assuming the maximum likelihood estimate is the true parameter. We found that, while the solutions this yielded were not far from optimal, the cost estimates from this model were far from accurate. In fact, we found that there were cases where the maximum likelihood estimate suggested a profit would be made, when in reality a cost would be incurred. We also confirmed that, in these instances, the distributionally robust model would, in fact, make the newsvendor aware that their order could result in a cost. Hence, using distributionally robust models avoids these issues.

There are a number of areas for future work that would prove interesting. The most immediate one is to extend our work to the case of dependent demands. It is common in real-life scenarios that demand for a product over different days will be correlated. However, this creates a much more complex model. We would like to study how this

could be done in future, for example by specifying the full covariance matrix in the multivariate normal case. For Poisson demands, it is less clear how this could be done. Another area for future research is to study other demand distributions, such as exponential or gamma. The two distributions considered here are common for demand in the newsvendor problem, but there are many others that might be used. Finally, we would like to improve the process used to compute the ambiguity sets. Since CS has been shown to be very fast, the only real bottleneck of our approach is the time taken to compute ambiguity sets. It would also be interesting to study how this process can be further improved, potentially with parallelisation and/or faster programming languages.

Chapter 4

Robust Markov Decision Processes Under Parametric Transition Distributions

This chapter considers robust Markov decision processes under parametric transition distributions. We assume that the true transition distribution is uniquely specified by some parametric distribution, and explicitly enforce that the worst-case distribution from the model is uniquely specified by a distribution in the same parametric family. After formulating the parametric robust model, we focus on developing algorithms for carrying out the robust Bellman updates required by robust value iteration.

We first formulate the update as a linear program by discretising the ambiguity set. Since this model scales poorly with problem size and requires large amounts of pre-computation, we develop two additional algorithms for solving the robust Bellman

update. Firstly, we present a cutting surface algorithm for solving this linear program in a shorter time. This algorithm requires the same pre-computation, but only ever solves the linear program over small subsets of the ambiguity set. Secondly, we present a novel projection-based bisection search algorithm that completely eliminates the need for discretisation and does not require any pre-computation.

We test our algorithms extensively on a dynamic multi-period newsvendor problem under binomial and Poisson demands. In addition, we compare our methods with the non-parametric ϕ-divergence based methods from the literature. Through extensive computational experiments, we show that our projection-based algorithm completes robust value iteration significantly faster than our other two parametric algorithms, and faster than its non-parametric equivalent.

4.1 Introduction

Markov decision processes (Bellman, 1957) (MDPs) are a mathematical framework for modelling dynamic decision making problems under uncertainty. Under the MDP framework, at each decision epoch in a finite or infinite time horizon, a decision maker utilises information about the current *state* of a system in order to select an *action* that yields them a reward. The action taken can affect the next state of the system, which is stochastically governed by a set of *transition probabilities*. The goal of the decision maker is to make decisions at each epoch in order to maximise the total (discounted) expected reward that they receive over the entire horizon.

A solution of an MDP is understood as a *policy*, which provides an action or a dis-

tribution over the set of actions to be taken in each state of the system. The policy is found prior to any decisions being made, and in practice the decision maker can instantaneously generate their action from the policy at any given epoch. Policies are usually found from algorithms based on dynamic programming and Bellman's optimality equations (Bellman, 1966), which are based around the concept of *value functions*. Value functions give the expected total future reward from starting in each state and following an optimal policy thereafter.

In classical MDPs, it is assumed that all parameters of the model (rewards, transition probabilities, etc.) are known exactly. However, in practice it can be difficult to determine these parameters exactly and they must often be replaced with estimates. In addition, it has been found that replacing the true parameters with estimates thereof can lead to policies that fail drastically when implemented, due to errors in estimation (Le Tallec, 2007; Wiesemann et al., 2013) and that the resulting value function estimates can have large variance and bias (Mannor et al., 2007). Due to these issues, *robust MDPs* (Satia and Lave, 1973) (RMDPs) have been proposed to explicitly represent uncertainty in model parameters. RMDPs do not assume that all parameters are known, but that they are known to lie in some pre-determined set. The decision maker then aims to find a policy with the best worst-case total reward over all parameters in the set. This limits the potential hazards of poor estimation.

In the case where only the transition probabilities are not known, we refer to this set as an *ambiguity set*. Ambiguity sets are designed so that the decision maker can be confident that the true transition distribution lies within the set. There are many

ways of constructing an ambiguity set. Early sets placed bounds on each transition probability (Satia and Lave, 1973; Givan et al., 2000). In more recent papers, it has become more common to bound the distance between any distribution in the set and some nominal distribution. For example, one can use the Kullback-Leibler divergence, modified χ^2-divergence or L_1-norms (Iyengar, 2005), or more general classes of distance measures such as ϕ-divergence functions (Ho et al., 2022). The choice of ambiguity set strongly affects the tractability of the resulting RMDP model. For general ambiguity sets, it is known that RMDPs are NP-hard. However, this chapter considers a special type of ambiguity set called *s-rectangular* ambiguity sets (Le Tallec, 2007). Such ambiguity sets allow the transition distributions for each state to be chosen independently of one another. The resulting RMDP is solvable in polynomial time via *robust value iteration* (Wiesemann et al., 2013).

Robust value iteration starts with some initial estimate of the value function, then iteratively updates the estimate until Bellman's optimality equations are satisfied. We refer to the process of finding the next value function estimate as solving a *robust Bellman update*. Much of the recent RMDP literature has focused on developing fast algorithms for solving the robust Bellman update in s-rectangular RMDPs. Due to the fact that only the value estimates themselves (and not the optimal policies responsible) are required to complete robust value iteration, many of these algorithms employ simple methods like bisection search (Grand-Clément and Kroer, 2021; Ho et al., 2022) to solve the update. Such methods allow us to compute the optimal policy only once, after value iteration ends, as opposed to in every iteration.

In this chapter, we will focus on developing algorithms for carrying out robust Bellman updates, with one key difference from the existing literature. In particular, we will focus on RMDPs where the true state transition distribution either lies in some parametric family or is specified by some external random variable that lies in some parametric family (e.g. demand, service times, failure rates). In the existing RMDP literature, transition distributions are assumed to be non-parametric, and ambiguity sets typically contain non-parametric distributions. However, in the case where the transition distribution is indeed parametric, non-parametric ambiguity sets necessarily contain distributions that cannot be equal to the true distribution. Our models use *parametric ambiguity sets* to enforce that every potential distribution in the set lies in the same parametric family as the true distribution.

Constructing an RMDP in this fashion has a number of benefits. Firstly, it means that we only need to find the worst-case parameters and not the entire worst-case distribution. The worst-case parameter is typically of much smaller dimension than the worst-case distribution, meaning that finding it can be much less cumbersome. As such, instead of ambiguity sets for the true distribution, we use ambiguity sets for the true parameters. Secondly, explicitly using ambiguity sets for the true parameters and using the corresponding parametric distributions in the model means that every worst-case distribution generated by the model will lie in the correct parametric family. In addition, we can use maximum likelihood estimation to build confidence-based ambiguity sets. Finally, parametric distributions are natural models for random variables affecting MDP state transitions in a number of problems.

An example of such a problem is the *dynamic multi-period newsvendor problem* (Arrow et al., 1958). In newsvendor models, demand is often considered as a parametric random variable. Newsvendor demand has been modelled as normal (Nahmias, 1994), negative binomial (Agrawal and Smith, 1996), lognormal and gamma (Gallego et al., 2007), and exponential (Siegel and Wagner, 2021). In addition, for such problems it has been shown that assuming that replacing the true parameters with estimates can lead to poor cost estimation (Rossi et al., 2014; Siegel and Wagner, 2021). Hence, a parametric ambiguity set provides a way to hedge against parameter uncertainty while ensuring the worst-case distribution is also parametric.

This chapter extends the concept of parametric ambiguity sets from Chapters 2 and 3, where we studied parametric *distributionally robust optimisation* (DRO) models for a resource planning problem and a newsvendor problem, into the RMDP literature. We formulate s-rectangular parametric ambiguity sets and solve the resulting RMDP via robust value iteration. Under such ambiguity sets, the robust Bellman update is a parametric DRO problem. As a benchmark, we reformulate the robust Bellman update as a *linear program* (LP) by discretising the ambiguity set.

Since this LP can become very slow for large problems, we develop two additional algorithms for carrying out the update. The first is a *cutting surface* (Mehrotra and Papp, 2014) (CS) algorithm that iteratively solves the LP over increasing subsets of the ambiguity set. The second algorithm is a parametric projection-based bisection search algorithm. This algorithm does not rely on any discretisation of the ambiguity set, and we will show that this means that it solves the robust Bellman updates

orders of magnitude faster than both CS and LP. In summary, the contributions of this chapter are as follows:

1. We extend the concept of parametric ambiguity sets from Chapters 2 and 3 into the RMDP literature. Such ambiguity sets have only been used for static DRO problems in the past. Since the DRO model used in the robust Bellman update must be solved multiple times in an iterative fashion, the scalability of the algorithms and computation is even more of a challenge in RMDPs.

2. We develop a fast projection-based bisection search algorithm for solving a robust Bellman-update, that does not rely on any discretisation of the parametric ambiguity set.

3. We apply our methods to a dynamic multi-period newsvendor model under binomial and Poisson demands. The results show that parametric robust value iteration can be solved faster than its non-parametric equivalent.

4.2 Literature Review

4.2.1 Robust Markov Decision Processes

RMDPs are a framework for modelling MDPs with unknown parameters, which has become common in recent years due to the fact that MDPs are extremely sensitive to small changes in their parameters (Mannor et al., 2007). RMDPs have been studied in the literature since the 1970s, where the first example of an RMDP used ambiguity sets based on assigning upper and lower bounds to each transition probability in the

set (Satia and Lave, 1973). Such ambiguity sets were common in the early RMDP literature. Givan et al. (2000) also studied bounded parameter RMDPs, which were solved by solving a collection of exact MDPs. Later, Bagnell et al. (2001) generalised the concepts of RMDPs to a variety of other ambiguity sets. Their only assumption was that the sets were convex and compact, meaning that the class they considered covered interval ambiguity sets as a special case.

In addition, finite horizon RMDPs were also studied, bringing forth a robust version of *dynamic programming* (DP) (Nilim and El Ghaoui, 2005). These authors further developed the ambiguity sets used in RMDPs to encompass distance-based sets, such as those based on the Kullback-Leibler divergence. Using such ambiguity sets allowed the Bellman optimality equations to be reformulated using Lagrangian dualisation and hence solved exactly or via bisection. Iyengar (2005) formalised these concepts further, studying finite and infinite horizon RMDPs with a variety of ambiguity sets. For example, they studied ambiguity sets built using the Kullback-Leibler divergence, modified χ^2-divergence and L_1 norm.

Since these early papers, s-rectangular ambiguity sets have become very common in RMDPs. An s-rectangular ambiguity set (Le Tallec, 2007) is one arising from the situation in which the state transitions for each state are independent of one another. Hence, the worst-case distributions for each state can be extracted independently of one another. Solving an RMDP with an s-rectangular ambiguity set is equivalent to finding a fixed point of the robust Bellman operator (Wiesemann et al., 2013), hence allowing a robust value iteration algorithm to solve the infinite horizon case.

Many recent papers have developed fast algorithms for solving the robust Bellman updates required by robust value iteration. For example, Behzadian et al. (2021) studied s-rectangular ambiguity sets defined by the L_∞ norm. They developed a homotopy method that was implemented within a bisection search algorithm for solving the robust Bellman update. Ho et al. (2021) applied a similar concept to weighted L_1 norm ambiguity sets, although they used a partial policy iteration algorithm instead of value iteration. For ellipsoidal and Kullback-Leibler ambiguity sets, Grand-Clément and Kroer (2021) proposed a first order method that is embedded in robust value iteration. Their algorithm is based on the observation that solving the robust Bellman update is equivalent to solving S bilinear saddle point problems. Ho et al. (2022) studied ϕ-divergence ambiguity sets, and showed that solving the update in this case corresponds to solving a set of highly structured simplex projection problems. They used dualisation to represent each projection problem as a univariate convex optimisation problem. Different to these algorithms, Derman et al. (2021) showed that solving an s-rectangular RMDP with reward uncertainty is equivalent to solving a regularised MDP.

In general, s-rectangular ambiguity sets are common due to the tractability of the resulting RMDP. However, since the state transitions for different states are not always independent, more general ambiguity sets have also been presented in the literature. Tirinzoni et al. (2018) state that s-rectangular ambiguity sets can lead to conservative policies, and do not facilitate knowledge transfer between states or across different decision processes. They instead use non-rectangular ambiguity sets that bound the

moments of state-action features, which are taken over entire MDP trajectories and not just those for one state. These RMDPs are solved by finding the optimal policy for a mixture of non-robust MDPs. Following a similar argument with regards to the conservativeness of s-rectangular ambiguity sets, Goyal and Grand-Clement (2022) develop a new class of non-rectangular ambiguity sets: *factor matrix* ambiguity sets. Each distribution in such an ambiguity set is a convex combination of a set of common feature vectors. This ambiguity set allows for the modelling of dependence across states and for the RMDP to be efficiently solved by a hybrid value iteration-policy improvement algorithm.

This chapter studies RMDPs with s-rectangular ambiguity sets, but with one key difference from those discussed here. We study transition distributions that are parametric, and our ambiguity sets contain only distributions that lie in the same parametric family as the true transition distribution. This represents, for example, problems in which the state transitions are defined by some external random variables such as demand, and that these random variables take parametric distributions. Despite the fact that the underlying transition distributions may be parametric, in RMDPs, ambiguity sets always contain non-parametric distributions. However, any distribution in the set that is not part of the same family as the true transition distribution cannot be equal to the true distribution.

As a result of this, we formulate *parametric RMDPs*, where the ambiguity sets used contain potential parameters of the transition distribution, not potential distributions. This allows us to ensure that the worst-case distribution lies in the correct parametric

family. We then only need to find the worst-case parameter, not the entire distribution. To solve the resulting RMDP, we present three algorithms that are used inside a robust value iteration. The first two are based on discretising the ambiguity set of parameters and formulating the update as an LP with one constraint for each parameter. The second is a fast bisection search algorithm that solves simplex projection problems to compute the update, similar to the approach of Ho et al. (2022).

4.2.2 Newsvendor Models

The model that we will use to illustrate our methods is the *newsvendor model* (Arrow et al., 1951). The newsvendor model is a classical model in inventory and operations management that describes a retailer deciding on how much stock to purchase in order to meet uncertain future demand as closely as possible. The newsvendor model has the distinguishing feature that failing to meet demand in any way is penalised. If too much stock is purchased, the newsvendor pays a holding cost in order to keep that stock for future customers. If demand is not met by the stock purchased, the newsvendor pays a backorder cost in order to meet the unmet demand. Due to this, demand uncertainty and correctly modelling said uncertainty plays a strong role in maximising profits. Since the initial model of Arrow et al. (1951), the newsvendor model has been extended in many ways. The extension most relevant to this chapter is the multi-period newsvendor model (Arrow et al., 1958). This is the natural extension of the problem to the case where the newsvendor needs to meet demand in multiple time periods, and is able to make separate orders for each.

Although some papers study static newsvendor models (Matsuyama, 2006; Chen et al., 2017; Ullah et al., 2019), where the newsvendor must commit to their order quantities prior to the selling period, it is more common in the literature to consider dynamic newsvendor models. In a dynamic newsvendor model, at the start of each period in the horizon, the newsvendor selects their order quantity for that day. This way, they have exact knowledge of the amount of inventory remaining at the time of ordering, as opposed to static models where future inventory levels must be estimated beforehand in order to estimate profits and select order quantities.

Early dynamic models were finite horizon discrete DP models where base-stock policies were optimal. An example of this comes from Bouakiz and Sobel (1992), who considered the case where the demand random variables are independent and identically distributed with a known distribution. A continuous time version of the dynamic model was later solved by Kogan and Lou (2003), who showed it to be equivalent to solving a set of discrete-time problems.

Soon after its introduction, papers on the dynamic multi-period model considered more complex situations with respect to demand behaviour and knowledge about its distribution. Levi et al. (2007) developed policies based on only samples from the true demand distribution, with no assumptions being made about the distribution itself. Other extensions include models with partially observable demand (Bensoussan et al., 2007), non-stationary demand (Kim et al., 2015) and service-dependent demand (Deng et al., 2014). These papers illustrate the importance of accurate demand modelling in dynamic newsvendor models, and highlight that it is very common in such problems

for demand information to be incomplete.

Another important extension of the newsvendor problem is the *distribution free* (DF) newsvendor model (Scarf, 1957). This model represents situations where the true demand distribution is not known exactly, but some of its moments are known exactly. The model then maximises the worst-case profit over the ambiguity set containing all distributions with said moments. Since the work of Scarf (1957) for the single-period single-product DF model, the DF concept has received significant attention in the newsvendor literature. Early extensions include models with multiple products and random yield (Gallego and Moon, 1993), balking (Moon and Choi, 1995), uncertainty in cost parameters (Ouyang and Chang, 2002), shortage penalty costs and budget constraints (Alfares and Elmorra, 2005).

Later models included additional complexities such as advertising and the costs thereof (Lee and Hsu, 2011), risk- and ambiguity-aversion (Han et al., 2014) and carbon emissions (Liu et al., 2015). Due to the fact that these models are not dynamic, they can usually be solved by either KKT conditions or Cauchy-Schwarz bounds on the worst-case cost. Although much less common, the DF concept has also been applied to the multi-period model. For example, Ahmed et al. (2007) studied a DF model arising from using coherent risk measures in the objective function. The model was solved as a finite horizon DP, and it was shown that a base-stock policy was optimal. Levina et al. (2010) considered a model where the only distributional information came from aggregating the opinions of multiple experts. This work was later extended to the case with shortage penalty costs by Zhang et al. (2017). These authors framed

the problem as online learning with expert advice, as opposed to an MDP model. In addition, Ullah et al. (2019) found optimal policies for static multi-period distribution free models with moment-based ambiguity sets. As far as we are aware, Ahmed et al. (2007) is the only example of an MDP-based DF newsvendor model.

It is clear from the literature on the DF model that newsvendor models commonly lack distributional information, but many of the MDP models for multi-period newsvendor problems do not account for this. With the recent advancements in RMDPs combined with the fact that many multi-period newsvendor models are formulated as MDPs, this problem is a highly appropriate application of our methods. Our research differs from the existing newsvendor literature in two key ways.

Where the majority of the DF newsvendor literature focuses on the case where some moments of the demand distribution are known, we do not make any such assumption. Using DF methods usually entails estimating the moments that are assumed to be known, but studies have found that this can lead to various complications. For example, it has been found that this can lead to overly conservative solutions (Wang et al., 2016), suboptimal solutions (Lee et al., 2021) and poor estimates of the true cost function (Rossi et al., 2014). As such, our approach is closer to the more recent papers in RMDPs (Grand-Clément and Kroer, 2021; Ho et al., 2022), who consider distance-based ambiguity sets.

In addition, unlike these two papers, we consider parametric ambiguity sets. This allows us to model cases where the newsvendor demand is parametric, and enforce that the worst-case distribution lies in the same parametric family as the true demand

distribution. As discussed earlier, it is very common to assume that demand distributions are parametric (Nahmias, 1994; Agrawal and Smith, 1996; Gallego et al., 2007; Siegel and Wagner, 2021), but DF models do not incorporate this. Our methodology allows parametric distributions to be used, but without the pitfalls of assuming that parameter estimates are truth.

4.3 Modelling and Algorithms

In this section, we define our model and present the algorithms used to solve it. The general robust formulation is presented in Section 4.3.1. The robust value iteration algorithm is presented in Section 4.3.2. Following this, Section 4.3.3 presents ϕ-divergence based non-parametric ambiguity sets and Section 4.3.4 details how the resulting RMDP is solved. We detail these methods since they will act as benchmarks for our parametric methods. In Section 4.3.5 we formulate our parametric ambiguity sets, and in Section 4.3.6 we detail our solution algorithms.

4.3.1 General Robust Model

The RMDP we consider is formulated as follows. The *state* and *action spaces* are defined as $\mathcal{S} = \{1, \ldots, S\}$ and $\mathcal{A} = \{1, \ldots, A\}$, respectively. Decisions are made at each *epoch* $t \in \mathcal{T} = \mathbb{N}_0$. The *state* at time t is a random variable, denoted by S_t. Similarly, we denote by a_t the *action* taken at time t. The *reward* for selecting action $a \in \mathcal{A}$ when in state $s \in \mathcal{S}$ and transitioning to state $s' \in \mathcal{S}$ is given by $r_{s,a,s'} \in \mathbb{R}_+$. We denote by Δ_n the *probability simplex* in $\mathbb{R}^n$: $\Delta_n = \{\boldsymbol{P} \in \mathbb{R}^n_+ : \sum_{i=1}^n P_i = 1\}$.

The distribution of the *initial state* S_0, i.e. the state at time $t = 0$, is denoted by $\boldsymbol{Q} \in \Delta_S$. The distribution of S_{t+1} given that action a is taken in state s at time t is given by the unknown distribution $\boldsymbol{P}^0_{s,a} = (P^0_{s,a,1}, \ldots, P^0_{s,a,S}) \in \Delta_S$. Here, $P^0_{s,a,s'} = \mathbb{P}(S_{t+1} = s' | S_t = s, a_t = a)$ for any $t \geq 0$. Similarly, we write $\boldsymbol{P}^0_s$ to denote a matrix where the element on the a^{th} row and $(s')^{\text{th}}$ column is $P^0_{s,a,s'}$. Denote by $\Pi = (\Delta_A)^S$ the set of all stationary, randomised policies. A *policy* $\boldsymbol{\pi}$ is a matrix $\boldsymbol{\pi} = (\pi_{s,a})_{s \in \mathcal{S}, a \in \mathcal{A}} \in \Pi$ such that $\pi_{s,a}$ gives the probability of taking action a when in state s for each $(s, a) \in \mathcal{S} \times \mathcal{A}$ under policy $\boldsymbol{\pi}$. Denote by $\mathcal{P} \subseteq (\Delta_S)^{S \times A}$ an *ambiguity set* for $\boldsymbol{P}^0$. Each $\boldsymbol{P} \in \mathcal{P}$ and $\boldsymbol{\pi} \in \Pi$ induce a stochastic process $\{(s_t, a_t)\}_{t=0}^{\infty}$ on the space $(\mathcal{S} \times \mathcal{A})^{\infty}$ of sample paths, and $\mathbb{E}_{\boldsymbol{P}, \boldsymbol{\pi}}$ refers to the expectation w.r.t. this process. Then, the robust MDP problem is given by:

$$\max_{\boldsymbol{\pi} \in \Pi} \min_{\boldsymbol{P} \in \mathcal{P}} \mathbb{E}_{\boldsymbol{P}, \boldsymbol{\pi}} \left[\sum_{t=0}^{\infty} \gamma^t r_{s_t, a_t, s_{t+1}} \middle| S_0 \sim \boldsymbol{Q} \right], \tag{4.3.1}$$

where $\gamma \in (0, 1)$ is a *discount factor*. We consider s-rectangular ambiguity sets, which are of the form:

$$\mathcal{P} = \mathcal{P}_1 \times \ldots \times \mathcal{P}_s, \quad \mathcal{P}_s \subseteq (\Delta_S)^A \ \forall \ s \in \mathcal{S}.$$

4.3.2 Statewise Bellman Equations and Robust Value Iteration

Given an initial estimate $v^0_s \ \forall \ s \in \mathcal{S}$, robust value iteration is performed by iteratively updating the estimates using the *robust Bellman equation* (4.3.2) for $n = 0, 1, \ldots$:

$$v^{n+1}_s = \max_{\boldsymbol{\pi}_s \in \Delta_A} \min_{\boldsymbol{P}_s \in \mathcal{P}_s} \sum_{a \in \mathcal{A}} \pi_{s,a} \sum_{s' \in \mathcal{S}} P_{s,a,s'} (r_{s,a,s'} + \gamma v^n_{s'}) \ \forall \ s \in \mathcal{S}. \tag{4.3.2}$$

Adapting the pseudocode by Powell (2007), this leads to the following robust value iteration algorithm:

1. Initialise $n = 0$, $\Delta = 0$, $\boldsymbol{v}^0 = \mathbf{0}$, and select ε.
2. While $\Delta \geq \frac{\varepsilon\gamma}{1-2\gamma}$:
   (a) For each $s \in \mathcal{S}$, solve (4.3.2) to find the value of v_s^{n+1}.
   (b) Set $\Delta = ||\boldsymbol{v}^{n+1} - \boldsymbol{v}^n||$ where $||\boldsymbol{v}|| = \max_{s\in\mathcal{S}} v_s$.
   (c) Set $n = n + 1$.
3. Set $\boldsymbol{v}^* = \boldsymbol{v}^n$ and let the policy that solves (4.3.2) under $\boldsymbol{v}^n = \boldsymbol{v}^*$ be $\boldsymbol{\pi}^*$.
4. Return $\boldsymbol{\pi}^*$ and compute the optimal total reward under $\boldsymbol{\pi}^*$ as $\sum_{s\in S} Q_s v_s^*$.

Step 2(a) is referred to as solving a *robust Bellman update*.

4.3.3 ϕ-divergence Ambiguity Sets

The most common ambiguity sets in RMDPs are non-parametric, i.e. they do not make use of any information about the family of distributions in which the true distribution lies. Common non-parametric ambiguity sets are distance-based (Grand-Clément and Kroer, 2021; Ho et al., 2022). Such ambiguity sets contain only distributions that lie within a pre-prescribed maximum distance from a nominal or estimated distribution $\hat{\boldsymbol{P}}_s$. In other words, a non-parametric distance-based ambiguity set is of the form given in (4.3.3).

$$\mathcal{P}_s = \left\{ \boldsymbol{P}_s \in (\Delta_A)^S : \sum_{a\in\mathcal{A}} d_a(\boldsymbol{P}_{s,a}, \hat{\boldsymbol{P}}_{s,a}) \leq \kappa \right\} \ \forall\ s \in \mathcal{S}. \tag{4.3.3}$$

Here, $d_a : \Delta_S \times \Delta_S \to \mathbb{R}_+$ is a *distance measure*. We will consider cases where d_a is a *ϕ-divergence*, i.e. it satisfies:

$$d_a(\boldsymbol{P}_{s,a}, \hat{\boldsymbol{P}}_{s,a}) = \sum_{s'=1}^{S} \hat{P}_{s,a,s'} \phi\left(\frac{P_{s,a,s'}}{\hat{P}_{s,a,s'}}\right),$$

where $\phi : \mathbb{R}_+ \to \mathbb{R}_+$ is a *ϕ-divergence function*. With different choices of ϕ, the class of ϕ-divergences encompasses many distances measures, such as the Kullback-Leibler divergence (KLD), χ^2-divergence, and Burg entropy. As described by Ben-Tal et al. (2013), one benefit of such ambiguity sets is that we can choose κ such that $\mathcal{P}_s$ is an approximate confidence set for the true distribution.

Suppose that the true distribution for state s, $\boldsymbol{P}_s^0$, lies in a parameterised set $\{\boldsymbol{P}_s^{\boldsymbol{\theta}} \mid \boldsymbol{\theta}_s \in \Theta_s\}$, and let the true parameter be $\boldsymbol{\theta}_s^0$. We will assume that only $\boldsymbol{\theta}_s$ is required to compute $\boldsymbol{P}_s^{\boldsymbol{\theta}}$ and that $\boldsymbol{P}^{\boldsymbol{\theta}} = (\boldsymbol{P}_1^{\boldsymbol{\theta}}, \ldots, \boldsymbol{P}_S^{\boldsymbol{\theta}})$ is parameterised by $\boldsymbol{\theta} = (\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_S)$, with $\boldsymbol{\theta}_s = (\theta_{s,a,l})_{a \in \mathcal{A}, o \in \{1,\ldots,o\}}$. Also suppose that the distributions $\boldsymbol{P}_{s,a}^0$ are independent. Then, for each $(s,a) \in \mathcal{S} \times \mathcal{A}$, $\boldsymbol{P}_{s,a}^0$ is a distribution parameterised by $\boldsymbol{\theta}_{s,a}^0$. Suppose that we take N sample transitions from each $\boldsymbol{P}_{s,a}^0$ and use these to create a *maximum likelihood estimate* (MLE) $\hat{\boldsymbol{\theta}}_{s,a}$ of $\boldsymbol{\theta}_{s,a}^0$. Then, if we choose κ according to (4.3.4), the set $\mathcal{P}_s$ is an approximate $100(1-\alpha)\%$ confidence set for $\boldsymbol{P}_s^0$ around $\hat{\boldsymbol{P}}_s = \boldsymbol{P}_s^{\hat{\boldsymbol{\theta}}}$.

$$\kappa = \frac{\phi''(1)}{2N} \chi^2_{oA,1-\alpha}. \tag{4.3.4}$$

In (4.3.4), $\chi^2_{oA,1-\alpha}$ is the $100(1-\alpha)^{\text{th}}$ percentile of the χ^2 distribution with oA degrees of freedom. The degrees of freedom used is oA since o is the number of unknown parameters required to compute $\boldsymbol{P}_{s,a}$, and so oA is the number required to compute $\boldsymbol{P}_s$.

4.3.4 Solving the Robust Bellman Update

In this section, we detail the algorithms that we will use to solve the robust Bellman update under non-parametric ambiguity sets, that will act as benchmarks for our methods. Firstly, we describe how to reformulate the update using the conjugate of the ϕ-divergence function. Following this, we describe the projection-based bisection search algorithm of Ho et al. (2022).

Solution via Reformulation

The robust Bellman update problem under ϕ-divergence ambiguity sets can be reformulated using the *convex conjugate* of a ϕ-divergence function:

$$\phi^*(z) = \sup_{\tau \geq 0} \{z\tau - \phi(\tau)\}.$$

Using this definition, following the steps given by Ben-Tal et al. (2013), we dualise the inner problem of (4.3.2) to arrive at the following reformulation:

$$\max_{\boldsymbol{\pi}_s \in \Delta_A, \boldsymbol{\nu}, \eta} \left\{ \nu - \eta\kappa - \sum_{a \in \mathcal{A}} \sum_{s' \in \mathcal{S}} \eta \hat{P}_{s,a,s'} \phi^* \left(\frac{\nu_a - \pi_{s,a}(r_{s,a,s'} + \gamma v_{s'})}{\eta} \right) : \boldsymbol{\nu} \in \mathbb{R}^A, \eta \in \mathbb{R}_+ \right\} \tag{4.3.5}$$

where $\nu_a \in \mathbb{R}$ is the Lagrange multiplier for the constraint $\sum_{s' \in \mathcal{S}} P_{s,a,s'} = 1$ for each $a \in \mathcal{A}$, $\nu = \sum_{a \in \mathcal{A}} \nu_a$ and $\eta \in \mathbb{R}_+$ is the Lagrange multiplier for the constraint $\sum_{a \in \mathcal{A}} d_a(\boldsymbol{P}_{s,a}, \hat{\boldsymbol{P}}_{s,a}) \leq \kappa$. Note that here we have written $\boldsymbol{v} = \boldsymbol{v}^n$ for shorthand. For a derivation of this reformulation, see Appendix C.1.1.

The model requires different approaches for different ϕ-divergence functions, due to the different forms that ϕ^* can take. As an example, for the modified χ^2-divergence,

this model can be reformulated as the following *convex quadratic program* (CQP):

$$
\begin{aligned}
&\max_{\boldsymbol{\pi}_s} \left\{ \nu + \eta(A - \kappa) - \frac{1}{4} \sum_{a \in \mathcal{A}} \sum_{s' \in \mathcal{S}} \hat{P}_{s,a,s'} u_{s,a,s'} \right\} \\
&\text{s.t. } \sqrt{4\zeta_{s,a,s'}^2 + (\eta - u_{s,a,s'})^2} \leq (\eta + u_{s,a,s'}) \ \forall\ a \in \mathcal{A} \ \forall\ s' \in \mathcal{S} \\
&\qquad \zeta_{s,a,s'} \geq 2\eta + \nu_a - \pi_{s,a}(r_{s,a,s'} + \gamma v_{s'}) \ \forall\ a \in \mathcal{A} \ \forall\ s' \in \mathcal{S} \\
&\qquad \zeta_{s,a,s'} \geq 0 \ \forall\ a \in \mathcal{A} \ \forall\ s' \in \mathcal{S} \\
&\qquad \sum_{a \in \mathcal{A}} \pi_{s,a} = 1 \\
&\qquad \pi_{s,a} \geq 0 \ \forall\ a \in \mathcal{A} \\
&\qquad \eta \geq 0 \\
&\qquad \boldsymbol{\nu} \in \mathbb{R}^A.
\end{aligned}
\tag{4.3.6}
$$

We present this model since it will be used as a benchmark for testing our parametric methods. For more details on the derivation of this reformulation, see Appendix C.1.2.

Projection-based Bisection Search Algorithms

Model (4.3.5) can become large when A and S are large, and so it is not always reasonable to solve it in every step of the value iteration algorithm. Hence, Ho et al. (2022) presented a fast projection-based algorithm for solving the corresponding robust Bellman update. We define a *simplex projection problem* as follows:

$$
\mathfrak{P}(\hat{\boldsymbol{P}}_{s,a}; \boldsymbol{b}, \beta) = \begin{bmatrix} \min_{\boldsymbol{P}_{s,a}} & d_a(\boldsymbol{P}_{s,a}, \hat{\boldsymbol{P}}_{s,a}) \\ \text{s.t.} & \sum_{s' \in \mathcal{S}} b_{s'} P_{s,a,s'} \leq \beta \\ & \boldsymbol{P}_{s,a} \in \Delta_S \end{bmatrix}. \tag{4.3.7}
$$

Where $\boldsymbol{b}$ and β are input parameters. Given this, the outline of the algorithm presented by Ho et al. (2022) is as follows. In each iteration n of the value iteration algorithm, for each $s \in \mathcal{S}$, the Bellman update is solved to ϵ-optimality via bisection search on the value of v_s^{n+1}. This is done via the following algorithm, which we will call *non-parametric bisection search* (NBS):

1. Initialise ϵ and define $\delta = \frac{\epsilon\kappa}{2A+\bar{R}_s(\boldsymbol{v}^n)+A\epsilon}$, $\overline{v}_s^0 = \bar{R}_s(\boldsymbol{v}^n) = \frac{\max_{(a,s')\in\mathcal{A}\times\mathcal{S}} r_{s,a,s'}}{1-\gamma}$, and $\underline{v}_s^0 = \max_{a\in\mathcal{A}} \min_{s'\in\mathcal{S}} \{r_{s,a,s'} + \gamma v_{s'}^n\}$.

2. For each $i = 0, \ldots$:

   (a) Set $\beta = \frac{\overline{v}_s^i + \underline{v}_s^i}{2}$.

   (b) For each $a \in \mathcal{A}$,

      i. If $\mathfrak{P}(\hat{\boldsymbol{P}}_{s,a}; \boldsymbol{r}_{s,a} + \gamma\boldsymbol{v}^n, \beta)$ is infeasible, i.e. $\min_{s'} \{r_{s,a,s'} + \gamma v_{s'}^n\} > \beta$, then set $\underline{d}_a$ and $\overline{d}_a$ equal to $\kappa + 1$ and $\kappa + 2$ respectively. Go to step 2(c).

      ii. Otherwise, solve the projection problem $\mathfrak{P}(\hat{\boldsymbol{P}}_{s,a}; \boldsymbol{r}_{s,a} + \gamma\boldsymbol{v}^n, \beta)$ to δ-optimality to obtain parameter action-wise upper and lower bounds $\underline{d}_a, \overline{d}_a$ on its objective value.

   (c) Use these bounds to update $\underline{v}_s^i$ and $\overline{v}_s^i$:

$$(\underline{v}_s^{i+1}, \overline{v}_s^{i+1}) = \begin{cases} (\underline{v}_s^i, \beta) & \text{if } \sum_{a\in\mathcal{A}} \overline{d}_a \leq \kappa, \\ (\beta, \overline{v}_s^i) & \text{if } \sum_{a\in\mathcal{A}} \underline{d}_a > \kappa \end{cases}]$$

   (d) $\overline{v}_s^{i+1} - \underline{v}_s^{i+1} < \epsilon$ or $\kappa \in [\sum_{a\in\mathcal{A}} \underline{d}_a, \sum_{a\in\mathcal{A}} \overline{d}_a)$ then go to step 3.

3. Return $\beta = \frac{\overline{v}_s^{i+1} + \underline{v}_s^{i+1}}{2}$.

This generates the updated value estimates v_s^{n+1} for $s \in \mathcal{S}$. In order to understand how (4.3.7) is used to update the bounds on v_s^{n+1}, consider the following. Firstly, note that the objective function of (4.3.1) is linear in $\boldsymbol{\pi}_s$, meaning that there exists a deterministic optimal policy. Hence, we can replace the maximisation over $\boldsymbol{\pi}_s$ with a maximisation over $a \in \mathcal{A}$ (Ho et al., 2022). Therefore, applying the classical min-max theorem we can write v_s^{n+1} as the objective value of the following model:

$$\begin{aligned} \min_{\boldsymbol{P}_s} \quad & \max_{a\in\mathcal{A}} \left\{ \sum_{s'\in\mathcal{S}} P_{s,a,s'} \left(r_{s,a,s'} + \gamma v_{s'}^n \right) \right\} \\ \text{s.t.} \quad & \sum_{a\in\mathcal{A}} d_a(\boldsymbol{P}_{s,a}, \hat{\boldsymbol{P}}_{s,a}) \leq \kappa \\ & \boldsymbol{P}_s \in (\Delta_S)^A. \end{aligned} \tag{4.3.8}$$

Now suppose that we are in iteration i of the bisection search algorithm. Given the current midpoint β, step 2(b) solves (4.3.7) to check whether β provides an upper or a lower bound for v_s^{n+1}. The way this works is as follows. When we set $\boldsymbol{b} = \boldsymbol{r}_{s,a} + \gamma \boldsymbol{v}^n$, we have that:

$$\sum_{s'\in\mathcal{S}} b_{s'} P_{s,a,s'} = \sum_{s'\in\mathcal{S}} \left(r_{s,a,s'} + \gamma v_{s'}^n \right) P_{s,a,s'}.$$

This expression corresponds to the long-run expected reward for taking action a when in state s if the value function is given by $\boldsymbol{v}^n$. Therefore, solving (4.3.7) can be seen as finding the minimum distance that $\boldsymbol{P}_{s,a}$ has to lie from $\hat{\boldsymbol{P}}_{s,a}$ in order to ensure that selecting action a cannot obtain a reward higher than β. Now suppose that the upper bounds on the objective values of (4.3.7) for each $a \in \mathcal{A}$ satisfy $\sum_{a\in\mathcal{A}} \overline{d}_a \leq \kappa$. Then there is a distribution $\boldsymbol{P}_{s,a}$ satisfying the constraints of (4.3.8), such that $\sum_{s'\in\mathcal{S}} \left(r_{s,a,s'} + \gamma v_{s'}^n \right) P_{s,a,s'} \leq \beta \; \forall \; a \in \mathcal{A}$. Therefore, we have that $\max_{a\in\mathcal{A}} \left\{ \sum_{s'\in\mathcal{S}} \left(r_{s,a,s'} + \gamma v_{s'}^n \right) P_{s,a,s'} \right\} \leq \beta$. Since $\boldsymbol{P}_s$ is a feasible solution to (4.3.8), we must have $v_s^{n+1} \leq \beta$. On the other hand, if $\sum_{a\in\mathcal{A}} \underline{d}_a > \kappa$ then we have that any

solution $\boldsymbol{P}_s$ satisfying the inequality $\sum_{s' \in \mathcal{S}} \left(r_{s,a,s'} + \gamma v_{s'}^n\right) P_{s,a,s'} \leq \beta \ \forall \ a \in \mathcal{A}$ does not satisfy the constraints of (4.3.8). Therefore, any feasible solution $\boldsymbol{P}_s$ of (4.3.8) satisfies the inequality $\max_{a \in \mathcal{A}} \sum_{s' \in \mathcal{S}} \left(r_{s,a,s'} + \gamma v_{s'}^n\right) P_{s,a,s'} > \beta$, meaning that $v_s^{n+1} > \beta$.

Step 2(b)i specifies how the values should be updated if the problem is infeasible. The logic behind this step is as follows. By Ho et al. (2022), the projection problem is infeasible for some action $a' \in \mathcal{A}$ if and only if $\min_{s'} \left\{r_{s,a',s'} + \gamma v_{s'}^n\right\} > \beta$. Suppose the projection problem is infeasible for action a' and let $\boldsymbol{P}_s \in (\Delta_S)^A$. Then, we have:

$$\begin{aligned}
\min_{s'} \left\{r_{s,a',s'} + \gamma v_{s'}^n\right\} > \beta \implies & \ r_{s,a',s'} + \gamma v_{s'}^n > \beta \ \forall \ s' \in \mathcal{S} \\
\implies & \sum_{s' \in \mathcal{S}} P_{s,a',s'} \left(r_{s,a',s'} + \gamma v_{s'}^n\right) > \sum_{s' \in \mathcal{S}} P_{s,a',s'} \beta \\
\implies & \sum_{s' \in \mathcal{S}} P_{s,a',s'} \left(r_{s,a',s'} + \gamma v_{s'}^n\right) > \beta \\
\implies & \max_{a \in \mathcal{A}} \left\{ \sum_{s' \in \mathcal{S}} P_{s,a,s'} \left(r_{s,a,s'} + \gamma v_{s'}^n\right) \right\} > \beta.
\end{aligned}$$

Therefore, every feasible solution of (4.3.8) has an objective value greater than β, and so $v_s^{n+1} > \beta$. Hence, the case when the projection problem is infeasible for any $a \in \mathcal{A}$ should be treated in the same way as when $\sum_{a \in \mathcal{A}} \underline{d}_a > \kappa$. To this effect, we set each $\underline{d}_a, \overline{d}_a$ to some arbitrary numbers above κ, say $\kappa + 1$ and $\kappa + 2$. Note that the value these bounds are set to do not matter, since all we need is that $\sum_{a \in \mathcal{A}} \underline{d}_a > \kappa$ so that β becomes a lower bound on v_s^{n+1}.

In step 2(b)ii, the projection problem is also usually solved via bisection search, and for some ambiguity sets defined by ϕ-divergences such as the Kullback-Leibler divergence and χ^2-divergence, Ho et al. (2022) showed how to solve the projection problem efficiently. For the modified χ^2-divergence, their method involves first dividing the

projection problem into $S + 1$ subproblems, and then reformulating each one as a univariate optimisation problem with at most 3 potential optimal solutions that can be found analytically. Solving the subproblem then corresponds to evaluating each of these potential solutions, and choosing the best of those that are feasible. Then, the subproblems' solutions are compared and the best one is selected. Details of this algorithm can be found in Appendix C.2.1. Following the completion of the value iteration algorithm, a policy must be retrieved. Since the algorithm of Ho et al. (2022) does not return a policy, it must be extracted from solving (4.3.5), using $\boldsymbol{v} = \boldsymbol{v}^*$.

4.3.5 Parametric Ambiguity Sets

We now present our formulation for the RMDP under parametric transition distributions. Suppose that the true transition distribution $\boldsymbol{P}^0$ is uniquely defined by the probability mass function (PMF) and/or cumulative distribution function (CDF) of a parametric probability distribution. In this section, we detail how our model allows us to enforce that the worst-case distribution maintains this structure.

Formulation

Suppose that, for each $(s, a) \in \mathcal{S} \times \mathcal{A}$, $\boldsymbol{P}^0_{s,a}$ is uniquely defined by the distribution of some exogenous random variable $X_{s,a}$ with support set $\mathcal{X}_{s,a}$. Let $f_{X_{s,a}}$ and $F_{X_{s,a}}$ be the PMF and CDF of $X_{s,a}$, which are parameterised by the parameter $\boldsymbol{\theta}^0_{s,a} = (\theta^0_{s,a,1}, \ldots, \theta^0_{s,a,o})$. Assume that the current state $S_t = s$ and action $a_t = a$ are given. We assume that the next state S_{t+1} is specified by some simple, known function g of

the exogenous random variable $X_{s,a}$:

$$S_{t+1} = g(X_{s,a}|s,a),$$

which can be referred to as a *transition function*. In other words, for a given realisation x of $X_{s,a}$, we can compute the next state as $s_{t+1} = g(x|s,a)$. We define the set of all realisations of $X_{s,a}$ that lead to $S_{t+1} = s'$ as:

$$\mathcal{X}_{s,a}(s') = \{x \in \mathcal{X}_{s,a} : g(x|s,a) = s'\}.$$

Then, the transition matrix corresponding to the parameter $\boldsymbol{\theta}^0$ is given by:

$$\begin{aligned} P^0_{s,a,s'} &= \mathbb{P}(S_{t+1} = s'|S_t = s, a_t = a) \\ &= \mathbb{P}(g(X_{s,a}|s,a) = s') \\ &= \sum_{x \in \mathcal{X}_{s,a}(s')} f_{X_{s,a}}(x|\boldsymbol{\theta}^0_{s,a}) \ \forall \ s' \in \mathcal{S}. \end{aligned}$$

Since g is known, in this case the value of $\boldsymbol{P}^0$ is uniquely specified by $\boldsymbol{\theta}^0$. Therefore, the only unknown element required to find the true distribution is $\boldsymbol{\theta}^0$. Hence, given that the worst-case distribution should maintain the structure of $\boldsymbol{P}^0$, we can simply construct ambiguity sets for $\boldsymbol{\theta}^0$. More specifically, we consider s-rectangular parametric ambiguity sets of the form:

$$\Theta_s \subseteq \mathbb{R}^o \ \forall \ s \in \mathcal{S}, \quad \Theta = \Theta_1 \times \ldots \times \Theta_S.$$

We can then reformulate the RMDP as:

$$\max_{\boldsymbol{\pi} \in \Pi} \min_{\boldsymbol{\theta} \in \Theta} \mathbb{E}_{\boldsymbol{\theta}, \boldsymbol{\pi}} \left[\sum_{t=0}^{\infty} \gamma^t r_{s_t, a_t, s_{t+1}} \middle| S_0 \sim \boldsymbol{Q} \right]. \tag{4.3.9}$$

Let $\boldsymbol{P}^{\boldsymbol{\theta}}$ represent the transition probabilities corresponding to $\boldsymbol{\theta}$. Similarly, for any $s \in \mathcal{S}$ and $\boldsymbol{\theta}_s \in \Theta_s$, write $\boldsymbol{P}^{\boldsymbol{\theta}}_s = (P^{\boldsymbol{\theta}}_{s,a,s'})_{a \in \mathcal{A}, s' \in \mathcal{S}}$. Note that, although the superscript

for $\boldsymbol{P}_s^{\boldsymbol{\theta}}$ is $\boldsymbol{\theta}$, only $\boldsymbol{\theta}_s$ is required to compute it and by rectangularity we can obtain $\boldsymbol{P}^{\boldsymbol{\theta}}$ simply by obtaining $\boldsymbol{P}_s^{\boldsymbol{\theta}}$ for all $s \in \mathcal{S}$. Similarly, only $\boldsymbol{\theta}_{s,a}$ is required to compute $\boldsymbol{P}_{s,a}^{\boldsymbol{\theta}} = (P_{s,a,s'}^{\boldsymbol{\theta}})_{s' \in \mathcal{S}}$. Now, using the information about $\boldsymbol{P}^0$'s structure, we compute $\boldsymbol{P}_s^{\boldsymbol{\theta}}$ according to:

$$P_{s,a,s'}^{\boldsymbol{\theta}} = \sum_{x \in \mathcal{X}_{s,a}(s')} f_{X_{s,a}}(x|\boldsymbol{\theta}_{s,a}) \quad \forall\ (a, s') \in \mathcal{A} \times \mathcal{S}.$$

The parametric robust state-wise Bellman equation can then be written as:

$$v_s^{n+1} = \max_{\boldsymbol{\pi}_s \in \Delta_A} \min_{\boldsymbol{\theta}_s \in \Theta_s} \sum_{a \in \mathcal{A}} \pi_{s,a} \sum_{s' \in \mathcal{S}} P_{s,a,s'}^{\boldsymbol{\theta}} (r_{s,a,s'} + \gamma v_{s'}^n)\ \forall\ s \in \mathcal{S}. \tag{4.3.10}$$

As discussed in Chapter 2, the non-linearities of the PMFs as functions of the parameters mean that above model is not readily solvable as a mathematical program if the parameters are treated as decision variables. One way to find v_s^{n+1} approximately is to use a discretisation Θ_s' of the ambiguity set Θ_s. This allows us to reformulate the problem in (4.3.10) as:

$$v_s^{n+1} = \max_{\boldsymbol{\pi}_s \in \Delta_A} \left\{ \vartheta : \vartheta \leq \sum_{a \in \mathcal{A}} \pi_{s,a} \sum_{s' \in \mathcal{S}} P_{s,a,s'}^{\boldsymbol{\theta}} (r_{s,a,s'} + \gamma v_{s'}^n)\ \forall\ \boldsymbol{\theta}_s \in \Theta_s' \right\} \forall\ s \in \mathcal{S}. \tag{4.3.11}$$

This problem can be solved as an LP with $|\Theta_s'| + 1$ constraints. Due to this, if a fine discretisation of Θ_s' is used, this model can be very slow to solve. While this is the approach used in parametric DRO in Chapters 2 and 3, in robust value iteration we only need to compute v_s^{n+1} and not the optimal policy itself. Hence, in certain cases, no mathematical programming formulation is necessary. We will discuss this in more detail in Section 4.3.6. However, please note that solving (4.3.11) is currently the only way to extract the optimal policy and worst-case probabilities, to the best of our knowledge.

Confidence Sets for the True Parameter

We assume that we have access to N samples from the true distribution of $\boldsymbol{X}$, i.e. the distribution that characterises $\boldsymbol{P}^0$. This allows us to create an MLE $\hat{\boldsymbol{\theta}}$ of the true parameter $\boldsymbol{\theta}^0$. In addition, by standard results in maximum likelihood theory (Millar, 2011) we have:

$$\left(\hat{\boldsymbol{\theta}}_{s,a} - \boldsymbol{\theta}^0_{s,a}\right)^T I_{\mathbb{E}}\left(\boldsymbol{\theta}^0_{s,a}\right)\left(\hat{\boldsymbol{\theta}}_{s,a} - \boldsymbol{\theta}^0_{s,a}\right) \sim \chi^2_o$$

approximately, for large N. Here, $I_{\mathbb{E}}\left(\boldsymbol{\theta}^0_{s,a}\right)$ is the expected *Fisher information matrix*, which is defined by (4.3.12). In (4.3.12), ℓ is the log-likelihood function for the observed data.

$$I_{\mathbb{E}}\left(\boldsymbol{\theta}_{s,a}\right) = \left(-\mathbb{E}_{X_{s,a}}\left[\frac{\partial}{\partial\theta_{s,a,i}\partial\theta_{s,a,j}}\ell(\boldsymbol{\theta_{s,a}})\right]\right)_{i,j=1,\ldots,o}. \tag{4.3.12}$$

By independence of the random variables $X_{s,a}$ for $a \in \mathcal{A}$, we have that:

$$\sum_{a\in\mathcal{A}}\left(\hat{\boldsymbol{\theta}}_{s,a} - \boldsymbol{\theta}^0_{s,a}\right)^T I_{\mathbb{E}}\left(\boldsymbol{\theta}^0_{s,a}\right)\left(\hat{\boldsymbol{\theta}}_{s,a} - \boldsymbol{\theta}^0_{s,a}\right) \sim \chi^2_{oA}.$$

Since the two are asymptotically equivalent, we can replace $I_{\mathbb{E}}\left(\boldsymbol{\theta}^0_{s,a}\right)$ with $I_{\mathbb{E}}\left(\hat{\boldsymbol{\theta}}_{s,a}\right)$. Therefore, an approximate $100(1-\alpha)\%$ confidence set for θ_s is given by:

$$\Theta^\alpha_s = \left\{\boldsymbol{\theta}_s \in \mathbb{R}^A \times \mathbb{R}^o : \sum_{a\in\mathcal{A}}\left(\hat{\boldsymbol{\theta}}_{s,a} - \boldsymbol{\theta}_{s,a}\right)^T I_{\mathbb{E}}\left(\hat{\boldsymbol{\theta}}_{s,a}\right)\left(\hat{\boldsymbol{\theta}}_{s,a} - \boldsymbol{\theta}_{s,a}\right) \leq \chi^2_{oA,1-\alpha}\right\}.$$

In our experiments, we will use Θ^α_s as an ambiguity set for our parametric model, for each $s \in \mathcal{S}$. We will refer to a discretisation of this set as $(\Theta^\alpha_s)'$.

4.3.6 Solving the Parametric Robust Bellman Update

It is often cited (e.g., by Ho et al. (2022)) that solving an infinite-horizon RMDP efficiently boils down to being able to solve the robust Bellman update efficiently. In

our parametric formulation, if we use the LP approximation, then the model that we solve in each iteration for each state $s \in \mathcal{S}$ is the LP (4.3.11), which has $|(\Theta_s^\alpha)'| + 1$ constraints. However, depending on the fineness of the discretisation used to construct $(\Theta_s^\alpha)'$, this set can impose thousands of constraints on the model. Hence, (4.3.11) can be slow to solve. For this reason, we develop two algorithms for solving the robust Bellman update under parametric transition distributions.

A Cutting Surface Algorithm

In Chapters 2 and 3, cutting surface (CS) algorithms performed very well at solving parametric DRO problems that are formulated using discrete ambiguity sets in the same way as (4.3.11). Hence, we now describe the CS algorithm that we will use for the RMDP. The idea behind the CS algorithm is as follows. Suppose we are at iteration n of the value iteration algorithm and currently solving for state $s \in \mathcal{S}$. Start with some initial singleton subset $\Theta_s^1 = \{\boldsymbol{\theta}_s^{\text{init}}\}$. Solve (4.3.11) using $\Theta_s = \Theta_s^1$ to generate a policy $\boldsymbol{\pi}_s^1$. Next, solve the *distribution separation problem* (4.3.13) with $k = 1$ to find the worst-case parameter $\boldsymbol{\theta}_s^1$ for the policy $\boldsymbol{\pi}_s^1$. Set $\Theta_s^2 = \Theta_s^1 \cup \{\boldsymbol{\theta}_s^1\}$ and repeat until stopping criteria are met.

$$\min_{\boldsymbol{\theta}_s \in (\Theta_s^\alpha)'} \sum_{a \in \mathcal{A}} \pi_{s,a}^k \sum_{s' \in \mathcal{S}} P_{s,a,s'}^{\boldsymbol{\theta}} (r_{s,a,s'} + \gamma v_{s'}^n) \tag{4.3.13}$$

The appeal of this algorithm is that it only ever solves the approximate robust Bellman update (4.3.11) over some small subset Θ_s^k of $(\Theta_s^\alpha)'$, meaning that the LP concerned only has $|\Theta_s^k| + 1 = k+1$ constraints at iteration k. Typically, in our previous research, we found that this algorithm typically never runs for more than $k = 5$ iterations. A

formal description of the algorithm for iteration n of the value iteration algorithm for state s is given below.

1. Initialise $\Theta_s^1 = \{\boldsymbol{\theta}_s^{\text{init}}\}$ for some $\boldsymbol{\theta}_s^{\text{init}} \in (\Theta_s^\alpha)'$ and optimality tolerance $\bar{\varepsilon}$. Set $k = 1$.

2. While $k \leq k^{\max}$:

   (a) Solve the LP (4.3.11) using $\Theta_s = \Theta_s^k$ to obtain policy $\boldsymbol{\pi}_s^k$, which has a worst-case reward of $\tilde{R}^k$ over Θ_s^k.

   (b) Evaluate the worst-case rewards:

   $$R(\boldsymbol{\pi}_s^k|\boldsymbol{\theta}_s) = \sum_{a\in\mathcal{A}} \pi_{s,a}^k \sum_{s'\in\mathcal{S}} P_{s,a,s'}^{\boldsymbol{\theta}}(r_{s,a,s'} + \gamma v_{s'}^n) \;\forall\; \boldsymbol{\theta}_s \in (\Theta_s^\alpha)',$$

   and find $\boldsymbol{\theta}_s^k = \text{argmax}_{\boldsymbol{\theta}_s\in(\Theta_s^\alpha)'} R(\boldsymbol{\pi}_s^k|\boldsymbol{\theta}_s)$. Set $R^k = R(\boldsymbol{\pi}_s^k|\boldsymbol{\theta}_s^k)$.

   (c) If $\tilde{R}^k \leq R^k + \frac{\bar{\varepsilon}}{2}$ or $\boldsymbol{\theta}_s^k \in \Theta_s^k$ then set $k = k^{\max} + 1$.

3. Return $\boldsymbol{\pi}_s^k$ with worst-case parameter $\boldsymbol{\theta}_s^k$ and worst-case reward R^k.

For this chapter, this algorithm will serve as a method for solving the approximate robust Bellman update (4.3.11). It will therefore be embedded into step 2(a) of the robust value iteration algorithm in Section 4.3.2.

A Projection-based Algorithm for Single Parameter Distributions

The main algorithms of Ho et al. (2022) are based around solving the robust Bellman update using bisection search. Within each iteration of the bisection algorithm, a set of $|\mathcal{A}|$ simplex projection problems (4.3.7) are solved to generate the next upper and lower bounds on the value function. The benefit of this is that the projection

problem, in the non-parametric case with ϕ-divergence ambiguity sets, can often be reformulated as a univariate convex optimisation problem.

Solving the projection problem $\mathfrak{P}(\hat{\boldsymbol{P}}_{s,a}; \boldsymbol{b}, \beta)$ corresponds to finding the closest distribution to $\hat{\boldsymbol{P}}_{s,a}$ that yields an objective value that is no larger than β, when action a is taken in state s. In the case of distributions where $\boldsymbol{P}^0_{s,a}$ is parametrised by only one parameter (such as when $X_{s,a}$ is binomial with a fixed number of trials, or Poisson), the parametric equivalent of this problem can be stated as:

$$\tilde{\mathfrak{P}}(\hat{\theta}_{s,a}; \boldsymbol{b}, \beta) = \left[\begin{array}{ll} \min_{\theta_{s,a}} & \left(\hat{\theta}_{s,a} - \theta_{s,a}\right)^2 I_{\mathbb{E}}\left(\hat{\theta}_{s,a}\right) \\ \text{s.t.} & \sum_{s' \in \mathcal{S}} b_{s'} P^{\boldsymbol{\theta}}_{s,a,s'} \leq \beta \\ & \theta_{s,a} \in [\theta^{\min}_{s,a}, \theta^{\max}_{s,a}] \end{array} \right], \tag{4.3.14}$$

If $\sum_{s' \in \mathcal{S}} b_{s'} P^{\hat{\boldsymbol{\theta}}}_{s,a,s'} \leq \beta$ then the model is trivially solved by $\theta_{s,a} = \hat{\theta}_{s,a}$ with an objective value of 0. Therefore, suppose that $\sum_{s' \in \mathcal{S}} b_{s'} P^{\hat{\boldsymbol{\theta}}}_{s,a,s'} > \beta$. Without any type of reformulation, the model is a univariate optimisation problem that can be solved via bisection. The only complication in solving this problem via bisection is the constraint $\sum_{s' \in \mathcal{S}} b_{s'} P^{\boldsymbol{\theta}}_{s,a,s'} \leq \beta$. Note that, since $I^{-1}_{\mathbb{E}}(\hat{\theta}_{s,a})$ is the asymptotic variance of the MLE $\hat{\theta}_{s,a}$, we have that $I_{\mathbb{E}}(\hat{\theta}_{s,a}) \geq 0$. Hence, since $I_{\mathbb{E}}(\hat{\theta}_{s,a})$ is constant in $\theta_{s,a}$, the objective of (4.3.14) is equivalent to:

$$\min_{\theta_{s,a}} |\hat{\theta}_{s,a} - \theta_{s,a}|.$$

Therefore, it is clear that the optimal solution to (4.3.14) is the closest $\theta_{s,a}$ to $\hat{\theta}_{s,a}$ in terms of absolute value that satisfies $\sum_{s' \in \mathcal{S}} b_{s'} P^{\boldsymbol{\theta}}_{s,a,s'} \leq \beta$. Since $\sum_{s' \in \mathcal{S}} b_{s'} P^{\hat{\boldsymbol{\theta}}}_{s,a,s'} > \beta$, the optimal solution must satisfy $\sum_{s' \in \mathcal{S}} b_{s'} P^{\boldsymbol{\theta}}_{s,a,s'} = \beta$. To see this, observe that any feasible solution with $\sum_{s' \in \mathcal{S}} b_{s'} P^{\boldsymbol{\theta}}_{s,a,s'} < \beta$ must be further left or right of $\hat{\theta}_{s,a}$ than a

solution with $\sum_{s' \in \mathcal{S}} b_{s'} P^{\boldsymbol{\theta}}_{s,a,s'} = \beta$. Suppose that the problem is feasible and let $\theta^{\min}_{s,a}$ and $\theta^{\max}_{s,a}$ be global lower and upper bounds on $\theta_{s,a}$. Then, there must be at least one $\theta_{s,a} \in [\theta^{\min}_{s,a}, \theta^{\max}_{s,a}]$ such that $\sum_{s' \in \mathcal{S}} b_{s'} P^{\boldsymbol{\theta}}_{s,a,s'} = \beta$. Based on this, we have three potential scenarios as discussed below:

1. There exists a root of $\sum_{s' \in \mathcal{S}} b_{s'} P^{\boldsymbol{\theta}}_{s,a,s'} = \beta$ in $[\theta^{\min}_{s,a}, \hat{\theta}_{s,a}]$. Let $\theta^{l}_{s,a}$ be the closest root of $\sum_{s' \in \mathcal{S}} b_{s'} P^{\boldsymbol{\theta}}_{s,a,s'} = \beta$ to $\hat{\theta}_{s,a}$ in the interval $[\theta^{\min}_{s,a}, \hat{\theta}_{s,a}]$.

2. There exists a root of $\sum_{s' \in \mathcal{S}} b_{s'} P^{\boldsymbol{\theta}}_{s,a,s'} = \beta$ in $[\hat{\theta}_{s,a}, \theta^{\max}_{s,a}]$. Let $\theta^{u}_{s,a}$ be the closest root of $\sum_{s' \in \mathcal{S}} b_{s'} P^{\boldsymbol{\theta}}_{s,a,s'} = \beta$ to $\hat{\theta}_{s,a}$ in the interval $[\hat{\theta}_{s,a}, \theta^{\max}_{s,a}]$.

3. $\theta^{l}_{s,a}$ and $\theta^{u}_{s,a}$ both exist as defined above.

Solving the projection problem then amounts to finding $\theta^{l}_{s,a}$ and $\theta^{u}_{s,a}$, and checking which is closest to $\hat{\theta}_{s,a}$. Given this, we solve our projection problem to δ-optimality for a given s, a using the following algorithm:

1. Initialise a gap $\tilde{\epsilon}$, the set of root containing intervals as $\rho = \emptyset$, and upper and lower bounds on $\theta_{s,a}$ as $\theta^{\min}_{s,a}, \theta^{\max}_{s,a}$.

2. Find interval containing closest left root:

   (a) Initialise $\theta_{s,a} = \hat{\theta}_{s,a}$, $E = \beta$.

   (b) While $E \geq \beta$ and $\theta_{s,a} \neq \theta^{\min}_{s,a}$:

      i. Set $\theta_{s,a} = \max\{\theta_{s,a} - \tilde{\epsilon}, \theta^{\min}_{s,a}\}$.

      ii. Compute $\boldsymbol{P}^{\theta}_{s,a}$ and set $E = \sum_{s' \in \mathcal{S}} b_{s'} P^{\boldsymbol{\theta}}_{s,a,s'}$.

   (c) If $E \leq \beta$ then set $\rho = \rho \cup \{[\theta_{s,a}, \theta_{s,a} + \tilde{\epsilon}]\}$.

3. Find interval containing closest right root:

   (a) Initialise $\theta_{s,a} = \hat{\theta}_{s,a}$, $E = \beta$.

   (b) While $E \geq \beta$ and $\theta_{s,a} \neq \theta_{s,a}^{\max}$:

      i. Set $\theta_{s,a} = \min\left\{\theta_{s,a} + \tilde{\epsilon}, \theta_{s,a}^{\max}\right\}$.

      ii. Compute $\boldsymbol{P}_{s,a}^{\theta}$ and set $E = \sum_{s' \in \mathcal{S}} b_{s'} P_{s,a,s'}^{\boldsymbol{\theta}}$.

   (c) If $E \leq \beta$ then set $\rho = \rho \cup \{[\theta_{s,a} - \tilde{\epsilon}, \theta_{s,a}]\}$.

4. Carry out a bisection search in each interval in ρ to find the roots $\theta_{s,a}^{l}$ and $\theta_{s,a}^{u}$, stopping once the difference between the upper and lower bounds on the objective function in the bisection interval is no larger than δ. Store the intervals $\left[\underline{\theta}_{s,a}^{x}, \overline{\theta}_{s,a}^{x}\right]$ for $x \in \{l, u\}$.

5. Return the interval $\left[\underline{\theta}_{s,a}^{*}, \overline{\theta}_{s,a}^{*}\right]$ whose midpoint is closest to $\hat{\theta}_{s,a}$ in terms of absolute value.

We use an iterative procedure starting from $\hat{\theta}_{s,a}$ in steps 2 and 3 in order to reduce the number of times we need to compute $\boldsymbol{P}_{s,a}^{\theta}$. Since we are only interested in the closest roots to $\hat{\theta}_{s,a}$, there is no need to enumerate all intervals of width $\tilde{\epsilon}$. Note that, in some cases, $\theta_{s,a}$ may not have both a global lower and upper bound. For example, if $\theta_{s,a}$ is a Poisson parameter, then it has no upper bound. However, if the solution to the projection problem does not lie in the ambiguity set, then it is treated the same as if the problem is infeasible. Hence, we are only interested in roots inside the ambiguity set and so in such cases we can use the bounds from the ambiguity set. If $\theta_{s,a}$ is a binomial parameter then we can pick $\theta_{s,a}^{\min}, \theta_{s,a}^{\max}$ either to be $0, 1$ or

$\min((\Theta_s^\alpha)'), \max((\Theta_s^\alpha)')$. Since we will split the interval $[\theta_{s,a}^{\min}, \theta_{s,a}^{\max}]$ into an equal number of sub-intervals and hence each choice results in the same amount of computation, which upper and lower bounds we pick are not of particular importance.

Given the above, we adapt the non-parametric bisection search algorithm from Section 4.3.4 into the following parametric algorithm, which we call *parametric bisection search* (PBS). PBS solves the parametric robust Bellman update (4.3.10) to $\tilde{\epsilon}$-optimality without discretising Θ_s^α.

1. Initialise ϵ and $\delta = \frac{\epsilon\kappa}{2A+\bar{R}_s(\boldsymbol{v}^n)+A\epsilon}$, $\overline{v}_s^0 = \bar{R}_s(\boldsymbol{v}^n) = \frac{\max_{(a,s')\in\mathcal{A}\times\mathcal{S}} r_{s,a,s'}}{1-\gamma}$, and $\underline{v}_s^0 = \max_{a\in\mathcal{A}} \min_{s'\in\mathcal{S}} \{r_{s,a,s'} + \gamma v_{s'}^n\}$.

2. For each $i = 0, \ldots$:

   (a) Set $\beta = \frac{\overline{v}_s^i + \underline{v}_s^i}{2}$.

   (b) For each $a \in \mathcal{A}$,

      i. If $\bar{\mathfrak{P}}(\hat{\theta}_{s,a}; \boldsymbol{r}_{s,a} + \gamma\boldsymbol{v}^n, \beta)$ is infeasible, i.e. $\min_{s'} \{r_{s,a,s'} + \gamma v_{s'}^n\} > \beta$, then set $\underline{c}_a$ and $\overline{c}_a$ equal to $\chi^2_{oA,1-\alpha} + 1$. Go to step 2(c).

      ii. Otherwise, solve the projection problem $\bar{\mathfrak{P}}(\hat{\theta}_{s,a}; \boldsymbol{r}_{s,a} + \gamma\boldsymbol{v}^n, \beta)$ to δ-optimality to obtain parameter action-wise upper and lower bounds $\underline{c}_a, \overline{c}_a$ on its objective value. If projection algorithm returns no solutions, set these values to $\chi^2_{oA,1-\alpha} + 1$ and $\chi^2_{oA,1-\alpha} + 2$, respectively.

(c) Use these bounds to update $\underline{v}_s^i$ and $\overline{v}_s^i$:

$$(\underline{v}_s^{i+1}, \overline{v}_s^{i+1}) = \begin{cases} (\underline{v}_s^i, \beta) & \text{if } \sum_{a\in\mathcal{A}} \overline{c}_a \leq \chi^2_{oA,1-\alpha}, \\ (\beta, \overline{v}_s^i) & \text{if } \sum_{a\in\mathcal{A}} \underline{c}_a > \chi^2_{oA,1-\alpha} \end{cases}$$

(d) $\overline{v}_s^{i+1} - \underline{v}_s^{i+1} < \epsilon$ or $\chi^2_{oA,1-\alpha} \in [\sum_{a\in\mathcal{A}} \underline{c}_a, \sum_{a\in\mathcal{A}} \overline{c}_a)$ then go to step 3.

3. Return $\beta = \frac{\overline{v}_s^i + \underline{v}_s^i}{2}$.

The logic behind this algorithm is the same as that of NBS from Section 4.3.4, but replacing the non-parametric problems with their parametric counterparts. Given this algorithm, we can efficiently carry out value iteration without ever needing a solver. However, after this is complete, the optimal policy will need to be retrieved by solving the approximate MIP reformulation (4.3.11) of the robust Bellman update. This can be done using the cutting surface algorithm of Section 4.3.6.

4.4 A Capacitated Dynamic Multi-period Newsvendor Problem

As an example problem, we consider a dynamic multi-period newsvendor problem. This version of the problem has discrete demands and actions, and a capacity limiting the amount that can be held in inventory for any given period. In Section 4.4.1, we describe the model in detail. Then, in Section 4.4.2, we formulate the model under binomial demands and perform computational experiments to test our algorithms in this case. Finally, in Section 4.4.3, we formulate the model and test our algorithms under Poisson demands.

4.4.1 Model

Suppose that S_t represents the amount of *inventory* in a system of some product affected by uncertain demand. Let, a_t be the amount of this product to order at the start of period t, to be sold during period t. Products are delivered immediately. We assume that there is a *capacity* C for holding stock in inventory, so that $\mathcal{S} = \{0, \ldots, C\}$. Given that action a is taken in state s, if $s + a > C$ then any excess product is lost as it cannot be stored. Although the newsvendor could technically order infinite stock, they have no reason to. Hence, $\mathcal{A} = \{0, \ldots, C\}$, and so $S = |\mathcal{S}| = C+1$ and $A = |\mathcal{A}| = C + 1$. We assume that every unit of stock that must be held for a period incurs a *holding cost* of h, and if the newsvendor runs out of stock then they pay a *stockout cost* of b'. Furthermore, assume that one unit of stock sells for c and is purchased for $w < c$. Let the demand for the product, $X_{s,a}$, be a random variable whose distribution is parameterised by the unknown parameter $\theta^0_{s,a}$ for each $(s, a) \in \mathcal{S} \times \mathcal{A}$. Then, given $S_t = s$ and $a_t = a$, we have:

$$S_{t+1} = \max\{0, \min\{s + a, C\} - X_{s,a}\}.$$

For shorthand, let $\bar{s} = \min\{s + a, C\}$ be the *post-action pre-demand state*. Then, we have that $g(x|s, a) = \max\{0, \bar{s} - x\}$, and therefore:

$$\mathcal{X}_{s,a}(s') = \begin{cases} \{\bar{s}, \bar{s} + 1, \ldots, C - 1, C\} & \text{if } s' = 0 \\ \bar{s} - s' & \text{if } s' > 0. \end{cases}$$

Therefore, the transition distribution satisfies:

$$P^0_{s,a,s'} = \begin{cases} \sum_{x=\bar{s}}^{C} f_{X_{s,a}}(x|\boldsymbol{\theta}^0_{s,a}) & \text{if } s' = 0, \\ f_{X_{s,a}}(\bar{s} - s'|\boldsymbol{\theta}^0_{s,a}) & \text{if } s' > 0. \end{cases}$$

$$= \begin{cases} 1 - \sum_{x=0}^{\bar{s}-1} f_{X_{s,a}}(x|\boldsymbol{\theta}^0_{s,a}) & \text{if } s' = 0, \\ f_{X_{s,a}}(\bar{s} - s'|\boldsymbol{\theta}^0_{s,a}) & \text{if } s' > 0. \end{cases}$$

The reward for taking action a in state s and moving to state s' is given by the following. Define the event of a stockout as $\mathbb{1}\{s' = 0\}$. Then, the rewards are:

$$r_{s,a,s'} = c \max\{\bar{s} - s', 0\} - wa - h(\bar{s} - \max\{\bar{s} - s', 0\}) - b'\mathbb{1}\{s' = 0\}. \tag{4.4.1}$$

The term $b'\mathbb{1}\{s' = 0\}$ will charge the newsvendor a flat cost of b' whenever they miss demand. It is more common in the newsvendor literature to incur a backorder cost for every unit of missed demand, representing the newsvendor paying an additional cost to meet this demand after initially not meeting it. This would involve adding cost of $b' \max\{X_{s,a} - \bar{s}, 0\}$ instead of $b'\mathbb{1}\{s' = 0\}$. Hence, the rewards would depend on $X_{s,a}$ and we need to formulate the robust Bellman update (4.3.2) in a different fashion. The main change would be that the distribution of $X_{s,a}$ would be required to calculate the expected rewards as opposed to simply the transition matrix. Hence, we would replace the inner minimisation over $\boldsymbol{P}_s$ with a minimisation over candidates $\boldsymbol{P}'_s \in \mathcal{P}'_s$ for the true distribution of demand $\boldsymbol{X}_s$. We would then replace the inner expected value with respect to the next state with an expectation w.r.t. $X_{s,a}$.

Since each $\boldsymbol{P}'_{s,a}$ has dimension $|\mathcal{X}_{s,a}|$, this would remain affect solvability in the non-parametric case for finite support demand random variables. However, since we do not

know any moments of the distribution of $X_{s,a}$, it would result in an infinite number of decision variables for infinite support demand random variables. This would not affect the parametric model, however, which would still find the worst-case parameter directly. For more details on the parametric and non-parametric reformulations in the case of a backorder cost, see Appendix C.3.

Considering a stockout cost instead of a backorder cost means that the robust Bellman update can be computed via an expectation over the finite set $\mathcal{S}$, regardless of whether or not $\mathcal{X}_{s,a}$ is finite. The downside of this formulation is that it can penalise the newsvendor for meeting demand exactly. However, if this is a concern then one can set $w + h > b'$ to ensure that the newsvendor would still prefer to meet demand exactly and pay a stockout cost than to purchase too much stock and hold one item for the following period. Also, it is important to note that shortage costs are implicitly represented in this model via missed profits, and the newsvendor can see how much demand was lost after the period is complete.

4.4.2 Numerical Experiments with Binomial Demands

To examine the efficacy of the algorithms described in this chapter, we now carry out numerical experiments on the dynamic newsvendor problem. Firstly, we describe the binomial ambiguity sets used. Then, we describe the parameters used. Following this, we discuss the times taken by each algorithm to finish value iteration and compute the optimal policy. Finally, we compare the parametric and non-parametric value functions and resulting policies.

Ambiguity Sets

Suppose that $X_{s,a} \sim \text{Bin}(C, p^0_{s,a})$ for $(s,a) \in \mathcal{S} \times \mathcal{A}$, and hence:

$$f_{X_{s,a}}(x|p^0_{s,a}) = \binom{C}{x}(p^0_{s,a})^x(1-p^0_{s,a})^{C-x} \quad (x \in \{0,\ldots,C\}).$$

We set the number of trials as C for the following reasons. Since a binomial random variable is bounded above by the number of trials, binomial demands might correspond to a scenario where a restriction is placed on demand by the newsvendor. In this case, the number of trials represents the maximum demand allowed by the newsvendor before no more orders are allowed. The number of trials provides a way for the newsvendor to limit the amount of unmet demand that is possible. Since any demand above C is guaranteed to be unmet regardless of the current stock levels, it is not reasonable for the number of trials to be set above C. This would not have any benefit for the newsvendor or the customers.

Another logical choice for the number of trials may be $\min\{s+a, C\}$. However, this would imply that the newsvendor would need to update the upper bound on demand after every order, and this information would need to be conveyed to customers. In addition, it suggests that the newsvendor is always able to meet demand exactly, which is not a realistic modelling assumption. Also, the newsvendor would be unable to observe how much demand was lost or if the demand met the capacity, which is inconvenient for improving their decision-making and capacity levels. When the maximum demand is C, the newsvendor can infer whether or not more capacity is required from how often a demand of C occurs. Similarly, they can decide if they have too much capacity if, for example, the demand is always less than the capacity.

Since the number of trials is fixed, the distribution of $\boldsymbol{X}$ is uniquely parameterised by $\boldsymbol{p}^0 = (p^0_{s,a})_{(s,a)\in\mathcal{S}\times\mathcal{A}}$. In the notation of Section 4.3.5, this means that $o = 1$. Suppose that we take a sample $\boldsymbol{x}_{s,a} = (x^1_{s,a}, \ldots, x^N_{s,a})$ from the distribution of $X_{s,a}$ for each $(s,a) \in \mathcal{S}\times\mathcal{A}$. Then, the MLE $\hat{p}_{s,a}$ of $p^0_{s,a}$ is given by:

$$\hat{p}_{s,a} = \frac{\sum_{j=1}^N x^j_{s,a}}{NC} \ \forall\ (s,a) \in \mathcal{S}\times\mathcal{A}.$$

In addition, the Fisher information (4.3.12) is given by:

$$I_{\mathbb{E}}(\hat{p}_{s,a}) = \frac{NC}{\hat{p}_{s,a}(1-\hat{p}_{s,a})}.$$

Therefore, our approximate $100(1-\alpha)\%$ confidence set for $\boldsymbol{p}^0_s$ is given by:

$$\Theta^\alpha_s = \left\{ \boldsymbol{p}_s \in [0,1]^A : \sum_{a\in\mathcal{A}} \frac{NC(p_{s,a} - \hat{p}_{s,a})^2}{\hat{p}_{s,a}(1-\hat{p}_{s,a})} \leq \chi^2_{A,1-\alpha} \right\} \tag{4.4.2}$$

As discussed in Section 4.3.5, in order for the parametric robust Bellman update (4.3.10) to be solvable so that we can generate policies, we consider discrete ambiguity sets. Since (4.4.2) is a multivariate set, it is difficult to discretise directly. Therefore, we will construct a set Θ^{base}_s such that $\Theta^\alpha_s \subseteq \Theta^{\text{base}}_s$ and discretise Θ^{base}_s instead. Then, we construct a discretisation of Θ^α_s by extracting all elements of Θ^{base}_s that also lie in Θ^α_s. Observe that the definition of Θ^α_s implies that every $\boldsymbol{p}_s \in \Theta^\alpha_s$ satisfies:

$$p_{s,a} \in p^{\text{I}}_{s,a} = \left[\max\left\{0, \hat{p}_{s,a} - \sqrt{\chi^2_{A,1-\alpha} I_{\mathbb{E}}(\hat{p}_{s,a})}\right\}, \min\left\{1, \hat{p}_{s,a} + \sqrt{\chi^2_{A,1-\alpha} I_{\mathbb{E}}(\hat{p}_{s,a})}\right\}\right]$$

for all $a \in \mathcal{A}$. Therefore, defining:

$$\Theta^{\text{base}}_s = p^{\text{I}}_{s,1} \times \ldots \times p^{\text{I}}_{s,A},$$

we have $\Theta^\alpha_s \subseteq \Theta^{\text{base}}_s$. Furthermore, define $p^l_{s,a}$ and $p^u_{s,a}$ as the lower and upper bounds of $p^{\text{I}}_{s,a}$ for each $(s,a) \in \mathcal{S}\times\mathcal{A}$. We can then find discretisations of each $p^{\text{I}}_{s,a}$ containing

M points as follows:

$$\tilde{p}^{\mathrm{I}}_{s,a} = \left\{ p^l_{s,a} + m\frac{p^u_{s,a} - p^l_{s,a}}{M-1} \right\}.$$

Then, a discretisation of Θ^{base}_s is given by $(\Theta^{\text{base}}_s)' = \tilde{p}^{\mathrm{I}}_{s,1} \times \ldots \times \tilde{p}^{\mathrm{I}}_{s,A}$. Finally, a discretisation of Θ^{α}_s is given by $(\Theta^{\alpha}_s)' = (\Theta^{\text{base}}_s)' \cap \Theta^{\alpha}_s$.

Experimental Design

We now detail the experiments used to test our algorithms on the dynamic newsvendor problem. The parameters used were as follows. We considered $w, h, b', c \in \{1, 5, 10\}$ such that $w > c$. The capacities we considered we $C \in \{1, 2, 3, 7, 9, 14\}$. This leads to $|\mathcal{S}| = |\mathcal{A}| \in \{2, 3, 4, 8, 10, 15\}$. We used a discount parameter of $\gamma = 0.5$ in all cases. For each algorithm, the value iteration algorithm was run for a maximum of $n^{\max} = 1000$ iterations. With regard to ambiguity sets, we always used $\alpha = 0.05$, the discretisation parameter was $M \in \{3, 5, 10\}$ and we took $N \in \{10, 50\}$ samples to create the MLEs. Each algorithm was given a maximum time of 4 hours to complete value iteration and find the optimal policy after value iteration ended.

In addition, the parametric algorithms were given a maximum of 4 hours to complete their precomputation, i.e. computing the discrete ambiguity set and corresponding transition probabilities. Note that this is not required for solving value iteration with PBS, but it is required to compute the optimal policy after value iteration ends. If an algorithm ran for 4 hours and the model was not solved, then the algorithm is said to have *timed out* for this instance. Both the parametric and non-parametric models used $100(1-\alpha)\%$ confidence sets as ambiguity sets. The parametric model

used (4.4.2) or a discretisation thereof, and the non-parametric model used (4.3.3) where κ is defined by (2.3.11). In addition, we used a value iteration tolerance of $\varepsilon = 10^{-6}$ and we initialised the value functions as $\boldsymbol{v}^0 = \mathbf{0}$. In PBS, we used a gap of $\tilde{\epsilon} = 0.01$ with $(\theta^{\min}_{s,a}, \theta^{\max}_{s,a}) = (0, 1)$ for all $(s, a) \in \mathcal{S} \times \mathcal{A}$. Finally, the bisection search tolerance used for PBS and NBS was $\epsilon = 10^{-7}$.

The above inputs generated 810 instances. We ran value iteration on each instance using 5 different algorithms, where each one is defined by how it solves each robust Bellman update. The algorithms and how they solve the update are as follows:

1. PBS: solve the parametric update (4.3.10) directly using the parametric projection-based bisection search algorithm of Section 4.3.6.

2. CS: the cutting surface algorithm of Section 4.3.6.

3. LP: use Gurobi to solve the approximate LP reformulation (4.3.11) of the parametric update (4.3.10).

4. QP: use Gurobi to solve the CQP reformulation (4.3.6) of the non-parametric update (4.3.2).

5. NBS: solve the non-parametric update (4.3.2) using the non-parametric projection-based bisection search algorithm of Section 4.3.4.

Value iteration was run until either $n^{\max}$ iterations had been completed, 4 hours of run time had been used, or the algorithm converged. After value iteration ended, for LP, CS and QP the policy was returned. For PBS, the policy was extracted using CS. For NBS, the policy was extracted using QP.

Times Taken

In this section, we summarise the times taken by the algorithms. We first present the number of times that each algorithm timed out. Firstly, LP and CS timed out while running value iteration 56 and 54 times respectively. No other algorithm timed out while running value iteration. Secondly, although PBS never timed out while running value iteration, CS timed out twice while computing the optimal policy for PBS's value functions. As we will show, PBS is a fast algorithm in itself, and these timeouts are a result of the slowness of CS in instances with large ambiguity sets.

Due to the above result, we present the times taken to run value iteration separately from the times taken to compute the policy. Table 4.4.1 summarises the amount of time that each algorithm spent running value iteration. This table shows that PBS took 31 seconds on average to finish value iteration, while CS took 17 minutes 30 seconds and LP took 26 minutes. It is therefore clear that PBS results in greatly reduced times to complete value iteration compared to these solver-based algorithms.

CS also saves approximately 12 minutes per iteration compared with LP on average. Note that LP and CS's average times per iteration are large because, when they timed out, they usually timed out after only one iteration. In addition, NBS took 43 seconds on average to complete value iteration, which is 33% slower than PBS. On average, NBS is much faster than its solver-based equivalent QP, which took over 6 minutes on average to finish value iteration. However, it is important to note that QP led to convergence issues in our experiments. While all other algorithms always finished value iteration in around 31 iterations, when using QP, value iteration

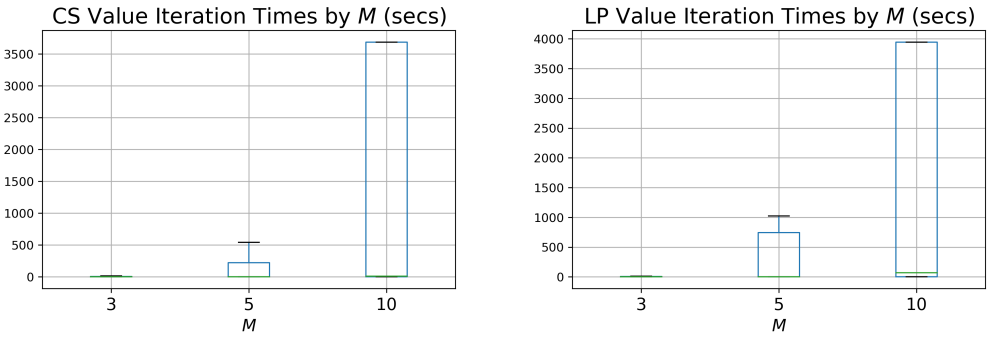
failed to converge in 378 instances. This was likely due to Gurobi being unable to provide precise enough optimal objective values. In addition, QP was also the fastest algorithm per iteration, and was only slow overall due to value iteration's failure to converge when using this algorithm.

Algorithm	Mean Time	Max Time	Mean Time Per Iteration
PBS	0:00:31.09	0:04:29.37	0:00:01.07
CS	0:17:30.90	4:00:00	0:04:25.77
LP	0:26:00.73	4:00:00	0:16:20.66
QP	0:06:16.24	0:55:59.19	0:00:00.41
NBS	0:00:43.52	0:09:21.02	0:00:01.48

Table 4.4.1: Summary of times taken to run value iteration (binomial)

It is clear from this table that CS and LP can both become very slow. The main reason for this is M, the parameter defining the fineness of the discretisation of Θ_s^a used by the parametric solver-based algorithms. We confirm this with Figure 4.4.1, which shows boxplots of CS and LP's value iteration run times by M. Figures 4.4.1a and 4.4.1b show that both CS and LP scale poorly with M in terms of value iteration run times. However, the effect of M is not particularly noticeable until $M = 10$. CS scales better than LP, but it still becomes slow for instances with large M or large C. Please note that, unlike the CS algorithms of Chapters 2 and 3, this CS algorithm is the optimal version which finds the worst-case parameter over the *entire* ambiguity set in every iteration. This explains why it does not offer the same level of time

savings when compared with LP as the CS algorithms of Chapters 2 and 3.

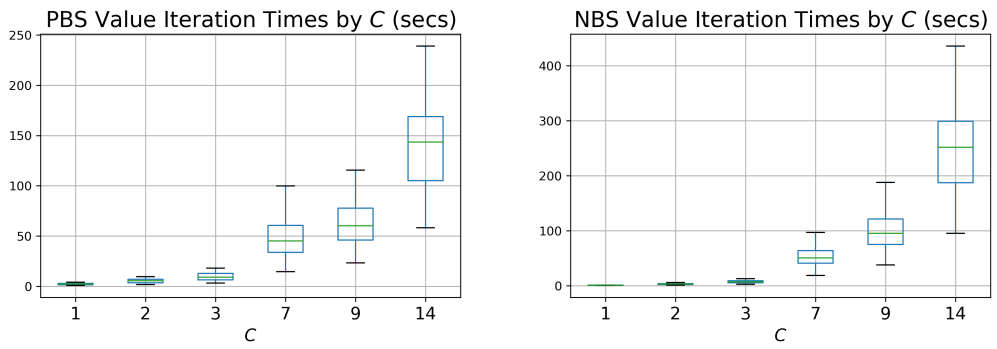


(a) (b)

Figure 4.4.1: Boxplots of value iteration run times of (a) CS and (b) LP, by M.

Since PBS and NBS do not rely on a discrete ambiguity set, their value iteration times are not affected by M. Therefore, the main parameter affecting their value iteration times is C. We present boxplots of PBS and NBS's value iteration times by C in Figure 4.4.2. Figures 4.4.2b and 4.4.2b show similar increases in times as C increases, but it is clear that PBS generally scaled better with C than NBS. For $C = 14$, PBS typically took no longer than 250 seconds to complete value iteration, while NBS typically took no longer than 450 seconds. On the other hand, NBS was slightly faster than PBS for small C. The reason for the difference in scaling is likely because NBS solves $S+1 = C+2$ sub-problems in order to solve a projection problem, whereas PBS always carries out a 3-step procedure to solve its projection problems. Hence, the number of steps involved in solving a projection problem increases with C for NBS, but not for PBS.

Although PBS was faster than NBS in running value iteration, it generally took longer for the parametric optimal policy to be computed in the parametric case than

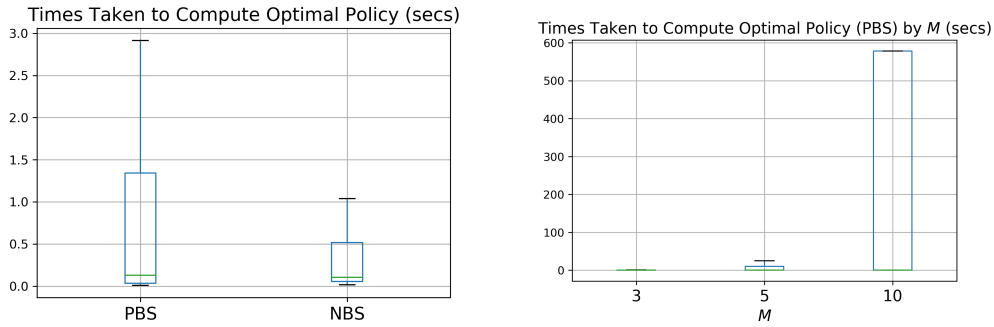


(a) (b)

Figure 4.4.2: Boxplots of value iteration run times of (a) PBS and (b) NBS, by C (binomial).

the non-parametric. This is because computing the policy is not part of PBS, and so CS had to be used for this. On average, it took 8 minutes 27 seconds for CS to compute the policy for PBS's value function, and only 0.43 for QP to compute NBS's. However, the parametric average is greatly affected by a small selection of very slow instances. Figure 4.4.3a shows a boxplot of the times taken to compute the policy for PBS and NBS's value functions after value iteration ended. Note that this boxplot does not show outliers, which are defined as any data that are further than 1.5 times the interquartile range above the 75th percentile or below the 25th percentile. Figure 4.4.3 shows that while the average time to compute the policy for PBS's values was 8 minutes 27 seconds, the median time was only 0.128 seconds. The 25th and 75th percentiles of the time taken to compute the parametric policy are 0.0325 and 1.34 seconds.

Figure 4.4.3b explains why the average time to compute the policy for PBS was large. In particular, it shows that this starts to take a very long time when $M = 10$. This



(a) (b)

Figure 4.4.3: Boxplots of times taken to compute optimal policy for (a) both BS algorithms and (b) PBS by M (binomial).

parameter defines the fineness of the discretisation of Θ_s^α used by CS when computing the policy. Since PBS uses Θ_s^α directly in value iteration, this parameter does not affect its value iteration run times. Hence, the slow times to compute a policy for PBS are a reflection on CS's scaling with respect to M, not PBS's. This can be confirmed by comparing Figure 4.4.3b with Figure 4.4.1a, and observing that the exact same pattern is present in both. Due to this, it is a downside that PBS does not provide an optimal policy. The same applies to NBS, but since QP is faster than CS in generating a policy, the effect is not so severe. Comparing the times taken to compute optimal policies is therefore not comparing the run times of PBS and NBS, but comparing the run times of CS and QP.

Comparison of Value Functions, Distributions and Policies

In this section, we compare the values, policies and worst-case transition distributions from the 5 algorithms tested. We first discuss the effect of the discretisation of Θ_s^α on the value functions from LP and CS. Following this, we compare the outputs from

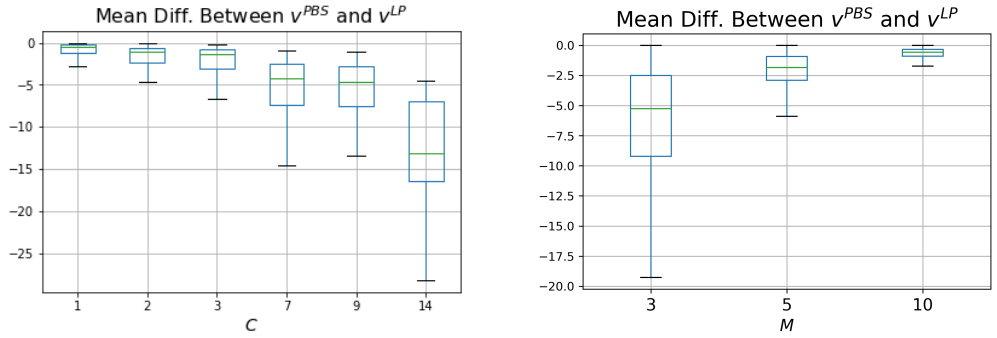
PBS and NBS in order to assess the benefits of incorporating additional distributional information into the model.

Let $\boldsymbol{v}^y$ be the value function generated by running value iteration with algorithm $y \in \mathcal{Y} = \{\text{PBS}, \text{CS}, \text{LP}, \text{QP}, \text{NBS}\}$. Similarly, define $\boldsymbol{\pi}^y$ as the policy and $\boldsymbol{P}^y$ as the worst-case transition distribution from using algorithm y in the value iteration algorithm. Then, we can summarise the differences between LP's approximate value functions and PBS's optimal value functions via Figure 4.4.4. Figures 4.4.4a and 4.4.4b show boxplots of the mean difference between $\boldsymbol{v}^{\text{LP}}$ and $\boldsymbol{v}^{\text{PBS}}$ over all instances where LP did not time out, by C and M, respectively. The mean differences are calculated as:

$$\frac{1}{|S|} \sum_{s \in \mathcal{S}} \left(v_s^{\text{PBS}} - v_s^{\text{LP}} \right). \tag{4.4.3}$$

We see from Figure 4.4.4a that the average difference between $\boldsymbol{v}^{\text{PBS}}$ and $\boldsymbol{v}^{\text{LP}}$ was always negative, with the magnitude of the difference growing larger as C increases. This means that LP's value function estimates are higher than PBS's optimal values. This is intuitive, since LP uses a discrete subset of Θ_s^α and therefore cannot find the true worst-case parameter for any given $\boldsymbol{\pi}$, in general. Therefore, LP's value function estimates overestimate the worst-case reward for a given policy. As is reflected in Figure 4.4.4b, the two value functions get closer as M increases, with average differences that are less than 2.5 in absolute value for $M = 10$. These plots suggest two results. Firstly, the effect of the discretisation increases as C increases. In other words, for larger C, the value function estimates resulting from the discrete approximations are less accurate. Secondly, the value function estimates resulting from the discretised ambiguity set appear to converge to their optimal values over the complete

(not discretised) ambiguity set.



(a) (b)

Figure 4.4.4: Boxplots of mean difference between $\boldsymbol{v}^{\text{LP}}$ and $\boldsymbol{v}^{\text{PBS}}$ by (a) C and (b) M (binomial).

We also compare the values from NBS with those from PBS in Figure 4.4.5. The quantities plotted here are the mean differences between $\boldsymbol{v}^{\text{PBS}}$ and $\boldsymbol{v}^{\text{NBS}}$:

$$\frac{1}{|S|}\sum_{s\in\mathcal{S}}\left(v_s^{\text{PBS}}-v_s^{\text{NBS}}\right).$$

Figure 4.4.5 shows that the value functions were generally quite close. The smallest and largest mean difference between $\boldsymbol{v}^{\text{PBS}}$ and $\boldsymbol{v}^{\text{NBS}}$ were -2.24 and 1.56 respectively. However, there is a clear pattern in the value function differences as C increases. For small C, Figure 4.4.5 shows that $\boldsymbol{v}^{\text{NBS}}$ and $\boldsymbol{v}^{\text{PBS}}$ were very close, with $\boldsymbol{v}^{\text{PBS}}$'s mean value (taken over s) being slightly higher. However, as C increases past 3 we see a clear pattern of $\boldsymbol{v}^{\text{NBS}}$'s mean value becoming larger than $\boldsymbol{v}^{\text{PBS}}$'s. The magnitude of this difference grows as C increases. This indicates that the worst-case distributions resulting from the parametric ambiguity set can be *worse* than those from the non-parametric set, i.e. they can lead to lower worst-case rewards.

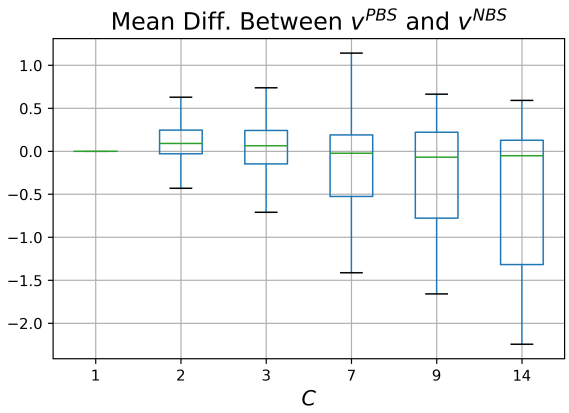
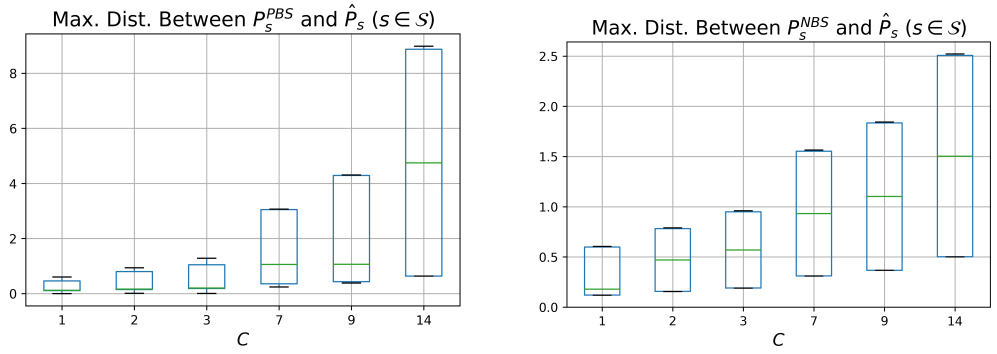


Figure 4.4.5: Boxplot of mean difference between $\boldsymbol{v}^{\text{NBS}}$ and $\boldsymbol{v}^{\text{PBS}}$ by C (binomial).

This result can be explained by differences between the parametric and non-parametric ambiguity sets. Recall that the non-parametric ambiguity set $\mathcal{P}_s$ is defined as containing all distributions $\boldsymbol{P}_s$ that satisfy the inequality $\sum_{a\in\mathcal{A}} d_\phi\left(\boldsymbol{P}_{s,a}, \hat{\boldsymbol{P}}_{s,a}\right) \leq \kappa$. In contrast, the parametric ambiguity set Θ_s^α is defined using an inequality restricting the distance from $\hat{\boldsymbol{\theta}}_s$ that $\boldsymbol{\theta}_s$ can take. This does not restrict the distance from $\hat{\boldsymbol{P}}_s$ that the parametric worst-case can take in the same way as the non-parametric ambiguity set does. To understand this, we evaluate $\max_{s\in\mathcal{S}} \sum_{a\in\mathcal{A}} d_\phi\left(\boldsymbol{P}^y_{s,a}, \hat{\boldsymbol{P}}_{s,a}\right)$ for $y \in \{\text{PBS}, \text{NBS}\}$, for every instance solved.

We provide boxplots of these values in Figure 4.4.6. Figure 4.4.6 shows that, for every value of C, the parametric worst-case distribution was allowed to be further from $\hat{\boldsymbol{P}}_s$ than the non-parametric worst-case. As C increases, difference between the maximum distances for the parametric and non-parametric worst-case distributions increases, explaining why the value functions are more different for large C. This happens since larger C occurs when A is larger, meaning the LHS of the inequality defining $(\Theta_s^\alpha)'$ is a sum of more terms, and since $\chi^2_{oA,1-\alpha}$ is increasing in A.



(a) (b)

Figure 4.4.6: Boxplots of $\max_{s\in\mathcal{S}} \sum_{a\in\mathcal{A}} d_\phi \left(\boldsymbol{P}^y_{s,a}, \hat{\boldsymbol{P}}_{s,a} \right)$ for (a) y = PBS and (b) y = NBS (binomial).

In general, this means that a distribution being binomial may lead to its inclusion in the parametric confidence set even though it is further from $\hat{\boldsymbol{P}}$ than any distribution in the non-parametric confidence set. Similarly, distributions that are not binomial need to be much closer to $\hat{\boldsymbol{P}}$ in order to be considered as candidates for the true distribution. The result in Figure 4.4.6 suggests that, for this problem, the parametric ambiguity sets are more risk-averse. It is worth noting, however, that NBS's values being slightly higher on average does not necessarily mean that more long-run reward would be obtained under the non-parametric model. This depends on the initial distribution $\boldsymbol{Q}$. For example, studying the value functions we see that $v_0^{\text{PBS}} \geq v_0^{\text{NBS}}$ was true in 66% of instances, $v_1^{\text{PBS}} \geq v_1^{\text{NBS}}$ was true in 65% of instances, and $v_2^{\text{PBS}} \geq v_2^{\text{NBS}}$ was true in 50.4% of the instances with $C \geq 2$. If the initial distribution satisfied, for example, $Q_s > 0$ only for $s \leq 2$, then the non-parametric approach would typically not achieve more long-term expected reward.

We now compare the policies from PBS and NBS. The first characteristic we study

is determinism. In these experiments, we find that PBS's optimal policy was deterministic in 42% of instances. NBS's policy was deterministic in 16% of instances. This indicates that the parametric model is more likely to yield deterministic policies than the non-parametric model. In addition, we studied the expected actions for each state in order to determine how conservative each model is. Specifically, we calculated $\sum_{a\in\mathcal{A}} a\pi^y_{s,a}$ for each s, for $y \in \{\text{PBS}, \text{NBS}\}$ for every instance that we solved. Generally speaking, the expected actions under PBS and NBS were similar.

However, we found two main results. Firstly, when the inventory level s is below 13, PBS will typically purchase slightly more stock. For the very smallest states, PBS only ordered between 5% and 6% more stock than NBS. However, for $s = 10$ and $s = 11$ PBS ordered 31% and 24% more respectively. Secondly, when the stock level is high, i.e. 13 or 14, NBS will typically purchase more stock. This was most noticeable for $s = 14$, where NBS ordered 38% more than PBS on average. From this, we can conclude that PBS's policies are typically less conservative for the majority of states, but that NBSs's are slightly less conservative for the largest 2 states.

It may seem odd that either algorithm makes positive orders when $s = C = 14$ since any stock above C is lost. This occurs due to differences in the worst-case distributions for different actions. For example, when in state C it may be the case that ordering a non-zero amount of stock leads to a much higher worst-case probability of then transitioning to state zero (and hence selling all stock). For example, this may happen when $\hat{p}_{C,a}$ is much larger for $a > 0$ than for $a = 0$, which can occur simply due to sampling variation. In some cases, spending some additional multiple of w

to purchase stock that would then be wasted actually results in a higher expected reward due to the fact that the newsvendor is then much more likely to sell all of their stock. Since NBS ordered more in the higher states, clearly this was of more benefit under NBS's worst-case distributions than PBS's. If this is not something that the newsvendor would like to allow, the policy can always be constrained to enforce that $\pi_{s,a} = 0$ for all $a \in \mathcal{A}$ such that $s + a > C$.

4.4.3 Numerical Experiments with Poisson Demands

We now carry out the same experiments as in Section 4.4.2, but where $X_{s,a} \sim \text{Pois}(\lambda^0_{s,a})$ for $(s,a) \in \mathcal{S} \times \mathcal{A}$. Firstly, we formulate the Poisson ambiguity sets. Following this, we describe the results of our experiments.

Ambiguity Sets

Suppose that $X_{s,a} \sim \text{Pois}(\lambda^0_{s,a})$ for $(s,a) \in \mathcal{S} \times \mathcal{A}$, and therefore:

$$f_{X_{s,a}}(x|\lambda^0_{s,a}) = \frac{(\lambda^0_{s,a})^x \exp(-\lambda^0_{s,a})}{x!} \quad (x \in \mathbb{N}_0).$$

The distribution of $\boldsymbol{X}$ is uniquely parameterised by $\boldsymbol{\lambda}^0 = (\lambda^0_{s,a})_{(s,a)\in\mathcal{S}\times\mathcal{A}}$. Similarly to in Section 4.4.2, we have $o = 1$ and suppose that we take the sample $\boldsymbol{x}_{s,a} = (x^1_{s,a}, \ldots, x^N_{s,a})$ from $X_{s,a}$ for each $(s,a) \in \mathcal{S} \times \mathcal{A}$. Then, the MLE $\hat{\lambda}_{s,a}$ of $\lambda^0_{s,a}$ is given by:

$$\hat{\lambda}_{s,a} = \frac{\sum_{j=1}^N x^j_{s,a}}{N} \ \forall \ (s,a) \in \mathcal{S} \times \mathcal{A}.$$

The Fisher information (4.3.12) is now given by:

$$I_{\mathbb{E}}(\hat{\lambda}_{s,a}) = \frac{N}{\hat{\lambda}_{s,a}}.$$

Hence, an approximate $100(1-\alpha)\%$ confidence set for $\boldsymbol{\lambda}_s^0$ (for large N) is given by:

$$\Theta_s^\alpha = \left\{ \boldsymbol{\lambda}_s \in \mathbb{R}_+^A : \sum_{a \in \mathcal{A}} \frac{N(\lambda_{s,a} - \hat{\lambda}_{s,a})^2}{\hat{\lambda}_{s,a}} \leq \chi^2_{A,1-\alpha} \right\}. \tag{4.4.4}$$

As in Section 4.4.2, we will construct a discretisation of Θ_s^α by creating a set Θ_s^{base} with $\Theta_s^\alpha \subseteq \Theta_s^{\text{base}}$ and discretising this set. Then, we extract elements of this discrete set that also lie in Θ_s^α. The definition of Θ_s^α implies that every $\boldsymbol{\lambda}_s \in \Theta_s^\alpha$ satisfies:

$$\lambda_{s,a} \in \lambda_{s,a}^{\text{I}} = \left[\max \left\{ 0, \hat{\lambda}_{s,a} - \sqrt{\frac{\chi^2_{A,1-\alpha} \hat{\lambda}_{s,a}}{N}} \right\}, \hat{\lambda}_{s,a} + \sqrt{\frac{\chi^2_{A,1-\alpha} \hat{\lambda}_{s,a}}{N}} \right]$$

for all $a \in \mathcal{A}$. Hence, we define $\Theta_s^{\text{base}} = \lambda_{s,1}^{\text{I}} \times \ldots \times \lambda_{s,A}^{\text{I}}$ and we have $\Theta_s^\alpha \subseteq \Theta_s^{\text{base}}$. Furthermore, define $\lambda_{s,a}^l$ and $\lambda_{s,a}^u$ as the lower and upper bounds of $\lambda_{s,a}^{\text{I}}$ for each $(s,a) \in \mathcal{S} \times \mathcal{A}$. We calculate the following discretisations of each $\lambda_{s,a}^{\text{I}}$, containing M points, as follows:

$$\tilde{\lambda}_{s,a}^{\text{I}} = \left\{ \lambda_{s,a}^l + m \frac{\lambda_{s,a}^u - \lambda_{s,a}^l}{M-1} \right\}.$$

Then, a discretisation of Θ_s^{base} is given by $(\Theta_s^{\text{base}})' = \tilde{\lambda}_{s,1}^{\text{I}} \times \ldots \times \tilde{\lambda}_{s,A}^{\text{I}}$. Finally, a discretisation of Θ_s^α is given by $(\Theta_s^\alpha)' = (\Theta_s^{\text{base}})' \cap \Theta_s^\alpha$.

Experimental Design

All of the main parameters for these experiments are the same as in Section 4.4.2. The only differences are with respect to the parameters used in the parametric bisection search algorithm of Section 4.3.6. We again use $\theta_{s,a}^{\min} = 0$ for all $(s,a) \in \mathcal{S} \times \mathcal{A}$, but since $\lambda_{s,a}$ is technically not bounded from above, there is no obvious value for $\theta_{s,a}^{\max}$. However, since any root of $\sum_{s' \in \mathcal{S}} P_{s,a,s'}^\theta b_{s'} = \beta$ that has $\lambda_{s,a} > \lambda_{s,a}^u$ cannot be an element of a $\boldsymbol{\lambda}$ that lies in the ambiguity set, we set $\theta_{s,a}^{\max} = \lambda_{s,a}^u$ for all $(s,a) \in \mathcal{S} \times \mathcal{A}$.

Since this creates a wider range for the potential roots than for the binomial case, we set $\tilde{\epsilon} = \frac{\theta_{s,a}^{\max} - \theta_{s,a}^{\min}}{100}$. This ensures that the same number of intervals are used here as in the binomial case, where we used $\tilde{\epsilon} = 0.01 \left(= \frac{1-0}{100}\right)$.

Times Taken

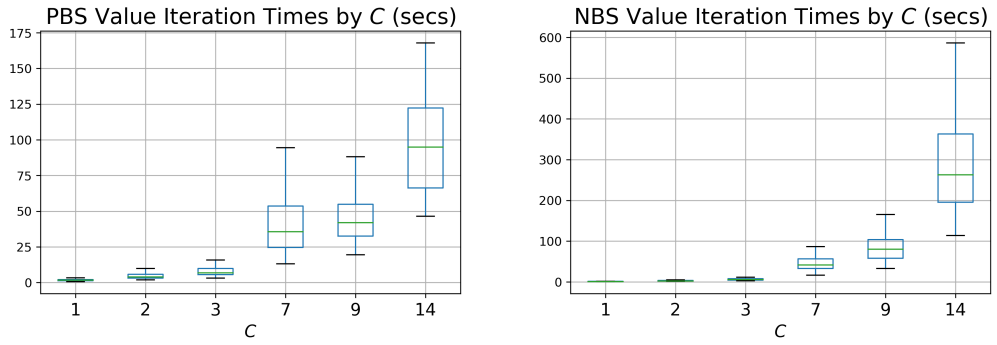
We now present the results of our experiments for Poisson demands. Of the 810 instances ran, we found that LP and CS timed out in 54. CS also timed out while finding PBS's optimal policy in 2 instances. PBS never timed out during value iteration. QP and NBS did not time out in any instance, but QP again resulted in convergence issues. When using QP to solve the Bellman updates, value iteration failed to converge in 384 instances. Table 4.4.2 summarises the times taken to run value iteration. Similar results to the binomial case can be found here, with PBS being faster than NBS, CS being slightly faster than LP, and QP being fast per iteration.

Algorithm	Mean Time	Max Time	Mean Time Per Iteration
PBS	0:00:24.44	0:03:59.44	0:00:00.85
CS	0:17:26.57	4:00:00	0:04:20.96
LP	0:23:45.70	4:00:00	0:16:16.30
QP	0:05:37.69	0:58:13.20	0:00:00.38
NBS	0:00:42.01	0:10:52.69	0:00:01.45

Table 4.4.2: Summary of times taken to run value iteration (Poisson)

Figure 4.4.7 compares the value iteration run times of PBS and NBS more closely. It

shows that, while NBS was slightly faster for small C, PBS scales much better with large C. For $C = 14$, PBS typically took no more than 3 minutes to finish value iteration. However, NBS took up to 10 minutes.

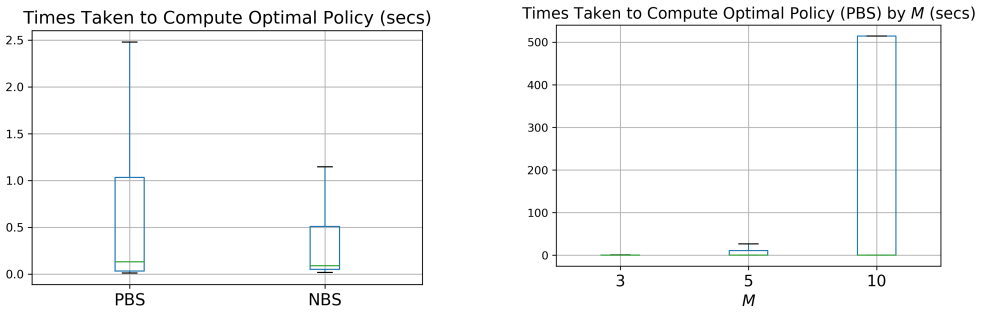


(a) (b)

Figure 4.4.7: Boxplots of value iteration run times of (a) PBS and (b) NBS, by C (Poisson).

While PBS is fast at finding the optimal values, it also took much longer to find a policy after running value iteration with PBS than with NBS. However, as before, this is due to CS being slow for large instances, and has nothing to do with PBS itself. On average, it took 8 minutes and 13 seconds for CS to find a policy for PBS's values, as opposed to 0.38 seconds for QP to compute the policy for NBS. However, the parametric times were skewed by large instances; the median time to compute the policy for PBS was 0.14 seconds.

Figure 4.4.8 shows boxplots of the times taken to find the policy for PBS's and NBS's values. While Figure 4.4.8a suggests that the speeds were similar for PBS and NBS's values for the majority of instances, Figure 4.4.8b show the drastic times taken under the parametric model for $M = 10$. As before, this is due to CS's slowness, not PBS's.



(a) (b)

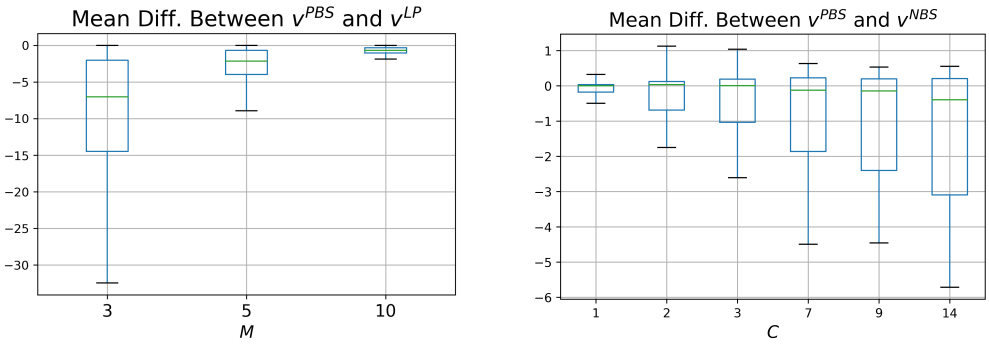
Figure 4.4.8: Boxplots of times taken to compute optimal policy for (a) both BS algorithms and (b) PBS by M (Poisson).

It is likely that if PBS were to be used in practice, a heuristic algorithm could be used in CS's place that would drastically speed up these times.

Comparison of Value Functions, Distributions and Policies

We now compare the outputs from the parametric and non-parametric models. Firstly, we compare the value functions from PBS with those from LP and NBS. Boxplots comparing these values are shown in Figure 4.4.9. As a reminder, these plots show $\frac{1}{|S|}\sum_{s\in\mathcal{S}}\left(v_s^{\text{PBS}} - v_s^y\right)$ for $y \in \{\text{LP}, \text{NBS}\}$. Figure 4.4.9a shows the convergence of LP's values to PBS's as M increases.

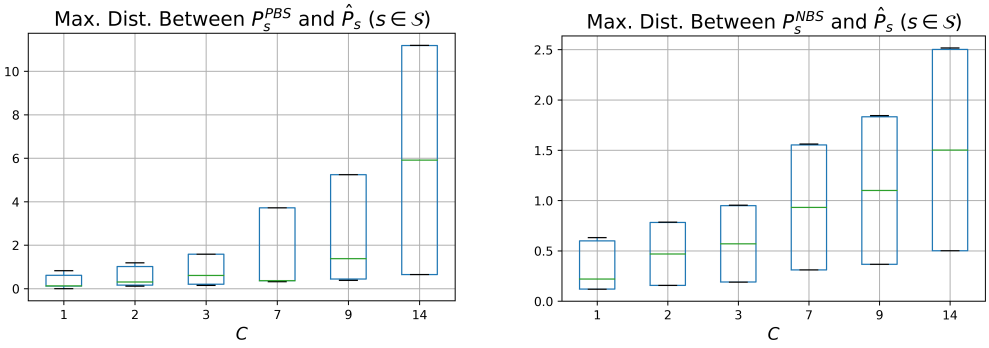
As for the binomial model, we see that LP's values are always higher, and they grow closer to PBS's on average as M increases. In addition, Figure 4.4.9b shows that PBS and NBS's values are similar for small C, but NBS's values are typically higher than PBS's for large C. In order to confirm that the same property of the ambiguity sets is responsible for this as for the binomial model, we plot the maximum distances from $\hat{\boldsymbol{P}}_s$ attained by $\boldsymbol{P}_s^{\text{PBS}}$ and $\boldsymbol{P}_s^{\text{NBS}}$ in Figure 4.4.10. This plot again suggests that the



(a) (b)

Figure 4.4.9: Boxplots of mean difference between $\boldsymbol{v}^{\text{PBS}}$ and (a) $\boldsymbol{v}^{\text{LP}}$ by M and (b) $\boldsymbol{v}^{\text{NBS}}$ by C (Poisson).

parametric worst-case distributions are much further from $\hat{\boldsymbol{P}}_s$.



(a) (b)

Figure 4.4.10: Boxplots of $\max_{s\in\mathcal{S}} \sum_{a\in\mathcal{A}} d_\phi\left(\boldsymbol{P}^y_{s,a}, \hat{\boldsymbol{P}}_{s,a}\right)$ for (a) y = PBS and (b) y = NBS (Poisson).

Finally, we compare the policies from the PBS and NBS. Similarly to the binomial model, we find that PBS's policies were more often deterministic. Specifically, 32% of PBS's policies were deterministic where only 14% of NBS's were. In addition, we also find that the parametric policies were slightly less conservative in terms of purchasing, for most states. However, for the largest 2 states, i.e. $s = 13, 14$, the non-parametric

policies were less conservative. NBS's polices ordered 30% more stock in these final two states than PBS's, whereas PBS's policies ordered between 5% and 25% more in the lower states.

4.5 Conclusions and Further Research

In this chapter, we studied robust Markov decision processes under parametric transition distributions. We focused on robust value iteration for s-rectangular ambiguity sets in particular. Based on a fast projection-based bisection search algorithm found in the literature for robust MDPs with ϕ-divergence ambiguity sets, we created a projection-based bisection algorithm for the parametric model in the case where the transition distribution is parametrised by one parameter. We also presented two other algorithms for solving the robust Bellman update, a linear programming algorithm and a cutting surface algorithm. These algorithms both discretise the ambiguity set for the true parameter in order to create a linear programming reformulation of the robust Bellman update. In addition, we showed how to use maximum likelihood estimation to create confidence sets for use as ambiguity sets in the parametric model.

In order to test our algorithms, we presented a dynamic multi-period newsvendor model and applied them to it. In particular, we carried out numerical experiments on the case where the demands in the newsvendor problem are binomial and Poisson. In both cases, we solved the non-parametric model in addition to the parametric model, in order to compare run times and solutions. We found 2 main results.

Firstly, our parametric bisection search algorithm was very fast at finishing value

iteration. In fact, it was faster than its non-parametric equivalent and offered significant time savings in comparison with the linear programming and cutting surface algorithms. This is due to the fact that the bisection algorithm does not rely on any discretisation of the ambiguity set, and hence does not need to carry out the large amount of pre-computation required by the solver-based algorithms. However, since our bisection algorithm does not return an optimal policy, one of the two solver-based algorithms had to be used to generate the policy after value iteration ended. This meant that using the parametric model sometimes resulted in large overall run times, since CS and LP scale poorly with the size of the ambiguity set.

Secondly, comparing the solutions from the parametric and non-parametric models, we found that the non-parametric value functions were typically higher than the parametric ones on average. This was due to the result that the parametric confidence sets allowed the corresponding worst-case distribution to be much further from the nominal distribution than was allowed by the non-parametric set. However, we also found that the parametric policies were less conservative than the non-parametric, for all states apart from the largest two.

There are two main directions for future research arising from this chapter. The most obvious direction is with regards to computing the optimal policy. Since both of our solver-based algorithms were very slow at this, it would be beneficial to find a faster way to extract the policy once our bisection algorithm finishes value iteration. Another potential area for further research is with regards to discretisation. As was the case in Chapters 2 and 3, discretisation of the ambiguity set is required to create

a linear programming model that approximates the true problem. This means that the transition matrices for every parameter in the ambiguity set must be computed prior to building the model, and also that the resulting model becomes very large for large ambiguity sets. Hence, we would like to study ways in which to circumvent the need for discretisation and reduce overall solution times.

Chapter 5

Conclusions and Further Research

In this chapter, we conclude the thesis. In Section 5.1, we summarise the contributions and findings of the thesis. Following this, in Section 5.2, we discuss potential areas for future research.

5.1 Contributions and Findings

This thesis has made a number of key contributions to the fields of optimisation under uncertainty and resource/inventory planning. The first major contribution is the parametric DRO framework introduced in Chapter 2. When the true distribution is parametric, our framework maintains this information by constructing ambiguity sets for the true distribution's parameters and only considering distributions lying in

the same parametric family as the true distribution. The PMF or PDF of this family is used directly in the objective function. The parametric DRO framework provides an intuitive way for parameter estimates and our uncertainty in these estimates to be included in optimisation models. For example, we showed how to construct confidence sets for the true parameters and use them as ambiguity sets in parametric DRO models. This hedges against the effects of inaccurate estimation and insufficient data, such as suboptimal solutions and poor estimates of the cost of a given solution.

The first application of the parametric DRO framework in this thesis was the resource planning problem of Chapter 2. For this problem, we developed the model under binomial intake random variables. This model became very large and slow to solve to optimality for large ambiguity sets. This led us to our second main contribution: a novel heuristic cutting surface algorithm for parametric DRO models. Cutting surface algorithms are known in the literature. However, our cutting surface algorithm uses theoretical properties of the objective function in order to construct a small set of extreme parameters in which to search for the worst-case parameter. We showed that our cutting surface algorithm offered significant improvements in solution time over solving the full DRO model, while losing a negligible amount of solution quality.

In Chapter 2, we also provided detailed comparisons of the parametric DRO solutions with the standard non-parametric solutions obtained from ϕ-divergence ambiguity sets. We found three main results. Firstly, the parametric model can be solved via our optimal cutting surface algorithm faster than the non-parametric model can be solved to optimality via Lagrangian reformulation. Secondly, the non-parametric

model was slightly less conservative than the parametric model; it pulled forward slightly more jobs on average. Finally, we found that the non-parametric model overestimated the worst-case costs associated with a given decision. This indicated that the non-parametric worst-case distributions were more extreme.

The model of Chapter 2 was a mixed integer linear program, and the distributions considered were finite and discrete. In Chapter 3, we extended the parametric DRO framework to problems with non-linear objective functions under either continuous or discrete random variables with infinite support. Specifically, we considered a multi-period budgeted newsvendor problem under normal and Poisson demands. In the case where the demand distribution was known, we developed a fast heuristic algorithm for the newsvendor problem. We tested the method that uses MLEs in place of the true parameters. While the solutions were quite near to optimal under the true distribution, this method resulted in poor estimates of the cost of a given decision. In some cases, it even predicted a profit for a solution that would result in a loss.

For the DRO models, we used piecewise linear approximations and constraints in our implementation in order to use standard solvers to generate solutions. Since the random variables for this problem had infinite support, it was no longer possible to calculate the entire distribution as we did for the binomial random variables in Chapter 2. Instead, we derived alternate expressions for the objective functions in terms of the PMFs/PDFs and CDFs of the parametric families. This allowed us to prove theoretical results about them as functions of the distribution's parameters. Using this information, we developed and tested two new versions of the heuristic

cutting surface algorithm of Chapter 2 and again showed that they provide very near-optimal solutions in a negligible amount of time.

The final contributions of this thesis were presented in Chapter 4, where we introduced the parametric RMDP framework. This extends the parametric DRO framework into the dynamic decision making space, where the MDP has an uncertain parametric transition matrix. We focused on s-rectangular ambiguity sets since then the RMDP can be solved in polynomial time via robust value iteration. For solving a robust Bellman update, we developed and tested a selection of algorithms.

The standout algorithm from Chapter 4 was our bisection search algorithm, PBS, which solves a robust Bellman update to ϵ-optimality via a bisection search using the solution of a set of parametric projection problems to update the current bounds on the value function. PBS has the unique characteristic that it does not rely on discretisation of the ambiguity set, unlike our other parametric algorithms. Testing our algorithms on a dynamic newsvendor model under binomial and Poisson demands, we found that PBS solved robust value iteration faster than its non-parametric equivalent and significantly faster than our other parametric algorithms.

5.2 Further Research

There are a number of areas for further research that would improve the applicability and practicality of our methods. This section provides details on these areas and how they might be implemented.

5.2.1 Solving Without Discrete Approximations

The main drawback of the PDRO approaches developed in this thesis is discretisation. Since the objective functions are often non-linear in the distribution's parameters, discretisation is employed to help reformulate the innner problem and create a model that is solvable as an LP or MILP. While it ultimately achieves this goal, it leads to many issues in terms of computation. Firstly, a large amount of pre-computation is often required in order to build the model prior to solving. Secondly, the resulting model can be very slow to solve due to the large number of constraints imposed by the discrete ambiguity set. Finally, using a discretised ambiguity set means we are only ever approximating the true PDRO model.

For these reasons, the most important direction for future research is to develop new ways to solve PDRO models without the need for discretisation. Since we have established in Chapter 2 that dualisation of the inner problem does not result in a solvable model, the most logical way to solve the PDRO model (1.3.1) without discretisation is to treat the entire model or the inner problem as a semi-infinite program (SIP). Model (1.3.1) can be written as:

$$\min_{\boldsymbol{x},\vartheta} \vartheta \tag{5.2.1}$$

$$\text{s.t. } \mathbb{E}_{\boldsymbol{\theta}}\left[h(\boldsymbol{x},\boldsymbol{Y})\right] - \vartheta \leq 0 \;\forall\; \boldsymbol{\theta} \in \Theta \tag{5.2.2}$$

$$\boldsymbol{x} \in \mathcal{X}. \tag{5.2.3}$$

Given this SIP formulation, there are a number of methods that may be applicable depending on the problem type. Hettich and Kortanek (1993) detail a number of

methodologies for solving SIP models. The first set of approaches is based around representing the feasible region with only finitely many constraints. The simplest way to do this is using a discrete approximation of the set, as we have done in this thesis. The second methodology suggested by Hettich and Kortanek (1993) considers a class of SIP problems where the feasible region in a neighbourhood of a feasible point can be equivalently represented by a finite set of constraints.

This method essentially reduces to solving the distribution separation problem (5.2.4) and then solving the model over a finite discrete set of constraints based on the solutions obtained.

$$\min_{\boldsymbol{\theta}\in\Theta} \mathbb{E}_{\boldsymbol{\theta}}[h(\boldsymbol{x}, \boldsymbol{Y})]. \tag{5.2.4}$$

Due to this, it is similar to the cutting surface algorithm of Mehrotra and Papp (2014), which was initially developed for SIP problems. In order to use the algorithms of Hettich and Kortanek (1993) or Mehrotra and Papp (2014) to solve (1.3.1), we would need to solve (5.2.4) directly instead of via discretisation as we have done in our approximate CS algorithms. In general, (5.2.4) will be non-linear and its objective function will be a high-order polynomial or contain logarithms or exponential terms. Therefore, it will generally not be solvable as a quadratic or conic program.

In order to select a methodology for solving (5.2.4), we would need to further determine the theoretical properties of $\mathbb{E}_{\boldsymbol{\theta}}[h(\boldsymbol{x}, \boldsymbol{Y})]$. The first property we should look for is convexity. If $\mathbb{E}_{\boldsymbol{\theta}}[h(\boldsymbol{x}, \boldsymbol{Y})]$ is convex in $\boldsymbol{\theta}$ then we may use algorithms designed for non-linear convex optimisation problems with inequality constraints, such as interior point methods (Potra and Wright, 2000). Since we have been able to establish

some univariate monotonicity and convexity results in this thesis, we believe that it is likely that the objective function is convex in $\boldsymbol{\theta}$. If it is not, however, we may need to develop decomposition algorithms (Grothey, 2001; Floudas et al., 1989) that decompose the problem into a sequence of NLP and MILP problems in a search for global optima.

Since the algorithms of Mehrotra and Papp (2014) and Hettich and Kortanek (1993) require solving (5.2.4), finding a strong and fast methodology to solve this problem would be our first step in solving (1.3.1) without discretisation. If the algorithm is slow, or (5.2.4) is too difficult to solve (e.g. due to nonconvexity), we would look to other SIP methods to solve the model in (5.2.1)-(5.2.3). For example, for convex models, López and Still (2007) provide a number of methods based on KKT conditions in addition to numerical methods based on reduction, similar to those of Hettich and Kortanek (1993). The field of non-convex SIPs is relatively new, but reviews such as that of Djelassi et al. (2021) provide descriptions of a number of methods that we may test for our problems, such as overestimation, relaxation and interval methods.

5.2.2 The Pre-computation Bottleneck

If we continue to solve our models using discretisation, we might consider improving our methodology by reducing the effect of discretisation on overall run times. Our general approach to parametric DRO and RMDP models in this thesis has been to first discretise the ambiguity set and then represent the inner objective function via a set of expected value constraints. This methodology leads to a large amount of

pre-computation. Discretising the ambiguity set is a process that consists of three steps. Firstly, we create a large superset of the ambiguity set. Secondly, we discretise this superset. Finally, we extract elements of the ambiguity set from the discretised superset. This final step is cumbersome, since it entails checking whether each element of the discretised superset satisfies the inequality defining the ambiguity set. In addition, in some cases the discretised superset can be so large that it cannot be held in memory. Once the discretised ambiguity set is obtained, for discrete distributions we then need to compute a large number of PMF values for each parameter in the ambiguity set in order to build the expected value constraints.

Since the models do not take a significant amount of time to solve with our algorithms but the pre-computation is always required (except from for PBS in robust value iteration), the pre-computation presents a significant bottleneck in terms of the total build and run time. The preferred solution to this problem would be to eliminate the need for discretisation, hence eliminating all pre-computation. However, for parametric DRO models we are not aware of any other way to create a solvable model. For RMDPs, while it is possible to run robust value iteration without carrying out the precomputation (as evidenced by PBS), it is still required in order to generate the optimal policy. Therefore, the most realistic solution to this bottlenecking problem is to reduce the time that it takes to carry out precomputation.

Since the pre-computation only consists of simple arithmetic operations, the most obvious first step in speeding this step up is to use a faster programming language. All experiments in this thesis were run exclusively in Python, but a language such as C++

may be more appropriate for carrying out pre-computation. In addition, improving the way in which we discretise the superset of our ambiguity set may help reduce the bottleneck. In this thesis, we discretised the superset by dividing intervals into sets containing a fixed number of equally spaced points. We found that the bottleneck was only severe for the finest discretisations. However, since the worst-case parameters are often those furthest from the MLEs, it may be the case that the fineness of the discretisation does not significantly impact decisions. If so, a smaller ambiguity set can be used without much effect on the model's output. Further research is required to determine whether or not this is the case.

In addition, the brute-force approach that we have used to construct the discrete ambiguity set from the discrete superset can likely be improved. Checking every parameter in the discrete superset necessarily leads to a large number of parameters that are not elements of the ambiguity set. In order to reduce the amount of parameters that need to be checked, we could employ a more sophisticated approach. For example, once we find a parameter that is not an element of the ambiguity set, we can construct the set of all parameters that are the same as this one apart from one element. Members of this set do not need to be checked if their corresponding element is further from its MLE than the one that we have just checked, since this means it cannot be a member of the ambiguity set. Not checking these parameters may save a large amount of computation time.

It may also be possible to speed up the computation of PMFs. For example, we might consider saving some common PMF values into a lookup table so that they do

not need to be computed during model building. This could either be done prior to running any code, or on-the-fly so that if this PMF value is required again then it does not have to be re-computed.

5.2.3 Dependent Demand Random Variables

In the multi-period models of Chapters 2 and 3, we assumed that the demand random variables for each period were independent of one another. In practice, however, this is not always the case. For example, in the context of demand, it is common for seasonality to exist, meaning that demands for the same day or same month in different weeks or years are often correlated with one another. When the joint distribution of demands can be assumed to be known, incorporating this type of dependence is quite straight forward.

Many methodologies exist for incorporating partial or complete correlation information into DRO models outside of the PDRO framework. Moment-based ambiguity sets provide a natural way to incorporate this information. While early moment-based models only specified marginal moments (Bertsimas et al., 2004), more recent models allow for an entire covariance matrix to be specifed in the ambiguity set (Natarajan et al., 2011). Natarajan et al. (2011) solve their model via semidefinite relaxations, and even provide details of how the method can be extended to account for scenarios where the mean vector and moment matrix are not known exactly. The semi-parametric models discussed in Section 1.3.1 also provide a way to specify correlation information in the model's ambiguity sets. Ahipasaoglu et al. (2019) further

develop this area by presenting semi-parametric models specifying partial correlation information that can be solved in polynomial time. In addition, recent papers have incorporated correlation information in Wasserstein ambiguity sets and used duality to create reformulations (Gao and Kleywegt, 2017b; Wang et al., 2018).

While the above methods incorporate correlation information, they still do not maintain the family in which the true distribution was assumed to lie. It is not yet clear how correlation information can be incorporated into our PDRO models. The main difficulty in extending in this direction is that the joint distribution of a set of parametric random variables generally cannot be computed from only the marginal parameters and covariance matrix. This makes it impossible to calculate the objective function in the same way as we have done in this thesis. As a starting point for this type of extension, it is likely that we would therefore consider multivariate normal demands, where computing the joint distribution from this information is possible.

Under multivariate normal demands, a number of possibilities exist. Firstly, we might consider assuming that the covariance matrix is fixed and known. We would then use the covariance matrix directly in computing the objective function via the PDF, and in the confidence set via the Fisher information matrix. It is worth noting that this complicates the discretisation of the confidence set, since the inequality defining it is no longer separable over the individual parameters. A new methodology for discretising the confidence set would therefore be required. When the covariance matrix is assumed unknown, the problem becomes more complex. One way to incorporate this into our model is to treat it as an unknown parameter in the same way as we

did for $\boldsymbol{\mu}$. We could then take MLEs of the covariance matrix and include it in the confidence set via the Fisher information matrix. This would again mean that a new discretisation method would be required, however.

In other cases such as Poisson or binomial marginals, it is not possible to compute the joint distribution based on only the marginals and covariance matrix. In such cases, we may wish to apply marginal distribution models with parametric marginals given by the MLE parameters, for example by applying the methods proposed by Chen et al. (2022) for linear DRO models. This would ensure that the worst-case distribution's marginals lie in the required family without assuming independence. However, it should only be used if the MLEs can be believed to be accurate. Another option is to attempt to combine PDRO with other methodologies in order to achieve the required result. Since PDRO generates worst-case marginals under the independence assumption, there may be scope to develop methods that use PDRO's marginals as inputs to marginal distribution models. This would be a heuristic procedure, and should be compared with the first method discussed in this paragraph.

5.2.4 Extensions to More Complex MDPs

The RMDPs that we considered in Chapter 4 were arguably the simplest kind: finite state and action space, infinite horizon MDPs with s-rectangular ambiguity sets. There are a number of extensions that could be made to these MDPs to ensure that they are more widely applicable. The first extension is to consider state-dependent action spaces. This is often much more realistic than assuming that the same set of

actions can be taken in each state. For example, in Chapter 4, we found that the policies for our RMDPs were ordering amounts of stock that would take them above the capacity of the newsvendor. In practice, however, this might not be allowed, since the newsvendor would lose all stock above the capacity. Therefore, the newsvendor would likely restrict their actions to those that do not take their stock above the capacity, given the current inventory level.

In addition, it would be interesting to extend our methods for RMDPs into finite horizon models. These are often more realistic in practice, and would provide dynamic alternatives to the problems studied in Chapters 2 and 3. For finite horizon problems, robust backward dynamic programming algorithms would be applied instead of robust value iteration. Each robust dynamic programming iteration would be similar to an iteration of robust value iteration, but solving the RMDP may require more iterations due to the enumerative nature of backward dynamic programming. It is therefore possible that more adjustments would need to be made to our methods to ensure that they scale well enough to be used within robust dynamic programming. Further extensions that we would like to make is to consider models with continuous state and action spaces, or consider other families of distributions in our robust models.

Appendices

Appendix A

Distributionally Robust Resource Planning Under Binomial Demand Intakes

A.1 Derivation of CQP Reformulation of Non-parametric Model

A.1.1 General Reformulation

For a ϕ-divergence ambiguity set with nominal distribution $\boldsymbol{Q}$, we can write the inner problem of the DRO model (2.3.1)-(2.3.1) as:

$$\max_{\boldsymbol{P}} \sum_{t=1}^{T} a_t \mathbb{E}_{\boldsymbol{P}}(R_t)$$

$$\text{s.t. } P_j \geq 0 \ \forall \ j = 1, \ldots, n$$

$$\sum_{j=1}^{n} P_j = 1$$

$$d_\phi(\boldsymbol{P}, \boldsymbol{Q}) \leq \kappa.$$

The Lagrangian of this model is given by:

$$L(\boldsymbol{P}, \eta, \nu) = \sum_{t=1}^{T} \sum_{j=1}^{n} P_j a_t R_t^{ij} + \eta \left(\kappa - d_\phi(\boldsymbol{P}, \boldsymbol{Q})\right) + \nu \left(1 - \sum_{j=1}^{n} P_j\right).$$

The objective function of the dual problem is therefore:

$$g(\eta, \nu) = \sup_{\boldsymbol{P} \geq 0} L(\boldsymbol{P}, \eta, \nu).$$

Since $\kappa > 0$ and $d_\phi(\boldsymbol{Q}, \boldsymbol{Q}) = 0 < \kappa$ where $\boldsymbol{Q}$ is a feasible choice of distribution, Slater's condition holds. Since the primal is concave, we have strong duality. We can hence write the objective of the dual of the inner problem as:

$$\min_{\eta \geq 0, \nu} g(\eta, \nu) = \min_{\eta \geq 0, \nu} \sup_{\boldsymbol{P} \geq 0} \left\{ \sum_{t=1}^{T} \sum_{j=1}^{n} P_j a_t R_t^{ij} + \eta \left(\kappa - d_\phi(\boldsymbol{P}, \boldsymbol{Q})\right) + \nu \left(1 - \sum_{j=1}^{n} P_j\right) \right\} \tag{A.1.1}$$

$$= \min_{\eta \geq 0, \nu} \left\{ \eta\kappa + \nu + \sup_{\boldsymbol{P} \geq 0} \left(\sum_{j=1}^{n} P_j \sum_{t=1}^{T} a_t R_t^{ij} - \eta d_\phi(\boldsymbol{P}, \boldsymbol{Q}) - \nu \sum_{j=1}^{n} P_j \right) \right\} \tag{A.1.2}$$

$$= \min_{\eta \geq 0, \nu} \left\{ \eta\kappa + \nu + \sup_{\boldsymbol{P} \geq 0} \left(\sum_{j=1}^{n} P_j \sum_{t=1}^{T} a_t R_t^{ij} - \eta \sum_{j=1}^{n} Q_j \phi\left(\frac{P_j}{Q_j}\right) - \nu \sum_{j=1}^{n} P_j \right) \right\} \tag{A.1.3}$$

$$= \min_{\eta \geq 0, \nu} \left\{ \eta\kappa + \nu + \sum_{j=1}^{n} \sup_{P_j \geq 0} \left(P_j \sum_{t=1}^{T} a_t R_t^{ij} - \eta Q_j \phi\left(\frac{P_j}{Q_j}\right) - \nu P_j \right) \right\} \tag{A.1.4}$$

$$= \min_{\eta \geq 0, \nu} \left\{ \eta\kappa + \nu + \sum_{j=1}^{n} \sup_{P_j \geq 0} \left(P_j \left(\sum_{t=1}^{T} a_t R_t^{ij} - \nu \right) - \eta Q_j \phi \left(\frac{P_j}{Q_j} \right) \right) \right\} \tag{A.1.5}$$

$$= \min_{\eta \geq 0, \nu} \left\{ \eta\kappa + \nu + \eta \sum_{j=1}^{n} Q_j \sup_{r_j \geq 0} \left(r_j \frac{R_t^{ij} - \nu}{\eta} - \phi\left(r_j\right) \right) \right\} \tag{A.1.6}$$

$$= \min_{\eta \geq 0, \nu} \left\{ \eta\kappa + \nu + \eta \sum_{j=1}^{n} Q_j \sup_{r_j \geq 0} \left(r_j s_j - \phi\left(r_j\right) \right) \right\} \tag{A.1.7}$$

$$= \min_{\eta \geq 0, \nu} \left\{ \eta\kappa + \nu + \eta \sum_{j=1}^{n} Q_j \phi^*(s_j) \right\}, \tag{A.1.8}$$

where $s_j = \frac{\sum_{t=1}^{T} a_t R_t^{ij} - \nu}{\eta}$ and $r_j = \frac{P_j}{Q_j}$. Note that we can replace the sum of maxima with a maximum of sums in (A.1.6) because the objective is separable over j. Finally, we require the dual feasibility constraint:

$$s_j \leq \left(\lim_{\tau \to \infty} \frac{\phi(\tau)}{\tau} \right) \ \forall\ j = 1, \ldots, n.$$

This ensures that ϕ^* does not grow to infinity. Consider $\phi^*(s_j) = \sup_{\tau \geq 0}\{s_j\tau - \phi(\tau)\}$. If $\frac{\phi(\tau)}{\tau} \to \infty$ as $\tau \to \infty$ then this constraint can be removed. If not, i.e. $\lim_{\tau \to \infty} \frac{\phi(\tau)}{\tau} = \bar{s} < \infty$, then for $s > \bar{s}$ we have $\phi^*(s) = \infty$. Note that, according to the definition given by Ben-Tal et al. (2013), we have $0\phi^*(s/0) := (0\phi^*)(s)$, which is zero if $s \leq 0$ and $+\infty$ if $s > 0$. Therefore, combining with the outer problem, we have:

$$\min_{\boldsymbol{y}, \boldsymbol{R}, \eta, \nu} \ \left\{ \eta\kappa + \nu + \eta \sum_{j=1}^{n} Q_j \phi^*(s_j) \right\},$$

$$\text{s.t.} \quad (2.3.1) - (2.3.8)$$

$$\eta \geq 0$$

$$\sum_{t=1}^{T} a_t R_t^{ij} - \nu \leq \eta \left(\lim_{\tau \to \infty} \frac{\phi(\tau)}{\tau} \right) \ \forall\ j = 1, \ldots, n.$$

A.1.2 Modified χ^2-divergence

Recall (2.3.12), which states that for a modified χ^2-divergence, we have:

$$d_{m\chi^2}(\boldsymbol{P}, \boldsymbol{Q}) = \sum_{j=1}^{n} \frac{(P_j - Q_j)^2}{Q_j}.$$

Reformulation

The conjugate of $\phi_{m\chi^2}$ is given by:

$$\begin{aligned}\phi^*_{m\chi^2}(s) &= \begin{cases} -1 & \text{if } s < -2 \\ \\ s + \frac{s^2}{4} & \text{if } s \geq -2 \end{cases} \\ &= \max\left\{\frac{s}{2} + 1, 0\right\}^2 - 1.\end{aligned}$$

Using ϕ^* to represent $\phi^*_{m\chi^2}$ for shorthand, we can expand $\phi^*(s_j)$ in order to write:

$$\begin{aligned}\eta\phi^*(s_j) &= \eta\left(\max\left\{\frac{s_j}{2} + 1, 0\right\}^2 - 1\right) \\ &= \eta \max\left\{\frac{\sum_{t=1}^{T} a_t R_t^{ij} - \nu}{2\eta} + 1, 0\right\}^2 - \eta \\ &= \frac{1}{4\eta}\max\left\{\sum_{t=1}^{T} a_t R_t^{ij} - \nu + 2\eta, 0\right\}^2 - \eta.\end{aligned}$$

In order to define $\phi^*(s_j)$ using convex quadratic constraints, we first need to remove the max operator from this expression. Hence, we define a dummy variable ζ_j to represent the value of $\max\left\{\sum_{t=1}^{T} a_t R_t^{ij} - \nu + 2\eta, 0\right\}$. We enforce $\boldsymbol{\zeta}$'s value via (A.1.9) and (A.1.10).

$$\zeta_j \geq \sum_{t=1}^{T} a_t R_t^{ij} - \nu + 2\eta \;\forall\; j = 1, \ldots, n \tag{A.1.9}$$

$$\zeta_j \geq 0 \;\forall\; j = 1, \ldots, n. \tag{A.1.10}$$

Then, we can define another dummy variable $u_j = \frac{\zeta_j^2}{\eta} = 4\eta\phi^*(s_j) + \eta$. We enforce the value of u_j using a conic quadratic constraint as follows:

$$u_j \geq \frac{\zeta_j^2}{\eta}$$

$$\eta u_j \geq \zeta_j^2$$

$$(\eta + u_j)^2 - (\eta - u_j)^2 \geq 4\zeta_j^2,$$

$$\sqrt{4\zeta_j^2 + (\eta - u_j)^2} \leq (\eta + u_j).$$

Hence, with dummy variables ζ_j, u_j for $j = 1, \ldots, n$, we can reformulate our inner problem as:

$$\min_{\eta \geq 0, \nu, \boldsymbol{\zeta}, \boldsymbol{u}} \left\{ \eta(\kappa - 1) + \nu + \frac{1}{4} \sum_{j=1}^{n} Q_j u_j \right\} \tag{A.1.11}$$

$$\sqrt{4\zeta_j^2 + (\eta - u_j)^2} \leq (\eta + u_j) \ \forall \ j = 1, \ldots, n \tag{A.1.12}$$

$$\zeta_j \geq \sum_{t=1}^{T} a_t R_t^{ij} - \nu + 2\eta \ \forall \ j = 1, \ldots, n \tag{A.1.13}$$

$$\zeta_j \geq 0 \ \forall \ j = 1, \ldots, n. \tag{A.1.14}$$

$$\eta \geq 0. \tag{A.1.15}$$

Therefore, combining with the outer model, we have:

$$\min_{\boldsymbol{y}, \boldsymbol{R}, \eta, \nu, \boldsymbol{\zeta}, \boldsymbol{u}} \left\{ \eta(\kappa - 1) + \nu + \frac{1}{4} \sum_{j=1}^{n} Q_j u_j \right\}, \tag{A.1.16}$$

$$\text{s.t.} \quad (2.3.1) - (2.3.8), \tag{A.1.17}$$

$$(\text{A.1.12}) - (\text{A.1.15}). \tag{A.1.18}$$

Note that, in the objective function, the $-\eta$ comes from the fact that $\eta\phi^*(s_j) = \frac{1}{4}u_j - \eta$.

Extracting Worst-case Distributions

In order to find the worst-case distribution, we must extract it from the optimal values of $\eta, \nu, \boldsymbol{\zeta}, \boldsymbol{u}$. Denote the optimal solution of (A.1.16)-(A.1.18) by $(\boldsymbol{y}^*, \boldsymbol{R}^*, \eta^*, \nu^*, \boldsymbol{\zeta}^*, \boldsymbol{u}^*)$. As discussed by Bayraksan and Love (2015), the worst-case distribution $\boldsymbol{P}^*$ satisfies:

$$\frac{P_j^*}{Q_j} \in \partial\phi^*(s_j^*), \qquad \sum_{j=1}^{n} Q_j \phi\left(\frac{P_j^*}{Q_j}\right) \leq \kappa, \qquad \sum_{j=1}^{n} P_j^* = 1.$$

Here, the notation $\partial f(x)$ is the set of *subgradients* of f at x. Suppose that $\eta^* > 0$ so that s_j^* is defined. By Bayraksan and Love (2015), if ϕ^* is differentiable then $(\phi^*)'(s_j^*)$ is a subgradient. This is true in our case, with $(\phi^*)'(s) = \max\left\{1 + \frac{s}{2}, 0\right\}$. This derivative is non-negative, and hence always gives a feasible solution for P_j^* by taking $P_j^* = Q_j(\phi^*)'(s_j^*)$ when $\eta^* > 0$. In our experiments we only ever observed $\eta^* > 0$ and hence $Q_j(\phi^*)'(s_j^*)$ always gave a solution. For more detail on how to extract the solution when $\eta^* = 0$, see Bayraksan and Love (2015).

A.2 Further Analysis of Results

A.2.1 The Effect of Workstacks on Solutions

In our experiments, we used only one value of the capacity $\boldsymbol{c}$ but varied the workstacks $\boldsymbol{D}$ to give a variety of possibilities for pulling forward. This was based on the number of pairs between which pulling forward was possible, i.e. $|\mathcal{F}^+(\boldsymbol{c}, \boldsymbol{D})|$ from Section 2.4.2. We give some examples of the values of $\boldsymbol{c} - \boldsymbol{D}$ and the corresponding $|\mathcal{F}^+(\boldsymbol{c}, \boldsymbol{D})|$ in Table A.2.1.

$\boldsymbol{c} - \boldsymbol{D}$	$\lvert\mathcal{F}^+(\boldsymbol{c}, \boldsymbol{D})\rvert$
(8, -15, -15, 8, -15)	3
(8, -15, 8, 8, 8)	5
(8, 8, 8, 8, 8)	7

Table A.2.1: Examples of $\boldsymbol{c} - \boldsymbol{D}$ values and corresponding number of pairs

Any more pairs than 7 is not possible for $T = 5$ and $K = 2$. We present a summary of the results broken down by $|\mathcal{F}^+(\boldsymbol{c}, \boldsymbol{D})|$ in Table A.2.2. This table shows three quantities: the average time taken by each algorithm, the average gaps and the average number of pairs of days which had a positive pulling forward decision. The table shows that we did not have any more non-zero decisions than 1, from any algorithm, until $|\mathcal{F}^+(\boldsymbol{c}, \boldsymbol{D})|$ reached its maximum value of 7. Days 1 and 2 were typically prioritised for rollover reduction via pulling forward in these cases. This is because jobs due on these days have the potential to roll over the most times. However, when $|\mathcal{F}^+(\boldsymbol{c}, \boldsymbol{D})| = 7$, we see between 1 and 6 pairs of days having a non-zero pulling forward decision.

The APGs are also shown in Table A.2.2. From this, we can see a number of results. Firstly, we see that the average time taken by each algorithm apart from AO was increasing in $|\mathcal{F}^+(\boldsymbol{c}, \boldsymbol{D})|$. This can be expected, since more feasible pairs leads to a more complex feasible region. Furthermore, AO performed the worse in selecting $\boldsymbol{y}$ as $|\mathcal{F}^+(\boldsymbol{c}, \boldsymbol{D})|$ increases. This is likely because reducing the set of intakes leads to less accurate approximations of the expected rollover. Interestingly, CS did not suffer from the same issue. In fact, for $|\mathcal{F}^+(\boldsymbol{c}, \boldsymbol{D})| = 7$, CS had an average $\boldsymbol{y}$-APG of 0.0%

$\lvert\mathcal{F}^+(\boldsymbol{c},\boldsymbol{D})\rvert$	Count	Alg.	Avg. $\boldsymbol{p}$-APG	Avg. $\boldsymbol{y}$-APG	Avg. t.t.	(Avg., Max) Non-zeros
3	189	opt	0.0%	0.0%	0:01:00.38	(1.0, 1)
		CS	0.0415%	0.0029%	0:00:07.58	(1.0, 1)
		CS_opt	0.0%	0.0%	0:00:09.75	(1.0, 1)
		AO	0.0084%	0.0059%	0:00:24.74	(1.0, 1)
		NP	-	-	0:00:02.62	(1.0, 1)
5	45	opt	0.0%	0.0%	0:02:12.72	(1.0, 1)
		CS	0.3465%	0.0506%	0:00:25.62	(1.0, 1)
		CS_opt	0.0%	0.0%	0:00:29.35	(1.0, 1)
		AO	0.0052%	0.2903%	0:00:19.04	(1.0, 1)
		NP	-	-	0:00:32.2	(1.0, 1)
7	45	opt	0.0%	0.0%	0:02:44.25	(2.0, 4)
		CS	0.0%	0.0%	0:01:27.85	(2.0, 4)
		CS_opt	0.0%	0.0%	0:01:31.68	(2.0, 4)
		AO	0.0%	0.5332%	0:00:21.16	(2.0, 4)
		NP	-	-	0:01:53.93	(2.8, 6)

Table A.2.2: Results by $|\mathcal{F}^+(\boldsymbol{c},\boldsymbol{D})|$.

and for all values of $|\mathcal{F}^+(\boldsymbol{c}, \boldsymbol{D})|$ this value was below 0.051%. This is because CS does not employ dimension reduction to the set of intakes like AO does.

As might be expected, there is no clear pattern in the $\boldsymbol{p}$-APGs. For AO and CS, this value was highest when $|\mathcal{F}^+(\boldsymbol{c}, \boldsymbol{D})| = 5$ and lowest when $|\mathcal{F}^+(\boldsymbol{c}, \boldsymbol{D})| = 7$. Finally, the final column shows the average and maximum numbers of pairs (t_1, t_2) that had $y_{t_1,t_2} > 0$ under each algorithm. The results for $|\mathcal{F}^+(\boldsymbol{c}, \boldsymbol{D})| = 7$ suggest that NP's solution was slightly less conservative than $\boldsymbol{P}$'s solution on average. We study this in more detail in Section 2.5.5. Interestingly, NP took almost as long as $\boldsymbol{P}$ in these instances. CS_opt again had all zero gaps and APGs, and its times taken were no more affected by $|\mathcal{F}^+(\boldsymbol{c}, \boldsymbol{D})|$ than the times taken by CS.

A.2.2 Comparison with Robust Optimisation Solutions

In this section, we compare the DRO decisions and objectives with those resulting from the robust optimisation (RO) version of the model. The RO model is obtained by replacing the inner objective with the maximisation of the total rollover cost over all intake vectors. The first result that we find is that the intake vector responsible for the worst-case cost for the chosen $\boldsymbol{y}$ value was always $\boldsymbol{i}^{\max}$. This shows that the RO model can be solved simply by assuming that all intakes take their maximum values at all times. As well as this, the RO model pulled forward less than the DRO model in 227 (82%) of our 279 instances. The RO solution also had a higher cost than the DRO solution in 269 (97%) of instances. This can be expected due to the way that their objective functions differ. These two facts support our claim that the RO model

is more conservative than the DRO model.

We present some more detailed results in Table A.2.3. This table compares the objective values, pulling forward decisions and times taken from the three models. Firstly, note that RO took around 16 seconds on average. RO also pulled forward less than DRO. Specifically, it pulled forward 1.3 jobs less than DRO, on average. Also, DRO pulled forward a maximum of 8 jobs whereas RO only pulled forward a maximum of 7 jobs. Furthermore, the objective values from RO were significantly higher than DRO. Comparing the RO objective with the DRO objective, we see that RO's objective values were around 9.5 higher than DRO's on average. This corresponds to almost a 200% increase in objective value. The $\boldsymbol{y}$-gap and $\boldsymbol{y}$-APGs assess the expected costs from RO's decisions when evaluated by DRO's objective function. This suggests that RO's decisions would result in around 2 more jobs being expected to roll over in the worst case than DRO's solution.

As already noted, RO is equivalent to the deterministic model under the assumption that $\boldsymbol{I} = \boldsymbol{i}^{\max}$ with probability 1. The results from this model are shown in the "RO det." column. This shows that this model took 0.01 seconds to build and solve, on average. Hence, our results indicate that the inclusion of the rollover constraints for the RO model led to around a 16 second increase in solution time for each instance. The inclusion of the expected value constraints for the DRO model resulted in over 1 minute of additional solution time in each instance. Table A.2.3 also shows that RO had an objective value that was three times larger than DRO's, on average.

From the results presented here, we can conclude three main results. Firstly, RO is

	RO Det.	RO	DRO
Avg. Obj. Gap	9.499	9.499	0
Avg. % Obj. Gap	199.662%	199.662%	0%
Avg. $\boldsymbol{y}$-gap	1.851	1.851	0
Avg. $\boldsymbol{y}$-APG	2.616%	2.616%	0%
Avg. $\sum_{t_1,t_2} y_{t_1,t_2}$	4	4	5.308
Max. $\sum_{t_1,t_2} y_{t_1,t_2}$	7	7	8
Avg. t.t.	0:00:00.01	0:00:15.89	0:01:22.85

Table A.2.3: Comparison of results from RO model with DRO solutions

more conservative than DRO for this problem, since it pulled forward fewer jobs on average. Secondly, RO results in significantly higher costs for the same $\boldsymbol{y}$ decision. However, the third conclusion is that RO is much faster than DRO. This indicates that the main factor affecting solution times for DRO is the inclusion of the expected value constraints.

A.3 A Benders Decomposition Approach

Our CS_opt algorithm can be viewed as a specialised *Benders decomposition* (Benders, 1962) approach that solves the distribution separation problem as a residual problem. However, it does not require us to create the dual of the distribution separation problem, and in our case we can simply solve this problem by enumeration. For comparison, we now present a classical Benders decomposition approach in order to

explain why CS_opt is preferred.

A.3.1 Residual Problem and its Dual

We create the *Benders residual problem* by taking $\boldsymbol{y}$ as the master problem variable and $\boldsymbol{R}, \vartheta$ as the *subproblem* variables. This is because the model's complexity comes from $\boldsymbol{R}$ and ϑ, not $\boldsymbol{y}$. For a fixed $\boldsymbol{y} = \bar{\boldsymbol{y}}$, the residual problem can be written as:

$$
\begin{aligned}
\min_{\boldsymbol{R},\vartheta} \quad & \vartheta, \\
\text{s.t. } & R_1^{\boldsymbol{i}} \geq i_1 + \sum_{t_1=2}^{\min\{1+K,T\}} y_{t_1,1} - (c_1 - D_1) \ \forall\, \boldsymbol{i} \in \mathcal{I} \\
& R_t^{\boldsymbol{i}} - R_{t-1}^{\boldsymbol{i}} \geq i_t + \sum_{t_1=t+1}^{\min\{t+K,T\}} y_{t_1,t} - \left(c_t - D_t + \sum_{t_2=\max\{t-K,1\}}^{t-1} y_{t,t_2} \right) \\
& \forall\, t = 2, \ldots, T-1 \ \forall\, \boldsymbol{i} \in \mathcal{I}, \\
& R_T^{\boldsymbol{i}} - R_{T-1}^{\boldsymbol{i}} \geq i_T - \left(c_T - D_T + \sum_{t_2=\max\{T-K,1\}}^{T-1} y_{t,t_2} \right) \ \forall\, \boldsymbol{i} \in \mathcal{I}, \\
& \vartheta - \sum_{t=1}^{T} a_t \sum_{\boldsymbol{i} \in \mathcal{I}} \mathbb{P}(\boldsymbol{I} = \boldsymbol{i} | \boldsymbol{p}) R_t^{\boldsymbol{i}} \geq 0 \ \forall\, \boldsymbol{p} \in \Theta.
\end{aligned}
$$

This model has $m = T|\mathcal{I}|+|\Theta|$ constraints. Hence, we have dual variables $U_{j,t}$ for $j = 1, \ldots, |\mathcal{I}|$ and $t = 1, \ldots, T$, and V_e for $e = 1, \ldots, |\Theta|$. The model has $T|\mathcal{I}|+1$ variables, and so we have $T|\mathcal{I}|+1$ constraints in the dual. The dual is given by:

$$
\max_{\boldsymbol{U},\boldsymbol{V}} \sum_{t=1}^{T} \sum_{j=1}^{n} b_{j,t}(\bar{\boldsymbol{y}}) U_{j,t} + \sum_{e=1}^{|\Theta|} \tilde{b}_e(\bar{\boldsymbol{y}}) V_e \tag{A.3.1}
$$

$$
\text{s.t.} \quad U_{j,1} - U_{j,2} - \sum_{e=1}^{|\Theta|} a_1 \mathbb{P}(\boldsymbol{I} = \boldsymbol{i}^j | \boldsymbol{p}^e) V_e \leq 0 \ \forall\, j = 1, \ldots, |\mathcal{I}|, \tag{A.3.2}
$$

$$
U_{j,t} - U_{j,t+1} - \sum_{e=1}^{|\Theta|} a_t \mathbb{P}(\boldsymbol{I} = \boldsymbol{i}^j | \boldsymbol{p}^e) V_e \leq 0 \ \forall\, j = 1, \ldots, |\mathcal{I}| \ \forall\, t = 2, \ldots, T-1, \tag{A.3.3}
$$

$$U_{j,T} - \sum_{e=1}^{|\Theta|} a_T \mathbb{P}(\boldsymbol{I} = \boldsymbol{i}^j | \boldsymbol{p}^e) V_e \leq 0 \ \forall \ j = 1, \ldots, |\mathcal{I}|, \tag{A.3.4}$$

$$\sum_{e=1}^{|\Theta|} V_e \leq 1, \tag{A.3.5}$$

$$\boldsymbol{U}, \boldsymbol{V} \geq 0, \tag{A.3.6}$$

where $b_{j,t}(\bar{\boldsymbol{y}})$ and $\tilde{b}_e(\bar{\boldsymbol{y}})$ are defined as:

$$b_{j,1}(\bar{\boldsymbol{y}}) = i_1 + \sum_{t_1=2}^{\min\{1+K,T\}} \bar{y}_{t_1,1} - (c_1 - D_1) \ \forall \ \boldsymbol{i} \in \mathcal{I}$$

$$b_{j,t}(\bar{\boldsymbol{y}}) = i_t + \sum_{t_1=t+1}^{\min\{t+K,T\}} \bar{y}_{t_1,t} - \left(c_t - D_t + \sum_{t_2=\max\{t-K,1\}}^{t-1} \bar{y}_{t,t_2} \right)$$

$$\forall \ t = 2, \ldots, T-1 \ \forall \ \boldsymbol{i} \in \mathcal{I},$$

$$b_{j,T}(\bar{\boldsymbol{y}}) = i_T - \left(c_T - D_T + \sum_{t_2=\max\{T-K,1\}}^{T-1} \bar{y}_{t,t_2} \right) \ \forall \ \boldsymbol{i} \in \mathcal{I},$$

$$\tilde{b}_e(\bar{\boldsymbol{y}}) = 0 \ \forall \ e = 1, \ldots, |\Theta|.$$

A.3.2 Benders Decomposition Algorithm

Our Benders decompositon algorithm is as follows.

1. Initialise ε, $LB = -\infty$, $UB = \infty$. Set feasible region for ξ as $\Xi = \mathbb{R}^+$. Set feasible region for $\boldsymbol{y}$ as Y, where $\boldsymbol{y} \in Y$ indicates that $\boldsymbol{y}$ is feasible for the model in (2.3.1)-(2.3.8).

2. While $UB - LB > \varepsilon$:

   (a) Solve master problem:

$$\min_{\xi \in \Xi, \boldsymbol{y} \in Y} \xi$$

to get a solution $\bar{\boldsymbol{y}}$ and objective value ξ^{M}.

(b) Set $LB = \xi^{\mathrm{M}}$.

(c) Solve Benders subproblem (A.3.1)-(A.3.6) with $\boldsymbol{y} = \bar{\boldsymbol{y}}$ to get a solution $\bar{\boldsymbol{U}}$, $\bar{\boldsymbol{V}}$ with objective ξ^{S}.

(d) If subproblem is unbounded, add feasibility cut:

$$\sum_{t=1}^{T}\sum_{j=1}^{n} b_{j,t}(\boldsymbol{y})U_{j,t} + \sum_{e=1}^{|\Theta|} \tilde{b}_e(\boldsymbol{y})V_e \leq 0$$

to Y.

(e) If subproblem is optimal, add optimality cut:

$$\xi \geq \sum_{t=1}^{T}\sum_{j=1}^{n} b_{j,t}(\boldsymbol{y})U_{j,t} + \sum_{e=1}^{|\Theta|} \tilde{b}_e(\boldsymbol{y})V_e$$

to Ξ.

(f) If $\xi^{\mathrm{S}} < UB$ then set $UB = \xi^{\mathrm{S}}$.

3. Find index of binding t constraint from the subproblem and use this to find worst-case $\boldsymbol{p}$.

4. Return $\boldsymbol{y}$, $\boldsymbol{p}$.

In the following section we will show that this approach is slow compared with CS_opt.

A.3.3 Results

We tested the Benders algorithm on each of our 279 instances, for $\varepsilon \in \{0.01, 10^{-6}, 10^{-8}\}$. We present the results in Table A.3.1. From these results, it is clear that $\varepsilon = 10^{-8}$

was required for $\boldsymbol{y}$ optimality. However, with this ε, the Benders algorithm took almost 6 minutes to solve, on average. In one instance, the algorithm timed out as it took longer than 4 hours. In comparison with CS_opt, which takes approximately 17 seconds on average, this version of Benders decomposition is very slow.

ε	Avg. $\boldsymbol{p}$-gap	Avg. $\boldsymbol{p}$-APG	Avg. $\boldsymbol{y}$-gap	Avg. $\boldsymbol{y}$-APG	Avg. t.t.	Max t.t.
1e-08	0.0	0.0%	-0.0000	0.0%	0:05:57.798	4:00:05.011
1e-06	0.0	0.0%	0.0029	0.0%	0:04:42.6717	1:24:55.625
1e-02	0.0	0.0%	0.0141	10.1984%	0:04:36.6348	1:21:58.415

Table A.3.1: Results of Benders algorithm

A.4 Large Results Tables

A.4.1 Results by $|\Theta|$

(N, M)	Avg. $\lvert\Theta\rvert$	Count	Algorithm	Avg. $\boldsymbol{p}$-APG	Avg. $\boldsymbol{y}$-APG	Avg. t.t.
$(100, 10)$	1.000	62	P	0.0%	0.0%	0:00:14.57
			CS	0.0%	0.0%	0:00:16
			CS_opt	0.0%	0.0%	0:00:15.81
			AO	0.0%	0.248%	0:00:01.32
			NP	-	-	0:00:25.58
$(50, 5)$	1.419	31	P	0.0%	0.0%	0:00:15.11
			CS	0.0%	0.0%	0:00:16.53
			CS_opt	0.0%	0.0%	0:00:16.46

			AO	0.0%	0.248%	0:00:01.35
			NP	-	-	0:00:27.13
$(50, 10)$	14.419	31	P	0.0%	0.0%	0:00:14.23
			CS	0.0%	0.0%	0:00:15.05
			CS_opt	0.0%	0.0%	0:00:15.42
			AO	0.0%	0.248%	0:00:01.4
			NP	-	-	0:00:25.22
$(10, 5)$	14.742	31	P	0.0%	0.0%	0:00:16.14
			CS	0.0%	0.0%	0:00:18.97
			CS_opt	0.0%	0.0%	0:00:19.42
			AO	0.0%	0.0004%	0:00:01.76
			NP	-	-	0:00:24.66
$(100, 15)$	16.871	31	P	0.0%	0.0%	0:00:16.24
			CS	0.0%	0.0%	0:00:16.48
			CS_opt	0.0%	0.0%	0:00:16.5
			AO	0.0452%	0.2391%	0:00:01.57
			NP	-	-	0:00:26.12
$(50, 15)$	93.129	31	P	0.0%	0.0%	0:00:24.51
			CS	0.5029%	0.0734%	0:00:20.8
			CS_opt	0.0%	0.0%	0:00:19.9
			AO	0.0105%	0.0%	0:00:02.9
			NP	-	-	0:00:27.16
$(10, 10)$	504.226	31	P	0.0%	0.0%	0:00:59.36

			CS	0.0339%	0.0%	0:00:17.85
			CS_opt	0.0%	0.0%	0:00:19.76
			AO	0.0%	0.0%	0:00:12.77
			NP	-	-	0:00:23.47
$(10, 15)$	4301.645	31	P	0.0%	0.0%	0:09:30.88
			CS	0.219%	0.0176%	0:00:19.66
			CS_opt	0.0%	0.0%	0:00:42.47
			AO	0.0031%	0.0%	0:02:11.27
			NP	-	-	0:00:23.22

Table A.4.1: Summary of results and times taken by N and M. Referred to in Section 2.5.3.

A.4.2 Results by $|\mathcal{I}|$

$\lvert\mathcal{I}\rvert$	Count	Algorithm	Avg. $\boldsymbol{p}$-APG	Avg. $\boldsymbol{y}$-APG	Avg. t.t.
392	27	P	0.0%	0.0%	0:00:33.63
		CS	0.0234%	0.0%	0:00:06.37
		CS˙opt	0.0%	0.0%	0:00:08.01
		AO	0.002%	0.0%	0:00:23.53
		NP	-	-	0:00:00.84
512	45	P	0.0%	0.0%	0:00:31.88
		CS	0.0952%	0.0024%	0:00:06.54
		CS˙opt	0.0%	0.0%	0:00:08.05

		AO	0.0%	0.0%	0:00:20.86
		NP	-	-	0:00:00.78
567	45	P	0.0%	0.0%	0:00:37.09
		CS	0.0572%	0.0107%	0:00:06.61
		CS˙opt	0.0%	0.0%	0:00:08.28
		AO	0.0116%	0.0%	0:00:25.75
		NP	-	-	0:00:00.98
2187	27	P	0.0%	0.0%	0:01:42.9
		CS	0.0536%	0.0%	0:00:08.72
		CS˙opt	0.0%	0.0%	0:00:11.69
		AO	0.0259%	0.0005%	0:00:25.93
		NP	-	-	0:00:04.25
2592	45	P	0.0%	0.0%	0:01:42
		CS	0.0043%	0.0%	0:00:09.69
		CS˙opt	0.0%	0.0%	0:00:12.82
		AO	0.0061%	0.0246%	0:00:26.58
		NP	-	-	0:00:06.18
8192	45	P	0.0%	0.0%	0:02:12.72
		CS	0.3465%	0.0506%	0:00:25.62
		CS˙opt	0.0%	0.0%	0:00:29.35
		AO	0.0052%	0.2903%	0:00:19.04

		NP	-	-	0:00:32.2
20000	45	P	0.0%	0.0%	0:02:44.25
		CS	0.0%	0.0%	0:01:27.85
		CS˙opt	0.0%	0.0%	0:01:31.68
		AO	0.0%	0.5332%	0:00:21.16
		NP	-	-	0:01:53.93

Table A.4.2: Summary of results and times taken by size of $\mathcal{I}$. Referred to in Section 2.5.3.

A.5 Tables of Notation

A.5.1 General Model Notation

Notation	**Meaning**
T	Number of days in a plan
K	Maximum number of days a job can be pulled forward
t, t_1, t_2	A day in the plan, value in $\{1, \ldots, T\}$
y_{t_1,t_2}	Number of jobs to pull forward from day $t_1 \in \{2, \ldots, T\}$ to $t_2 \in \{\max t_1 - K, 1, \ldots, t_1 - 1\}$.
R_t	Number of jobs to roll over from day t to $t+1$.
a_t	Cost of a job rolling over from day t to $t+1$.
c_t	Number of hours of capacity available on day t.

D_t	Number of jobs currently due on day t.
$\mathbb{N}_0$	Set of non-negative integers.
I_t	Random variable representing number of jobs arriving between the time of planning and day t that will be due on day t (intake).
i_t	Realisation of I_t.
$\boldsymbol{R}^i$	Realisation of $\boldsymbol{R} = (R_1, \ldots, R_T)$ corresponding to realisation i of I.
$\mathcal{I}_t$	Set of all possible realisations of I_t.
$\mathcal{I}$	Set of all possible realisations of the vector of intakes I.
$\mathcal{P}$	General ambiguity set constaining distributions of intake.
$\boldsymbol{P}$	A discrete probability distribution over the set of intakes $\mathcal{I}$.
$\boldsymbol{Q}$	Nominal distribution of intake.
$i_t^{\max}$	The maximum value I_t can take.
p_t	A variable representing success probability parameter of the binomial distribution of intake I_t.
p_t^0	True success probability of intake I_t.
$\hat{p}_t$	MLE of p_t^0 taken from N samples of I_t.
$\boldsymbol{P^p}$	Binomial distribution of intake with success probability parameter $\boldsymbol{p}$.
$\hat{\boldsymbol{P}}$	MLE of distribution resulting from $\boldsymbol{p} = \hat{\boldsymbol{p}}$.

$\mathcal{P}_\Theta$	Set of all probability distributions $\boldsymbol{P}$ that are binomial with a value $\boldsymbol{p} \in \Theta$.
Θ	Set of vectors $\boldsymbol{p}$ obtained from a distribution $\boldsymbol{P}$ in $\mathcal{P}_\Theta$.
Θ_α	$100(1-\alpha)\%$ confidence set for $\boldsymbol{p}^0$ around the MLE $\hat{\boldsymbol{P}}$.
Θ'_α	A discretisation of Θ_α.

Table A.5.1: General model notation from Section 2.3.

A.5.2 Non-parametric Model Notation

Notation	**Meaning**
ϕ	ϕ-divergence function.
d_ϕ	ϕ-divergence measure resulting from ϕ-divergence function ϕ.
ϕ^*	Conjugate of ϕ-divergence function ϕ.
$\phi_{m\chi^2}$	ϕ-divergence function for modified χ^2 distance.
$\chi^2_{o,1-\alpha}$	$100(1-\alpha)\%$ percentile of χ^2 distribution with o degrees of freedom.
$\mathcal{P}_\kappa$	Non-parametric confidence set for true distribution $\boldsymbol{P}^0$.
η, ν	Lagrange multipliers for SQP reformulation of NP model.
u_j, ζ_j, s_j	Dummy variables used to reformulate NP model.
κ	Maximum distance, measured by d_ϕ, from $\boldsymbol{Q}$ that $\boldsymbol{P}$ can be under the non-parametric model.

$\partial f(x)$	Set of subgradients of a function f at a point x.
$\cdot^*$	Optimal value of $\cdot$ under the non-parametric model, for $\cdot \in \{\boldsymbol{R}, \boldsymbol{s}, \boldsymbol{P}, \boldsymbol{y}, \nu, \eta\}$.

Table A.5.2: Notation used in the non-parametric model in Section 2.3.3

A.5.3 CS/CS_opt/AO Notation

Notation	**Meaning**
k	Index for the iteration of CS/CS_opt algorithm that we are currently carrying out.
$p_t^{\max}$	Maximum value that p_t takes over $\boldsymbol{p} \in \Theta$.
$\Theta_t^{\max}$	Set of $\boldsymbol{p}$ parameters such that p_t is maximised.
Θ^{ext}	Set of extreme parameters used by CS.
$\tilde{\Theta}$	General ambiguity set used by CS algorithms. $\tilde{\Theta} = \Theta$ for CS_opt and $\tilde{\Theta} = \Theta^{\text{ext}}$ for CS.
$k^{\max}$	Maximum number of iterations of CS/CS_opt algorithm allowed to run.
Θ^k	Current subset of Θ being used at iteration k of CS/CS_opt.
$\boldsymbol{y}^k$	Pulling forward decision generated by solving outer problem at iteration k of CS/CS_opt.
$\boldsymbol{p}^k$	Probability vector generated by solving distribution separation problem at iteration k of CS/CS_opt.
ε	Optimality tolerance of CS/CS_opt algorithm.

ϑ^k	Objective value of problem obtained by solving outer problem at iteration k of CS.
β	Minimum probability an intake must have of occurring in order to be used in the AO algorithm.
$\tilde{\mathcal{I}}$	Set of intakes with probability of occurring higher than β.

Table A.5.3: Notation used in CS/AO Algorithms (Section 2.3.6)

A.5.4 Input Parameter and Results Notation

Notation	**Meaning**
$\mathcal{F}$	Set of pairs of days between which pulling forward is allowed.
$\mathcal{F}^+(\boldsymbol{c}, \boldsymbol{D})$	Set of pairs of days between which pulling forward is feasible given $\boldsymbol{c}$ and $\boldsymbol{D}$.
$n(\boldsymbol{i}^{\max})$	Number of days with maximum intake higher than remaining capacity given $\boldsymbol{i}^{\max}$, $\boldsymbol{c}$, and $\boldsymbol{D}$.
M	Number of values each probability in $\boldsymbol{p}$ can take in our discretised ambiguity set.
$\hat{\boldsymbol{i}}$	MLE of mean intake vector.
κ	Maximum distance from $\hat{\boldsymbol{P}}$ we allow $\boldsymbol{P}$ to be under NP, measured by the chosen ϕ-divergence.

Table A.5.4: Input parameter notation used in Section 2.4

Notation	Meaning
$f(\boldsymbol{y}, \boldsymbol{p})$	Shorthand for expected rollover cost given pulling forward decision $\boldsymbol{y}$ and distribution parameter $\boldsymbol{p}$.
x	An algorithm, namely in $\{\text{S\&S}, \text{CS}, \text{AO}\}$.
$\boldsymbol{y}^x, \boldsymbol{p}^x$	$\boldsymbol{y}, \boldsymbol{p}$ solution obtained by algorithm x.
$g_{\boldsymbol{p}}(\boldsymbol{y}^x, \boldsymbol{p}^x)$	$\boldsymbol{p}$-gap of algorithm x's solution. The difference between the worst-case expected cost for $\boldsymbol{y}^x$ and the expected cost obtained by the algorithm.
z^*	Overall optimal objective value.
$g_{\boldsymbol{p}}(\boldsymbol{y}^x)$	$\boldsymbol{y}$-gap. Difference between worst-case expected cost for $\boldsymbol{y}^x$ over all distributions and the overall optimal objective value.

Table A.5.5: Results analysis notation from Section 2.5

Appendix B

Parametric Distributionally Robust Optimisation Models for Budgeted Multi-period Newsvendor Problems

B.1 Proof of Theorem 3.3.1

Proof. To prove this theorem, we need to differentiate the objective function. To do so, firstly we evaluate the expected values. For ease of notation, let $Q_t = \sum_{l=1}^{t} q_l$ and $Y_t = \sum_{l=1}^{t} X_l$. Then, $\tilde{F}_t$ and $\tilde{f}_t$ are the CDF and PDF of Y_t. In addition, we have:

$$\mathbb{E}_F\left[I_t^+\right] = \mathbb{E}_F\left[\max(I_t, 0)\right]$$

$$= \mathbb{P}(I_t \geq 0)\mathbb{E}_F(I_t|I_t \geq 0)$$

$$= \mathbb{P}(Y_t \leq Q_t)\mathbb{E}_F\left(Q_t - Y_t|Y_t \leq Q_t\right)$$

$$= \tilde{F}_t\left(Q_t\right)\left[Q_t - \mathbb{E}_{\tilde{F}_t}\left(Y_t|Y_t \leq Q_t\right)\right]$$

where $\tilde{F}_t$ and $\tilde{f}_t$ are the CDF and PDF of Y_t. Now note that we have:

$$\frac{\partial \tilde{F}_t\left(Q_t\right)}{\partial q_j} = \begin{cases} \tilde{f}_t\left(Q_t\right) & \text{if } j \leq t, \\ 0 & \text{otherwise} \end{cases}.$$

To differentiate the expected values we need the following. Firstly, the random variable $Y_t|Y_t \leq Q_t$ is Y_t truncated above at Q_t. Therefore, the PDF of this random variable is:

$$\bar{f}_t(u) = \begin{cases} \frac{\tilde{f}_t(u)}{\tilde{F}_t(Q_t)} & \text{if } u \leq Q_t \\ 0 & \text{otherwise} \end{cases}$$

and therefore:

$$\mathbb{E}_{\tilde{F}_t}\left(Y_t|Y_t \leq Q_t\right) = \int_{-\infty}^{\infty} u\bar{f}_t(u)\mathrm{d}u = \frac{\int_{-\infty}^{Q_t} u\tilde{f}_t(u)\mathrm{d}u}{\tilde{F}_t\left(Q_t\right)}.$$

This means that:

$$\mathbb{E}_F[I_t^+] = \tilde{F}_t\left(Q_t\right)\left[Q_t - \frac{\int_{-\infty}^{Q_t} u\tilde{f}_t(u)\mathrm{d}u}{\tilde{F}_t\left(Q_t\right)}\right]$$
$$= \tilde{F}_t\left(Q_t\right)Q_t - \int_{-\infty}^{Q_t} u\tilde{f}_t(u)\mathrm{d}u.$$

Applying the second fundamental theorem of calculus and the chain rule we find that:

$$\frac{\partial}{\partial q_j}\int_{-\infty}^{Q_t} u\tilde{f}_t(u)\mathrm{d}u = \frac{\partial}{\partial q_j}\left(\left[\int u\tilde{f}_t(u)\mathrm{d}u\right]\Bigg|_{u=Q_t} - \lim_{v\to-\infty}\left[\int u\tilde{f}_t(u)\mathrm{d}u\right]\Bigg|_{u=v}\right)$$
$$= \frac{\partial}{\partial q_j}\left[\int u\tilde{f}_t(u)\mathrm{d}u\right]\Bigg|_{u=Q_t}$$

$$= \left(Q_t \times \tilde{f}_t\left(Q_t\right)\right) \times \frac{\partial}{\partial q_j} Q_t$$

$$= Q_t \times \tilde{f}_t\left(Q_t\right)$$

for $j \leq t$ and 0 otherwise. Therefore:

$$\frac{\partial}{\partial q_j} \mathbb{E}_F[I_t^+] = \tilde{f}_t\left(Q_t\right) Q_t + \tilde{F}_t\left(Q_t\right) - \tilde{f}_t\left(Q_t\right) Q_t$$

$$= \tilde{F}_t\left(Q_t\right).$$

for $j \leq t$ and 0 otherwise. Similarly, we can find that:

$$\mathbb{E}_F\left[I_t^-\right] = \mathbb{E}_F\left[\max(-I_t, 0)\right]$$

$$= -\mathbb{P}(I_t \leq 0)\mathbb{E}_F(I_t | I_t \leq 0)$$

$$= \left(\tilde{F}_t\left(Q_t\right) - 1\right) Q_t + \int_{Q_t}^{\infty} u \tilde{f}_t(u) \mathrm{d}u.$$

Applying the same calculus as before, we find:

$$\frac{\partial}{\partial q_j} \int_{Q_t}^{\infty} u \tilde{f}_t(u) \mathrm{d}u = -Q_t \times \tilde{f}_t\left(Q_t\right).$$

Hence:

$$\frac{\partial}{\partial q_j} \mathbb{E}_F[I_t^-] = \tilde{f}_t\left(Q_t\right) Q_t + \left(\tilde{F}_t\left(Q_t\right) - 1\right) - Q_t \times \tilde{f}_t\left(Q_t\right)$$

$$= \tilde{F}_t\left(Q_t\right) - 1$$

for $j \leq t$, 0 otherwise. Hence, we have:

$$\frac{\partial}{\partial q_j} C_F(\boldsymbol{q}) = c\left(\tilde{F}_T\left(Q_T\right) - 1\right) + \sum_{t=j}^{T} \left[(h+b)\tilde{F}_t\left(Q_t\right) - b\right] + w_j$$

$$= \sum_{t=j}^{T} \left[(h + b + c\mathbb{1}\{t = T\})\tilde{F}_t\left(Q_t\right) - b\right] + w_j - c.$$

Note that this means that the stationarity condition (3.3.5) is equivalent to:

$$\sum_{t=j}^{T}\left[(h+b+c\mathbb{1}\{t=T\})\tilde{F}_t\left(Q_t^*\right)-b\right]=c-(1+\nu)w_j\ \forall\ j=1,\ldots,T.$$

Now, assume that $\boldsymbol{q}^*$ is as given in (3.3.9). Then, by condition 1. and 2. in Theorem 3.3.1, we have:

$$\frac{b-(1+\nu)(w_t-w_{t+1})}{h+b}\geq 0\ \forall\ t\in\{1,\ldots,T-1\}, \tag{B.1.1}$$

$$\frac{b+c-(1+\nu)w_t}{h+c+b}\geq 0, \tag{B.1.2}$$

and since $w_t-w_{t+1}\geq 0$, we also have:

$$\frac{b-(1+\nu)(w_t-w_{t+1})}{h+b}\leq\frac{b}{h+b}\leq 1\ \forall\ t\in\{1,\ldots,T-1\},$$
$$\frac{b+c-(1+\nu)w_T}{h+c+b}\leq\frac{b+c}{h+b+c}\leq 1.$$

Hence, the left hand sides of (B.1.1) and (B.1.2) lie between 0 and 1, meaning the inverse CDFs are defined at these points. Hence, $\boldsymbol{q}^*$ is well-defined. Now, note that we have:

$$\sum_{l=1}^{t}q_l^*=\begin{cases}\tilde{F}_t^{-1}\left(\frac{b-(1+\nu)(w_t-w_{t+1})}{h+b}\right) & \text{if } t<T,\\ \tilde{F}_T^{-1}\left(\frac{b-(1+\nu)w_T+c}{h+b+c}\right) & \text{if } t=T.\end{cases}$$

Substituting (3.3.9) in to the Lagrangian's derivative, we get:

$$\begin{aligned}\frac{\partial}{\partial q_j}L(\boldsymbol{q}^*,\nu)&=\sum_{t=j}^{T-1}[b-(1+\nu)(w_t-w_{t+1})-b]+[b-(1+\nu)w_T+c-b]-c+(1+\nu)w_j\\&=\sum_{t=j}^{T-1}[(1+\nu)(w_{t+1}-w_t)]-(1+\nu)w_T+(1+\nu)w_j\\&=(1+\nu)(w_T-w_j)-(1+\nu)(w_T-w_j)\\&=0.\end{aligned}$$

Hence, the stationarity condition is satisfied. □

B.2 Details on FD and its Benchmarks

B.2.1 Deriving FD

The general idea of our algorithm is similar to the multi-product algorithm of Alfares and Elmorra (2005). A high-level outline of their algorithm is as follows:

1. Set $\nu = 0$ (i.e. no budget constraint) and solve using KKT conditions, setting negative order quantities to zero.
2. If budget constraint is met then go to step 5. Else go to step 3.
3. Starting from $\nu = 0$ increase ν until the first occurrence of either of the following:
   (a) An order quantity becomes negative. Go to step 4.
   (b) Budget constraint met. Go to step 5.
4. Set the negative order quantity to zero, remove from the set of products and go to step 1.
5. Return solution.

In essence, we solve with no constraints, and increase the Lagrange multiplier ν (e.g., by line search) until either the solution becomes feasible, or one order quantity becomes negative. If an order quantity becomes negative, we remove the corresponding product from consideration and start again. The main difference between our model and that of Alfares and Elmorra (2005) is the structure of the optimal KKT solution that results from the inventory tracking. Their solution is of the form:

$$q_l = F^{-1}(P_l) \; \forall \; l \in \{1, \ldots, L\},$$

for some CDF F and probability P_l. This means that the optimal unconstrained orders for different items are independent. However, our solution is of the form:

$$q_t^* = \tilde{F}_t^{-1}(P_t) - \tilde{F}_{t-1}^{-1}(P_{t-1}), \tag{B.2.1}$$

where the subtraction of $\tilde{F}_{t-1}^{-1}(P_{t-1})$ is done to ensure that

$$\sum_{l=1}^{t} q_l^* = \tilde{F}_t^{-1}(P_t). \tag{B.2.2}$$

This relation holding true is what ensures that our KKT conditions are met. In the algorithm of Alfares and Elmorra (2005), when an item is removed and set to zero, this is because its marginal profit has become negative. Therefore, if we continue to increase ν then this value will become negative and stay negative. We can apply similar logic to our problem, with one slight difference. When Alfares and Elmorra set one variable to zero, the other variables can be found via their respective formulas and the corresponding derivatives will still be zero. However, if we set $q_{t-1}^* = 0$ then the value of q_t^* from (B.2.1) will not give (B.2.2). In fact, q_{t-1}^* being zero gives:

$$\begin{aligned}\sum_{l=1}^{t} q_l^* &= \sum_{l=1}^{t-2} q_l^* + q_t^* \\ &= \tilde{F}_{t-2}^{-1}(P_{t-2}) + (\tilde{F}_t^{-1}(P_t) - \tilde{F}_{t-1}^{-1}(P_{t-1})).\end{aligned}$$

In order to satisfy (B.2.2), the optimal value of q_t^* becomes $\tilde{F}_t^{-1}(P_t) - \tilde{F}_{t-2}^{-1}(P_{t-2})$. Hence, when we remove a day from the set of days, we need to adjust the remaining orders as well. Suppose we have a set $\mathcal{T}^0$ of days that have been set to zero permanently. Then, when re-calculating our next solution we use $\boldsymbol{q} = \tilde{\boldsymbol{q}}^*$, where $\tilde{\boldsymbol{q}}^*$ is defined by (3.3.10). This solution subtracts the sum of all previous orders from each order to ensure that (B.2.2) holds.

B.2.2 FD's Line Search Algorithm

The line search algorithm inside FD works as follows:

1. Initialise $\nu = 0$, select scaling parameter $\tau \in (0, 1)$, stepsize δ, minimum stepsize $\delta^{\min}$, budget tolerance $\tilde{\varepsilon}$.

2. Set $\boldsymbol{q}' = \boldsymbol{q} = \tilde{\boldsymbol{q}}^*(\nu)$, if $\sum_{t=1}^{T} q_t \leq W$ then set `done` $= True$, else set `done` $= False$.

3. While not `done`:

   (a) While $\nu + \delta > \nu^{\text{UB}}$:

      i. $\delta = \tau\delta$.

      ii. $\delta^{\min} = \min\{\delta, \delta^{\min}\}$.

   (b) Set $\nu' = \nu + \delta$, $\boldsymbol{q}' = \tilde{\boldsymbol{q}}^*(\nu')$:

      i. If either $\sum_{t=1}^{T} w_t q'_t > W$ or $\delta = \delta^{\min}$, then set $\nu = \nu'$.

      ii. Otherwise, define $\delta' = \tau\delta$. If $\delta' \leq \delta^{\min}$ then set $\delta^{\min} = \delta$. Otherwise set $\delta = \delta'$ and $\boldsymbol{q}' = \boldsymbol{q}$.

   (c) If either $\left(|\sum_{t=1}^{T} q'_t - W| \leq \tilde{\varepsilon} \text{ and } q'_t \geq 0 \ \forall \ t \in \mathcal{T} \right)$ or $\min \boldsymbol{q}' < 0$, set `done` $= True$.

The idea of the algorithm is as follows. We first evaluate $\tilde{\boldsymbol{q}}^*(0)$ and if this solution is feasible, we return this solution. If not, we know that ν needs to be increased in order to either make the solution feasible or make an order quantity negative. Step 3 therefore increases ν until one of these situations occurs.

Step 3(a) reduces δ until $\nu' = \nu + \delta < \nu^{\text{UB}}$ if this does not hold already. This is to ensure that we can get arbitrarily close to ν^{UB} in order to find the first negative or feasible order. Then, in step 3(b), we evaluate the order quantities for $\nu' = \nu + \delta$. If the new solution $\boldsymbol{q}'$ does not satisfy the budget constraint, then we move to 3(c) and check whether any orders are negative. If there is a negative order, we know that this occurs before the budget constraint is met, so the algorithm ends. If there are no negative orders, we know that ν has to be increased further and we go to 3(a). If $\sum_{t=1}^{T} w_t q'_t \leq W$ in 3(b)i, then we have found a solution that satisfies the budget constraint. Therefore, we need to decide if this is the first such solution, or if a smaller ν can be found that also achieves this. If $\delta = \delta^{\min}$ (which is 10^{-100} in our experiments), then it is very unlikely that this is not the first solution that satisfies the budget constraint. Hence, we move to 3(c). If, on the other hand, $\delta > \delta^{\min}$, then it is possible that we have skipped over the first solution that satisfies the budget constraint. Hence, we reduce δ in step 3(b)ii. This means that, if we need to go back to 3(a), our next solution will be closer to the first solution that satisfies the budget constraint.

B.2.3 FD's Benchmark Algorithms

The algorithms that we compare FD with are described in more detail as follows:

1. SLSQP: *Sequential Least Squares Quadratic Programming* (Kraft, 1988). This is an algorithm that can be found in Python's baseline package for scientific computing, Scipy (Virtanen et al., 2020). The algorithm is a line search algo-

rithm, where at each iteration the next search direction is chosen by solving a quadratic programming relaxation of the problem's Lagrangian. The algorithm ends when the gradient of the true objective function is sufficiently small at the candidate solution.

2. TC: *Trust Constraint* (Conn et al., 2000). This is a trust region algorithm that can be found in Scipy. Trust region algorithms are similar to line search algorithms, but where the next solution is generated by solving an approximate version of the model over a region where the approximation is "trusted". In each iteration, the ratio of the improvement in the true objective function versus the approximate objective function is used to determine whether the region should be enlarged or reduced. The algorithm ends when the gradient of the true objective function is sufficiently small at the candidate solution.

3. PLA: *Piecewise Linear Approximation*. This entails using a piecewise linear approximation to the objective function, allowing it to be solved by Gurobi (Gurobi Optimization, LLC, 2022).

B.3 Appendix for Normally Distributed Demands

B.3.1 Proof of Lemma 3.4.1

Proof. Again, let $Q_t = \sum_{l=1}^{t} q_l$ and $Y_t = \sum_{l=1}^{t} X_l$. Since the X_t's are independent normal, random variables, we have:

$$Y_t \sim \mathcal{N}\left(\sum_{l=1}^{t} \mu_l, \sum_{l=1}^{t} \sigma_l^2\right),$$

Furthermore, since $I_t = Q_t - Y_t$, we have:

$$I_t \sim \mathcal{N}\left(Q_t - \sum_{l=1}^{t} \mu_l, \sum_{l=1}^{t} \sigma_l^2\right) \ \forall\ t = 1, \ldots, T,$$

Furthermore, $I_t | I_t \geq 0$ is a normal random variable truncated below at 0. Therefore, by Barr and Sherrill (1999), we have:

$$\mathbb{E}_{\boldsymbol{\theta}}(I_t | I_t \geq 0) = \left(Q_t - \sum_{l=1}^{t} \mu_l\right) + \frac{\phi(\beta_t)}{1 - \Phi(\beta_t)} \sqrt{\sum_{l=1}^{t} \sigma_l^2},$$

where

$$\beta_t = \frac{0 - \mathbb{E}_{\boldsymbol{\theta}}(I_t)}{\sqrt{\mathrm{Var}_{\boldsymbol{\theta}}(I_t)}} = \frac{\sum_{l=1}^{t} \mu_l - Q_t}{\sqrt{\sum_{l=1}^{t} \sigma_l^2}}.$$

Therefore, we have:

$$\begin{aligned}
\mathbb{E}_{\boldsymbol{\theta}}[I_t^+] &= \mathbb{P}(I_t \geq 0)\mathbb{E}_{\boldsymbol{\theta}}[I_t | I_t \geq 0] \\
&= \mathbb{P}(Y_t \leq Q_t)\mathbb{E}_{\boldsymbol{\theta}}[I_t | I_t \geq 0] \\
&= \Phi(-\beta_t)\left(\left(Q_t - \sum_{l=1}^{t} \mu_l\right) + \frac{\phi(\beta_t)}{1 - \Phi(\beta_t)} \sqrt{\sum_{l=1}^{t} \sigma_t^2}\right) \\
&= \Phi(-\beta_t)\left(Q_t - \sum_{l=1}^{t} \mu_l\right) + \phi(\beta_t) \sqrt{\sum_{l=1}^{t} \sigma_l^2},
\end{aligned}$$

Similarly, we have:

$$\begin{aligned}
\mathbb{E}_{\boldsymbol{\theta}}[I_t^-] &= \mathbb{E}_{\boldsymbol{\theta}}[\max\{-I_t, 0\}] \\
&= \mathbb{P}(-I_t \geq 0)\mathbb{E}_{\boldsymbol{\theta}}[-I_t | -I_t \geq 0] \\
&= \Phi(\beta_t)\left(-\left(Q_t - \sum_{l=1}^{t} \mu_l\right) + \sqrt{\sum_{l=1}^{t} \sigma_l^2} \frac{\phi(-\beta_t)}{1 - \Phi(-\beta_t)}\right) \\
&= \Phi(\beta_t)\left(\sum_{l=1}^{t} \mu_l - Q_t\right) + \phi(\beta_t) \sqrt{\sum_{l=1}^{t} \sigma_l^2},
\end{aligned}$$

This means we can write the objective function as:

$$\begin{aligned}
C_{\boldsymbol{\theta}}(\boldsymbol{q}) = &\sum_{t=1}^{T} \Bigg\{ h \left[\Phi(-\beta_t) \left(Q_t - \sum_{l=1}^{t} \mu_l \right) + \phi(\beta_t) \sqrt{\sum_{l=1}^{t} \sigma_l^2} \right] \\
&+ (b + c\mathbb{1}\{t = T\}) \left[\Phi(\beta_t) \left(\sum_{l=1}^{t} \mu_l - Q_t \right) + \phi(\beta_t) \sqrt{\sum_{l=1}^{t} \sigma_l^2} \right] + w_t q_t - c\mu_t \Bigg\} \\
= &\sum_{t=1}^{T} \Bigg\{ (h(\Phi(\beta_t) - 1) + (b + c\mathbb{1}\{t = T\})\Phi(\beta_t)) \left(\sum_{l=1}^{t} \mu_l - Q_t \right) \\
&+ (h + b + c\mathbb{1}\{t = T\})\phi(\beta_t) \sqrt{\sum_{l=1}^{t} \sigma_l^2} + w_t q_t - c\mu_t \Bigg\} \\
= &\sum_{t=1}^{T} \Bigg\{ ((b + c\mathbb{1}\{t = T\} + h)\Phi(\beta_t) - h) \left(\sum_{l=1}^{t} \mu_l - Q_t \right) \\
&+ (h + b + c\mathbb{1}\{t = T\})\phi(\beta_t) \sqrt{\sum_{l=1}^{t} \sigma_l^2} + w_t q_t - c\mu_t \Bigg\}
\end{aligned}$$

or

$$C_{\boldsymbol{\theta}}(\boldsymbol{q}) = \sum_{t=1}^{T} \left\{ (a_t \Phi(\beta_t) - h) \sum_{l=1}^{t} \mu_l - Q_t + a_t \phi(\beta_t) \sqrt{\sum_{l=1}^{t} \sigma_l^2} + q_t w_t - c\mu_t \right\}.$$

Here $\mathbb{1}\{t = T\}$ is 1 if $t = T$ and 0 otherwise, and $a_t = b + c\mathbb{1}\{t = T\} + h$. □

B.3.2 Proof of Proposition 3.4.2

Proof. The likelihood function for the data is:

$$\begin{aligned}
\mathcal{L}(\boldsymbol{x}|\boldsymbol{\theta}) &= \prod_{n=1}^{N} f_{\boldsymbol{X}}(\boldsymbol{x}|\boldsymbol{\theta}) \\
&= \prod_{n=1}^{N} \prod_{t=1}^{T} \frac{1}{\sigma_t \sqrt{2\pi}} \exp\left(-\frac{1}{2\sigma_t^2} (x_t^n - \mu_t)^2 \right) \\
&\propto \left(\prod_{t=1}^{T} (\sigma_t)^{-1} \right)^N \times \exp\left(-\sum_{n=1}^{N} \sum_{t=1}^{T} \frac{1}{2\sigma_t^2} (x_t^n - \mu_t)^2 \right).
\end{aligned}$$

The log-likelihood is:

$$\ell(\boldsymbol{\theta}) = -N\sum_{t=1}^{T}\log(\sigma_t) - \sum_{n=1}^{N}\sum_{t=1}^{T}\frac{1}{2\sigma_t^2}(x_t^n - \mu_t)^2.$$

Taking derivatives, we find:

$$\begin{aligned}
\frac{\partial\ell(\boldsymbol{\theta})}{\partial\mu_t} &= \frac{1}{\sigma_t^2}\sum_{n=1}^{N}(x_t^n - \mu_t) \\
\frac{\partial\ell(\boldsymbol{\theta})}{\partial\sigma_t} &= -\frac{N}{\sigma_t} + \frac{1}{\sigma_t^3}\sum_{n=1}^{N}(x_t^n - \mu_t)^2 \\
\frac{\partial^2\ell(\boldsymbol{\theta})}{\partial\mu_t\partial\sigma_t} &= -\frac{2}{\sigma_t^3}\sum_{n=1}^{N}(x_t^n - \mu_t) \\
\frac{\partial^2\ell(\boldsymbol{\theta})}{\partial\mu_t^2} &= -\frac{N}{\sigma_t^2} \\
\frac{\partial^2\ell(\boldsymbol{\theta})}{\partial\sigma_t\partial\mu_t} &= -\frac{2}{\sigma_t^3}\sum_{n=1}^{N}(x_t^n - \mu_t) \\
\frac{\partial^2\ell(\boldsymbol{\theta})}{\partial\sigma_t^2} &= \frac{N}{\sigma_t^2} - \frac{3}{\sigma_t^4}\sum_{n=1}^{N}(x_t^n - \mu_t)^2,
\end{aligned}$$

and all other second partial derivatives are zero. As expected, this yields the MLEs:

$$\begin{aligned}
\hat{\mu}_t &= \frac{1}{N}\sum_{n=1}^{N}x_t^n \\
\hat{\sigma}_t &= \sqrt{\frac{1}{N}\sum_{n=1}^{N}(x_t^n - \hat{\mu}_t)^2}.
\end{aligned}$$

The expected values of the non-zero second derivatives are given by:

$$\begin{aligned}
\mathbb{E}_{\boldsymbol{\theta}}\left(\frac{\partial^2\ell(\boldsymbol{\theta})}{\partial\mu_t\partial\sigma_t}\right) &= 0 \\
\mathbb{E}_{\boldsymbol{\theta}}\left(\frac{\partial^2\ell(\boldsymbol{\theta})}{\partial\mu_t^2}\right) &= -\frac{N}{\sigma_t^2} \\
\mathbb{E}_{\boldsymbol{\theta}}\left(\frac{\partial^2\ell(\boldsymbol{\theta})}{\partial\sigma_t\partial\mu_t}\right) &= 0 \\
\mathbb{E}_{\boldsymbol{\theta}}\left(\frac{\partial^2\ell(\boldsymbol{\theta})}{\partial\sigma_t^2}\right) &= \frac{-2N}{\sigma_t^2}.
\end{aligned}$$

Therefore, the Fisher information matrix is given by:

$$I_{\mathbb{E}}(\boldsymbol{\theta}) = \begin{pmatrix} \frac{N}{\sigma_1^2} & 0 & \dots & \dots & \dots & \dots & \dots & 0 \\ 0 & \frac{N}{\sigma_2^2} & 0 & \dots & \dots & \dots & \dots & 0 \\ \vdots & & \ddots & & & & & \vdots \\ 0 & \dots & 0 & \frac{N}{\sigma_T^2} & 0 & \dots & \dots & 0 \\ 0 & \dots & \dots & 0 & \frac{2N}{\sigma_1^2} & 0 & \dots & 0 \\ 0 & \dots & \dots & \dots & 0 & \frac{2N}{\sigma_2^2} & \dots & 0 \\ \vdots & & & & & & \ddots & \vdots \\ 0 & \dots & \dots & \dots & \dots & \dots & 0 & \frac{2N}{\sigma_T^2} \end{pmatrix}$$

Hence, by standard results in maximum likelihood theory (see e.g. Millar (2011)), we have that (for large N):

$$\hat{\boldsymbol{\theta}} - \boldsymbol{\theta}^0 \sim \mathcal{N}\left(0, I_{\mathbb{E}}^{-1}(\boldsymbol{\theta}^0)\right),$$

which is asymptotically equivalent to:

$$\hat{\boldsymbol{\theta}} - \boldsymbol{\theta}^0 \sim \mathcal{N}\left(0, I_{\mathbb{E}}^{-1}(\hat{\boldsymbol{\theta}})\right).$$

Therefore, for large N we have:

$$(\hat{\boldsymbol{\theta}} - \boldsymbol{\theta}^0) I_{\mathbb{E}}(\hat{\boldsymbol{\theta}}) (\hat{\boldsymbol{\theta}} - \boldsymbol{\theta}^0)^T \sim \chi^2_{2T}.$$

In other words:

$$\sum_{t=1}^{T} \left(\frac{N}{\hat{\sigma}_t^2} \left(\hat{\mu}_t - \mu_t^0\right)^2 + \frac{2N}{\hat{\sigma}_t^2} \left(\hat{\sigma}_t - \sigma_t^0\right)^2 \right) \sim \chi^2_{2T}.$$

Thus, we can create an approximate $100(1-\alpha)\%$ confidence set for $\boldsymbol{\theta}^0$ using:

$$\Theta_\alpha = \left\{ (\boldsymbol{\mu}, \boldsymbol{\sigma}) \in \mathbb{R}^{2T} : \sum_{t=1}^{T} \left(\frac{N}{\hat{\sigma}_t^2} \left(\hat{\mu}_t - \mu_t\right)^2 + \frac{2N}{\hat{\sigma}_t^2} \left(\hat{\sigma}_t - \sigma_t\right)^2 \right) \leq \chi^2_{2T,1-\alpha} \right\}.$$

□

B.3.3 Piecewise Linear Approximation of DRO Model

Since the objective function (3.4.4) is nonlinear, and also not quadratic, we use a *piecewise linear approximation* (PLA) of the problem to solve it. In order to create a PLA of the model, we must define a PLA of the entire non-linear part of the objective function. Firstly, for $i = 1, \ldots, |\Theta'_\alpha|$ define $\beta_t^i = \frac{\sum_{l=1}^t (\mu_l^i - q_l)}{\sqrt{\sum_{l=1}^t (\sigma_l^i)^2}}$, where $\boldsymbol{\theta}^i$ is the i^{th} element of Θ'_α. Let $g_t^i(\beta_t^i)$ be the non-linear part of the objective function for day t, under parameter $\boldsymbol{\theta}^i \in \Theta'_\alpha$. Then, g_t^i can be written as

$$g_t^i(\beta_t^i) = \sqrt{\sum_{l=1}^t (\sigma_l^i)^2} \left\{ (a_t \Phi(\beta_t^i) - h)\beta_t^i + a_t \phi(\beta_t^i) \right\}, \ \forall\, t = 1, \ldots, T \ \forall\, i = 1, \ldots, |\Theta'_\alpha|.$$

In order to define a piecewise linear approximation of g_t^i, we need to define a set of points $Z^{i,t} = \left\{ z_1^{i,t}, \ldots, z_J^{i,t} \right\}$ such that $z_1^{i,t} \leq \beta_t^i \leq z_J^{i,t}\ \forall\, \boldsymbol{q}$. In our experiments, we will assume that these points are equally spaced, i.e. $z_j^{i,t} = z_{j-1}^{i,t} + \varepsilon$, where ε is referred to as a *gap*. Since $\beta_t^i = \frac{\sum_{l=1}^t (\mu_l^i - q_l)}{\sqrt{\sum_{l=1}^t (\sigma_l^i)^2}}$, we have:

$$(\beta_t^i)^{\min} := \frac{\sum_{l=1}^t \left(\mu_l^i - \frac{W}{w_l} \right)}{\sqrt{\sum_{l=1}^t (\sigma_l^i)^2}} \leq \beta_t^i \leq \frac{\sum_{l=1}^t \mu_l^i}{\sqrt{\sum_{l=1}^t (\sigma_l^i)^2}} =: (\beta_t^i)^{\max}$$

for each $i = 1, \ldots, |\Theta'_\alpha|$ and $t = 1, \ldots, T$. The first inequality is due to two reasons. Firstly, the maximum value q_l can take is when all other orders are zero. Secondly, this means that the budget constraint becomes $w_l q_l \leq W$ i.e. $q_l \leq \frac{W}{w_l}$. The second inequality is true since $q_l \geq 0\ \forall\, l$. We can therefore define $Z^{i,t}$ as follows:

$$Z^{i,t} = \left\{ (\beta_t^i)^{\min} + j\varepsilon : j = 0, \ldots, \left\lfloor \frac{(\beta_t^i)^{\max} - (\beta_t^i)^{\min}}{\varepsilon} \right\rfloor \right\}.$$

The reason for having a different range for each (i, t) pair is that if we were to use only one, this would be the largest range of all of the individual ranges. Having one

for each pair allows us to reduce the number of points in total that are used to define the piecewise linear model. Given the sets $Z^{i,t}$, a piecewise linear approximation of g_t^i is given by:

$$G_t^i(\beta_t^i) = \frac{\beta_t^i - z_{j+1}^{i,t}}{z_j^{i,t} - z_{j+1}^{i,t}} g_t^i(z_j^{i,t}) + \frac{\beta_t^i - z_j^{i,t}}{z_{j+1}^{i,t} - z_j^{i,t}} g_t(z_{j+1}^{i,t}),$$

where $j \in \{1, \ldots, J-1\}$ is chosen such that $\beta_t^i \in [z_j, z_{j+1}]$. Given this, we can construct a piecewise linear approximation of the DRO model via the following. Firstly, create dual-indexed dummy decision variables $\tilde{\boldsymbol{\beta}} = (\tilde{\beta}_t^i)_{i \in \{1,\ldots,|\Theta'_\alpha|\}, t \in \mathcal{T}}$ and $\tilde{\boldsymbol{g}} = (\tilde{g}_t^i)_{i \in \{1,\ldots,|\Theta'_\alpha|\}, t \in \mathcal{T}}$ such that an index i corresponds to the value of the variables when the distribution is given by $\boldsymbol{\theta}^i \in \Theta'_\alpha$. Secondly, define a dummy variable θ to represent the worst-case cost. Then, our piecewise linear model is given by:

$$\min_{\boldsymbol{q}, \tilde{\boldsymbol{g}}, \tilde{\boldsymbol{\beta}}, \vartheta} \quad \vartheta \tag{B.3.1}$$

$$\text{s.t. } \vartheta \geq \sum_{t=1}^{T} \left\{ \tilde{g}_t^i + q_t w_t - c\mu_t^i \right\} \ \forall\ i = 1, \ldots, |\Theta'_\alpha| \tag{B.3.2}$$

$$\sum_{t=1}^{T} w_t q_t \leq W, \tag{B.3.3}$$

$$\tilde{g}_t^i = G_t^i(\tilde{\beta}_t^i)\ \forall\ t \in \mathcal{T},\ \forall\ i = 1, \ldots, |\Theta'_\alpha| \tag{B.3.4}$$

$$\tilde{\beta}_t^i = \frac{\sum_{l=1}^{t} (\mu_l^i - q_l)}{\sqrt{\sum_{l=1}^{t} (\sigma_l^i)^2}}\ \forall\ t \in \mathcal{T},\ \forall\ i = 1, \ldots, |\Theta'_\alpha| \tag{B.3.5}$$

$$q_t \geq 0\ \forall\ t \in \mathcal{T}. \tag{B.3.6}$$

Our numerical experiments are conducted using Gurobi and (B.3.1)-(B.3.6) is presented for reproducibility when using this solver. Here, (B.3.5) ensures that $\tilde{\beta}_t^i = \beta_t^i$. We do not technically need a dummy variable for β_t^i as it is linear in the q_l, but Gurobi requires the PLA function G_t to be evaluated at a decision variable. Then,

(B.3.4) ensures that $\tilde{g}_t^i$ is equal to the PLA of g_t evaluated at the dummy variable $\tilde{\beta}_t^i$. Constraint (B.3.2) ensures that the value of ϑ returned by the model is equal to the worst-case expected cost for the chosen $\boldsymbol{q}$.

B.3.4 Proof of Theorem 3.4.3

Proof. In order to differentiate the objective function, we first differentiate the various terms. Let $j \in \{1, \ldots, T\}$. Then μ_j and σ_j appear in the t^{th} term of the sum for each $t \geq j$. In fact, recalling that $\beta_t = \frac{-\sum_{l=1}^t (q_l - \mu_l)}{\sqrt{\sum_{l=1}^t \sigma_l^2}}$, we have

$$\frac{\partial \beta_t}{\partial \mu_j} = \begin{cases} 0 & \text{if } j > t, \\ \frac{1}{\sqrt{\sum_{l=1}^t \sigma_l^2}}, & \text{if } j \leq t. \end{cases}$$

$$\frac{\partial \beta_t}{\partial \sigma_j} = \begin{cases} 0 & \text{if } j > t, \\ \frac{\sum_{l=1}^t (q_l - \mu_l)}{\sigma_j \left(\sum_{l=1}^t \sigma_l^2\right)^{\frac{3}{2}}} & \text{if } j \leq t. \end{cases}$$

Using the chain rule, we then find that:

$$\begin{aligned} \frac{\partial \Phi(\beta_t)}{\partial \mu_j} &= \frac{\phi(\beta_t)}{\sqrt{\sum_{l=1}^t \sigma_l^2}} \\ \frac{\partial \Phi(\beta_t)}{\partial \sigma_j} &= \frac{\sum_{l=1}^t (q_l - \mu_l)}{\sigma_j \left(\sum_{l=1}^t \sigma_l^2\right)^{\frac{3}{2}}} \phi(\beta_t) \\ &= \frac{-\beta_t}{\sigma_j \sum_{l=1}^t \sigma_l^2} \phi(\beta_t) \end{aligned}$$

for $j \leq t$. Furthermore, we have:

$$\begin{aligned} \frac{\partial \phi(\beta_t)}{\partial \mu_j} &= \frac{1}{\sqrt{\sum_{l=1}^t \sigma_l^2}} (-\beta_t \phi(\beta_t)) \\ \frac{\partial \phi(\beta_t)}{\partial \sigma_j} &= \frac{-\beta_t}{\sigma_j \sum_{l=1}^t \sigma_l^2} (-\beta_t \phi(\beta_t)) \end{aligned}$$

$$= \frac{1}{\sigma_j \sum_{l=1}^{t} \sigma_l^2} \beta_t^2 \phi(\beta_t)$$

Now, we can differentiate the objective as given in Lemma 3.4.1 to find:

$$\frac{\partial C_{\boldsymbol{\theta}}(\boldsymbol{q})}{\partial \mu_j} = \sum_{t=j}^{T} \left\{ (a_t \Phi(\beta_t) - h) + a_t \frac{\phi(\beta_t)}{\sqrt{\sum_{l=1}^{t} \sigma_l^2}} \sum_{l=1}^{t} (\mu_l - q_l) - a_t \frac{\beta_t \phi(\beta_t)}{\sqrt{\sum_{l=1}^{t} \sigma_l^2}} \sqrt{\sum_{l=1}^{t} \sigma_l^2} - c \right\}$$

and we can simplify this to:

$$\begin{aligned}
\frac{\partial C_{\boldsymbol{\theta}}(\boldsymbol{q})}{\partial \mu_j} &= \sum_{t=j}^{T} \left\{ -h + a_t \left(\Phi(\beta_t) + \frac{\phi(\beta_t)}{\sqrt{\sum_{l=1}^{t} \sigma_l^2}} \sum_{l=1}^{t} (\mu_l - q_l) \right) - a_t \beta_t \phi(\beta_t) - c \right\} \\
&= \sum_{t=j}^{T} \left\{ -h + a_t \left(\Phi(\beta_t) + \beta_t \phi(\beta_t) \right) - a_t \beta_t \phi(\beta_t) - c \right\} \\
&= \sum_{t=j}^{T} \left\{ a_t \Phi(\beta_t) - h - c \right\}
\end{aligned}$$

Furthermore, with $\Sigma_t = \sqrt{\sum_{l=1}^{t} \sigma_l^2}$ we have:

$$\frac{\partial^2 C_{\boldsymbol{\theta}}(\boldsymbol{q})}{\partial \mu_j^2} = \sum_{t=j}^{T} a_t \frac{\phi(\beta_t)}{\Sigma_t} \geq 0$$

i.e. the objective is convex in each μ_j. We also have:

$$\begin{aligned}
\frac{\partial C_{\boldsymbol{\theta}}(\boldsymbol{q})}{\partial \sigma_j} &= \sum_{t=j}^{T} \left\{ a_t \left(\frac{-\beta_t \phi(\beta_t)}{\sigma_j \sum_{l=1}^{t} \sigma_l^2} \right) \sum_{l=1}^{t} (\mu_l - q_l) + \frac{a_t \phi(\beta_t) \sigma_j}{\sqrt{\sum_{l=1}^{t} \sigma_l^2}} + a_t \sqrt{\sum_{l=1}^{t} \sigma_l^2} \frac{\alpha_t^2 \phi(\beta_t)}{\sigma_j \sum_{l=1}^{t} \sigma_l^2} \right\} \\
&= \sum_{t=j}^{T} \left\{ a_t \left(\frac{-\alpha_t^2 \phi(\beta_t)}{\sigma_j \sqrt{\sum_{l=1}^{t} \sigma_l^2}} \right) + \frac{a_t \phi(\beta_t) \sigma_j}{\sqrt{\sum_{l=1}^{t} \sigma_l^2}} + a_t \frac{\alpha_t^2 \phi(\beta_t)}{\sigma_j \sqrt{\sum_{l=1}^{t} \sigma_l^2}} \right\} \\
&= \sum_{t=j}^{T} \frac{a_t \phi(\beta_t) \sigma_j}{\sqrt{\sum_{l=1}^{t} \sigma_l^2}} \\
&\geq 0 \; \forall \; j \in \mathcal{T}.
\end{aligned}$$

Hence, $C_{\boldsymbol{\theta}}(\boldsymbol{q})$ is increasing in σ_j for all $j \in \mathcal{T}$. □

B.4 Appendix for Poisson Demands

B.4.1 Proof of Lemma 3.5.1

Proof. In order to evaluate the objective function, we need to evaluate $\mathbb{E}_{\boldsymbol{\lambda}}[I_t^+]$ and $\mathbb{E}_{\boldsymbol{\lambda}}[I_t^-]$. Recall that:

$$\mathbb{E}_{\boldsymbol{\lambda}}[I_t^+] = \mathbb{E}_{\boldsymbol{\lambda}}[\max\{I_t, 0\}]$$

$$\mathbb{E}_{\boldsymbol{\lambda}}[I_t^-] = \mathbb{E}_{\boldsymbol{\lambda}}[\max\{-I_t, 0\}]$$

where $I_t = \sum_{l=1}^t (q_t - X_t)$, $\mathbb{P}(I_t \geq 0) = \mathbb{P}\left(\sum_{l=1}^t X_l \leq \sum_{l=1}^t q_l\right)$ and $\mathbb{E}_{\boldsymbol{\lambda}}[I_t | I_t \geq 0] = \sum_{l=1}^t q_l - \mathbb{E}_{\boldsymbol{\lambda}}[\sum_{l=1}^t X_l | \sum_{l=1}^t X_l \leq \sum_{l=1}^t q_l]$. To simplify notation, let us define $Q_t = \sum_{l=1}^t q_l$ and $\Lambda_t = \sum_{l=1}^t \lambda_l$. Now, we can calculate the expected values as follows:

$$\begin{aligned}
\mathbb{E}_{\boldsymbol{\lambda}}[\max\{I_t, 0\}] &= \mathbb{E}_{\boldsymbol{\lambda}}\left[\max\left\{Q_t - \sum_{l=1}^t X_l, 0\right\}\right] \\
&= \sum_{x=0}^{\infty} \max\{Q_t - x, 0\} \tilde{f}_t(x) \\
&= \sum_{x=0}^{Q_t} (Q_t - x) \tilde{f}_t(x) \\
&= Q_t \tilde{F}_t(Q_t) - \sum_{x=0}^{Q_t} x \frac{(\Lambda_t)^x \exp(-\Lambda_t)}{(x)!} \\
&= Q_t \tilde{F}_t(Q_t) - \sum_{x=1}^{Q_t} \frac{(\Lambda_t)^x \exp(-\Lambda_t)}{(x-1)!} \\
&= Q_t \tilde{F}_t(Q_t) - \Lambda_t \sum_{x=1}^{Q_t} \frac{(\Lambda_t)^{x-1} \exp(-\Lambda_t)}{(x-1)!} \\
&= Q_t \tilde{F}_t(Q_t) - \Lambda_t \tilde{F}_t(Q_t - 1),
\end{aligned}$$

where $\tilde{f}_t$ and $\tilde{F}_t$ are the PMF and CDF of $\sum_{l=1}^t X_l$. Similarly, we have:

$$\mathbb{E}_{\boldsymbol{\lambda}}[\max\{-I_t, 0\}] = \sum_{x=0}^{\infty} \max\{x - Q_t, 0\} \tilde{f}_t(x)$$

$$
\begin{aligned}
&= \sum_{x=Q_t+1}^{\infty} (x - Q_t)\tilde{f}_t(x) \\
&= -Q_t(1 - \tilde{F}_t(Q_t)) + \sum_{x=Q_t+1}^{\infty} x\frac{(\Lambda_t)^x \exp(-\Lambda_t)}{(x)!} \\
&= -Q_t(1 - \tilde{F}_t(Q_t)) + \Lambda_t - \sum_{x=0}^{Q_t} x\frac{(\Lambda_t)^x \exp(-\Lambda_t)}{(x)!} \\
&= -Q_t(1 - \tilde{F}_t(Q_t)) + \Lambda_t - \sum_{x=1}^{Q_t} \frac{(\Lambda_t)^x \exp(-\Lambda_t)}{(x-1)!} \\
&= -Q_t(1 - \tilde{F}_t(Q_t)) + \Lambda_t(1 - \tilde{F}_t(Q_t - 1)).
\end{aligned}
$$

This gives the following objective function (with $a_t = b + c\mathbb{1}\{t = T\} + h$):

$$
\begin{aligned}
C_{\boldsymbol{\lambda}}(\boldsymbol{q}) &= c\mathbb{E}_{\boldsymbol{\lambda}}[I_T^-] + \sum_{t=1}^{T}(h\mathbb{E}_{\boldsymbol{\lambda}}[I_t^+] + b\mathbb{E}_{\boldsymbol{\lambda}}[I_t^-] + w_t q_t - c\mathbb{E}_{\boldsymbol{\lambda}}[X_t]) \\
&= \sum_{t=1}^{T}(h\mathbb{E}_{\boldsymbol{\lambda}}[I_t^+] + (b + c\mathbb{1}\{t = T\})\mathbb{E}_{\boldsymbol{\lambda}}[I_t^-] + w_t q_t - c\mathbb{E}_{\boldsymbol{\lambda}}[X_t]) \\
&= \sum_{t=1}^{T}\Big(h(Q_t\tilde{F}_t(Q_t) - \Lambda_t\tilde{F}_t(Q_t - 1)) + (b + c\mathbb{1}\{t = T\})(-Q_t(1 - \tilde{F}_t(Q_t)) \\
&\quad + \Lambda_t(1 - \tilde{F}_t(Q_t - 1))) + w_t q_t - c\lambda_t\Big) \\
&= \sum_{t=1}^{T}\Big(a_t Q_t\tilde{F}_t(Q_t) - \Lambda_t a_t\tilde{F}_t(Q_t - 1) + (b + c\mathbb{1}\{t = T\})(\Lambda_t - Q_t) + w_t q_t - c\lambda_t\Big),
\end{aligned}
$$

as required. □

B.4.2 Proof of Proposition 3.5.2

Proof. By standard results in maximum likelihood estimation (Millar, 2011), for large N we have the approximate result that:

$$
\hat{\lambda}_t \sim \mathcal{N}\left(\lambda_t^0, \frac{\lambda_t^0}{N}\right),
$$

or equivalently:

$$\hat{\lambda}_t \sim \mathcal{N}\left(\lambda_t^0, \frac{\hat{\lambda}_t}{N}\right).$$

Now, by independence of the T MLEs, we have the approximate result that:

$$\sum_{t=1}^{T} \frac{N}{\hat{\lambda}_t}(\hat{\lambda}_t - \lambda_t^0)^2 \leq \chi_T^2.$$

Hence, we can calculate an approximate $100(1-\alpha)\%$ confidence set for $\boldsymbol{\lambda}^0$ using:

$$\Theta_\alpha = \left\{\boldsymbol{\lambda} \in \mathbb{R}_+^T : \sum_{t=1}^{T} \frac{N}{\hat{\lambda}_t}(\hat{\lambda}_t - \lambda_t^0)^2 \leq \chi_{T,1-\alpha}^2\right\}.$$

□

B.4.3 Piecewise Linear DRO Model

Note that, for Poisson demands, the objective function is already piecewise linear. The Poisson CDF $\tilde{F}_t$ is piecewise constant and hence piecewise linear. In addition, the non-linear term $Q_t\tilde{F}_t(Q_t)$ is piecewise linear, since $\tilde{F}_t(Q_t)$ is piecewise constant. Since $\tilde{F}_t(Q_t)$ is constant on the interval $[Q_t, Q_t + 1)$ for any $Q_t \in \mathbb{N}_0$, the objective function is linear in these intervals. Therefore, we only need to consider $\varepsilon = 1$ as the gap between points for this model, and in addition the piecewise linear objective function is not approximate.

We now present the piecewise linear model formulation, for reproducibility when using Gurobi. Despite the fact that the cost function is already piecewise linear, we still use Gurobi's piecewise linear constraints in order to formulate the model. In order to do so, define $\boldsymbol{\lambda}^i$ as the i^{th} element of Θ'_α and define:

$$D_t^i(Q) = a_t Q\tilde{F}_t^i(Q) - \Lambda_t^i a_t \tilde{F}_t^i(Q-1) + (b + c\mathbb{1}\{t = T\})(\Lambda_t^i - Q) + w_t Q - c\lambda_t^i$$

for each $i = 1, \dots |\Theta'_\alpha|$ and $t = 1, \dots, T$, where $\tilde{F}_t^i$ is the CDF of $\sum_{l=1}^t X_l$ under $\boldsymbol{\lambda}^i$. To define the piecewise linear constraint in Gurobi, we need a set $U^t = \{u_1^t, \dots, u_{J'}^t\}$ for each $t \in \mathcal{T}$ such that $u_1^t \leq Q_t \leq u_{J'}^t$ holds for any $\boldsymbol{q}$. Since $Q_t = \sum_{l=1}^t q_l$, it is easy to see that:

$$0 \leq Q_t \leq \sum_{l=1}^{t} \frac{W}{w_l} \ \forall \ t = 1, \dots, T.$$

Hence, we take $U^t = \left\{0, 1, \dots, \sum_{l=1}^t \frac{W}{w_l}\right\}$ as our points for each t. Then, the model can be formulated using:

$$\begin{aligned}
\min_{\boldsymbol{q},\boldsymbol{Q},\boldsymbol{d},\vartheta} \ & \vartheta \\
\text{s.t. } & \vartheta \geq \sum_{t=1}^{T} d_t^i \ \forall \ i = 1, \dots, |\Theta'_\alpha|, \\
& Q_t = \sum_{l=1}^{t} q_t \ \forall \ t = 1, \dots, T, \\
& d_t^i = D_t^i(Q_t), \ \forall \ i = 1, \dots, |\Theta'_\alpha| \ \forall \ t = 1, \dots, T \\
& \sum_{t=1}^{T} w_t q_t \leq W \\
& q_t \in \mathbb{N}_0 \ \forall \ t = 1, \dots, T.
\end{aligned}$$

B.4.4 Proof of Theorem 3.5.3

Proof. In order to differentiate $C_{\boldsymbol{\lambda}}(\boldsymbol{q})$, we first differentiate the Poisson CDF to find that:

$$\begin{aligned}
\frac{\partial \tilde{F}_t(Q_t)}{\partial \lambda_j} &= \frac{\partial}{\partial \lambda_j} \left(\sum_{x=0}^{Q_t} \frac{\exp\left(-\Lambda_t\right) \left(\Lambda_t\right)^x}{x!} \right) \\
&= \sum_{x=0}^{Q_t} \frac{-\exp\left(-\Lambda_t\right) \left(\Lambda_t\right)^x + \exp\left(-\Lambda_t\right) x \left(\Lambda_t\right)^{x-1}}{x!}
\end{aligned}$$

$$
\begin{aligned}
&= \sum_{x=1}^{Q_t} \frac{\exp\left(-\Lambda_t\right)\left(\Lambda_t\right)^{x-1}}{(x-1)!} - \sum_{x=0}^{Q_t} \frac{\exp\left(-\Lambda_t\right)\left(\Lambda_t\right)^{x}}{x!} \\
&= \sum_{x=0}^{Q_t-1} \frac{\exp\left(-\Lambda_t\right)\left(\Lambda_t\right)^{x}}{x!} - \sum_{x=0}^{Q_t} \frac{\exp\left(-\Lambda_t\right)\left(\Lambda_t\right)^{x}}{x!} \\
&= \tilde{F}_t(Q_t - 1) - \tilde{F}_t(Q_t) \\
&= -\tilde{f}_t(Q_t)
\end{aligned}
$$

for $j \leq t$. Similarly,

$$
\frac{\partial \tilde{F}_t(Q_t - 1)}{\partial \lambda_j} = -\tilde{f}_t(Q_t - 1) \quad (j \leq t).
$$

Hence, using the expression given in Lemma 3.5.1, we have:

$$
\frac{\partial C_{\boldsymbol{\lambda}}(\boldsymbol{q})}{\partial \lambda_j} = \sum_{t=j}^{T} \Big(-a_t Q_t \tilde{f}_t(Q_t) + \Lambda_t a_t \tilde{f}_t(Q_t - 1) - a_t \tilde{F}_t(Q_t - 1) + (a_t - h) - c \Big) \tag{B.4.1}
$$

$$
= \sum_{t=j}^{T} \Big(-a_t \tilde{F}_t(Q_t - 1) + (a_t - h) - c \Big) \tag{B.4.2}
$$

where (B.4.2) is true because:

$$
\begin{aligned}
Q_t \tilde{f}_t(Q_t) &= Q_t \frac{\exp(-\Lambda_t)(\Lambda_t)^{Q_t}}{Q_t!} \\
&= \frac{\exp(-\Lambda_t)(\Lambda_t)^{Q_t}}{(Q_t - 1)!} \\
&= \Lambda_t \tilde{f}_t(Q_t - 1).
\end{aligned}
$$

Differentiating again, we find:

$$
\begin{aligned}
\frac{\partial^2 C_{\boldsymbol{\lambda}}(\boldsymbol{q})}{\partial \lambda_j^2} &= \sum_{t=j}^{T} a_t \tilde{f}_t(Q_t - 1) \\
&\geq 0 \ \forall \ j = 1, \ldots, T.
\end{aligned}
$$

Therefore, the objective function is convex in λ_j for each j. □

Appendix C

Robust Markov Decision Processes Under Parametric Transition Distributions

C.1 Derivation of Reformulation of Robust Bellman Update

C.1.1 General Reformulation

The inner problem of (4.3.2) can be written as:

$$\min_{\boldsymbol{P}_s} \sum_{a\in\mathcal{A}} \pi_{s,a} \sum_{s'\in\mathcal{S}} P_{s,a,s'} B_{s,a,s'}$$

$$\text{s.t. } \sum_{a\in\mathcal{A}} d_a(\boldsymbol{P}_{s,a}, \hat{\boldsymbol{P}}_{s,a}) \leq \kappa$$

$$\sum_{s' \in \mathcal{S}} P_{s,a,s'} = 1 \; \forall \; a \in \mathcal{A},$$

with $\boldsymbol{B}_{s,a} = \boldsymbol{r}_{s,a} + \gamma \boldsymbol{v}$ and $\boldsymbol{v} = \boldsymbol{v}^n$. The Lagrangian of this problem is given by:

$$\begin{aligned} L(\boldsymbol{\pi}, \boldsymbol{\nu}, \eta) &= \sum_{a \in \mathcal{A}} \pi_{s,a} \sum_{s' \in \mathcal{S}} P_{s,a,s'} B_{s,a,s'} + \eta \left(\sum_{a \in \mathcal{A}} d_a(\boldsymbol{P}_{s,a}, \hat{\boldsymbol{P}}_{s,a}) - \kappa \right) + \sum_{a \in \mathcal{A}} \nu_a \left(1 - \sum_{s' \in \mathcal{S}} P_{s,a,s'} \right) \\ &= -\kappa\eta + \nu + \sum_{a \in \mathcal{A}} \sum_{s' \in \mathcal{S}} \left[\pi_{s,a} P_{s,a,s'} B_{s,a,s'} + \eta \hat{P}_{s,a,s'} \phi \left(\frac{P_{s,a,s'}}{\hat{P}_{s,a,s'}} \right) - \nu_a P_{s,a,s'} \right], \end{aligned}$$

with $\nu = \sum_{a \in \mathcal{A}} \nu_a$. Therefore, the objective of the dual of the inner problem is given by:

$$\begin{aligned} g(\boldsymbol{\nu}, \eta) &= \inf_{\boldsymbol{P}_s \geq 0} \left\{ -\kappa\eta + \nu + \sum_{a \in \mathcal{A}} \sum_{s' \in \mathcal{S}} \left[\pi_{s,a} P_{s,a,s'} B_{s,a,s'} + \eta \hat{P}_{s,a,s'} \phi \left(\frac{P_{s,a,s'}}{\hat{P}_{s,a,s'}} \right) - \nu_a P_{s,a,s'} \right] \right\} \\ &= -\kappa\eta + \nu + \sum_{a \in \mathcal{A}} \sum_{s' \in \mathcal{S}} \inf_{P_{s,a,s'} \geq 0} \left\{ \pi_{s,a} P_{s,a,s'} B_{s,a,s'} + \eta \hat{P}_{s,a,s'} \phi \left(\frac{P_{s,a,s'}}{\hat{P}_{s,a,s'}} \right) - \nu_a P_{s,a,s'} \right\} \\ &= -\kappa\eta + \nu + \sum_{a \in \mathcal{A}} \sum_{s' \in \mathcal{S}} \eta \hat{P}_{s,a,s'} \inf_{P_{s,a,s'} \geq 0} \left\{ \frac{P_{s,a,s'}}{\hat{P}_{s,a,s'}} \frac{\pi_{s,a} B_{s,a,s'}}{\eta} + \phi \left(\frac{P_{s,a,s'}}{\hat{P}_{s,a,s'}} \right) - \frac{\nu_a}{\eta} \frac{P_{s,a,s'}}{\hat{P}_{s,a,s'}} \right\} \\ &= -\kappa\eta + \nu + \sum_{a \in \mathcal{A}} \sum_{s' \in \mathcal{S}} \eta \hat{P}_{s,a,s'} \inf_{P_{s,a,s'} \geq 0} \left\{ \frac{P_{s,a,s'}}{\hat{P}_{s,a,s'}} \frac{\pi_{s,a} B_{s,a,s'} - \nu_a}{\eta} + \phi \left(\frac{P_{s,a,s'}}{\hat{P}_{s,a,s'}} \right) \right\} \\ &= -\kappa\eta + \nu - \sum_{a \in \mathcal{A}} \sum_{s' \in \mathcal{S}} \eta \hat{P}_{s,a,s'} \sup_{P_{s,a,s'} \geq 0} \left\{ \frac{P_{s,a,s'}}{\hat{P}_{s,a,s'}} \frac{\nu_a - \pi_{s,a} B_{s,a,s'}}{\eta} - \phi \left(\frac{P_{s,a,s'}}{\hat{P}_{s,a,s'}} \right) \right\} \\ &= -\kappa\eta + \nu - \sum_{a \in \mathcal{A}} \sum_{s' \in \mathcal{S}} \eta \hat{P}_{s,a,s'} \phi^* \left(\frac{\nu_a - \pi_{s,a} B_{s,a,s'}}{\eta} \right). \end{aligned}$$

Here, we used $\inf(A) = -\sup(-A)$ for any set A in order to replace inf with sup. Therefore, the dual of the inner problem is given by:

$$\max_{\eta, \boldsymbol{\nu}} \left\{ -\kappa\eta + \nu - \sum_{a \in \mathcal{A}} \sum_{s' \in \mathcal{S}} \eta \hat{P}_{s,a,s'} \phi^* \left(\frac{\nu_a - \pi_{s,a} B_{s,a,s'}}{\eta} \right) : \eta \in \mathbb{R}_+, \boldsymbol{\nu} \in \mathbb{R}^A \right\}.$$

Combining with the outer problem, our reformulation is given by:

$$\max_{\boldsymbol{\pi}_s, \eta, \boldsymbol{\nu}} \left\{ -\kappa\eta + \nu - \sum_{a \in \mathcal{A}} \sum_{s' \in \mathcal{S}} \eta \hat{P}_{s,a,s'} \phi^* \left(\frac{\nu_a - \pi_{s,a} B_{s,a,s'}}{\eta} \right) \right\}$$

$$\text{s.t.} \sum_{a \in \mathcal{A}} \pi_{s,a} = 1$$

$$\pi_{s,a} \geq 0 \; \forall \; (s,a) \in \mathcal{S} \times \mathcal{A}$$

$$\eta \in \mathbb{R}_+, \boldsymbol{\nu} \in \mathbb{R}^A.$$

C.1.2 Reformulation for Modified χ^2-divergence

For the modified χ^2-divergence, we have:

$$\phi^*(z) = \max\left\{1 + \frac{z}{2}, 0\right\}^2 - 1.$$

Hence, we have:

$$\begin{aligned}\eta\phi^*\left(\frac{\nu_a - \pi_{s,a}B_{s,a,s'}}{\eta}\right) &= \eta \max\left\{1 + \frac{\nu_a - \pi_{s,a}B_{s,a,s'}}{2\eta}, 0\right\}^2 - \eta \\ &= \frac{1}{4\eta}\max\left\{2\eta + \nu_a - \pi_{s,a}B_{s,a,s'}, 0\right\}^2 - \eta.\end{aligned}$$

We can reformulate this using conic quadratic constraints as follows. Firstly, define the dummy variables $\zeta_{s,a,s'}$ for $a \in \mathcal{A}$ and $s' \in \mathcal{S}$ using the following constraints:

$$\zeta_{s,a,s'} \geq 2\eta + \nu_a - \pi_{s,a}B_{s,a,s'} \; \forall \; a \in \mathcal{A} \; \forall \; s' \in \mathcal{S}$$

$$\zeta_{s,a,s'} \geq 0 \; \forall \; a \in \mathcal{A} \; \forall \; s' \in \mathcal{S}.$$

Now define dummy variables $u_{s,a,s'} \; \forall \; a \in \mathcal{A} \; \forall \; s' \in \mathcal{S}$ using:

$$u_{s,a,s'} \geq \frac{\zeta_{s,a,s'}^2}{\eta}$$

which is equivalently represented by:

$$\sqrt{4\zeta_{s,a,s'}^2 + (\eta - u_{s,a,s'})^2} \leq (\eta + u_{s,a,s'}).$$

Then, at optimality we will have $\eta\phi^*\left(\frac{\nu_a-\pi_{s,a}B_{s,a,s'}}{\eta}\right) = \frac{1}{4}u_{s,a,s'}-\eta$. Therefore, the CQP reformulation of (4.3.2) is given by:

$$\begin{aligned}
&\max_{\boldsymbol{\pi}_s}\left\{\nu+\eta(A-\kappa)-\frac{1}{4}\sum_{a\in\mathcal{A}}\sum_{s'\in\mathcal{S}}\hat{P}_{s,a,s'}u_{s,a,s'}\right\}\\
&\text{s.t. } \sqrt{4\zeta_{s,a,s'}^2+(\eta-u_{s,a,s'})^2}\leq(\eta+u_{s,a,s'})\ \forall\ a\in\mathcal{A}\ \forall\ s'\in\mathcal{S}\\
&\quad\zeta_{s,a,s'}\geq 2\eta+\nu_a-\pi_{s,a}B_{s,a,s'}\ \forall\ a\in\mathcal{A}\ \forall\ s'\in\mathcal{S}\\
&\quad\zeta_{s,a,s'}\geq 0\ \forall\ a\in\mathcal{A}\ \forall\ s'\in\mathcal{S}\\
&\quad u_{s,a,s'}\geq 0\ \forall\ a\in\mathcal{A}\ \forall\ s'\in\mathcal{S}\\
&\quad\sum_{a\in\mathcal{A}}\pi_{s,a}=1\\
&\quad\pi_{s,a}\geq 0\ \forall\ a\in\mathcal{A}\\
&\quad\eta\geq 0\\
&\quad\boldsymbol{\nu}\in\mathbb{R}^A.
\end{aligned}$$

Note that the term $A\eta$ comes from $\sum_{a\in\mathcal{A}}\sum_{s'\in\mathcal{S}}\hat{P}_{s,a,s'}\eta = A\eta$.

C.2 Solving Modified χ^2-divergence Projection Problems

Since we focus on the modified χ^2-divergence in this paper, we will describe the method for this distance only.

C.2.1 Solution by Sorting and Subproblems

The method used by Ho et al. (2022) consists of the following steps. Firstly, use Lagrangian duality to reformulate the projection problem (4.3.7) as:

$$\max_{\xi,\psi} -\beta\xi + \psi - \sum_{s'\in\mathcal{S}} \hat{P}_{s,a,s'}\phi^*(-\xi b_{s'} + \psi)$$

$$\text{s.t. } \xi \in \mathbb{R}^+, \psi \in \mathbb{R}.$$

Then, recalling that $\phi^*(z) = \max\left\{1 + \frac{z}{2}, 0\right\}^2 - 1$, we wish to eliminate the max operator in order to make the model tractable. In order to do so, we observe that at optimality, we necessarily have that $\phi^*(-\xi b_{s'} + \psi) = -1$ holds for exactly $\hat{S}$ values of s', for some $\hat{S} \in \{0, \ldots, S\}$. In order to find the optimal solution, we can therefore solve the model resulting from enforcing each value of $\hat{S}$ explicitly, and select the solution with the best objective value. In order to do so, w.l.o.g. we sort the elements of $\boldsymbol{b}$ so that they are non-increasing. Then, for each $\hat{S} \in \{0, \ldots, S\}$ we create a subproblem of the reformulated projection problem by constraining ξ, ψ to enforce that $\phi^*(-\xi b_{s'} + \psi) = -1$ for each $s' \in \{1, \ldots, \hat{S}\}$. The final term in the objective function becomes:

$$\begin{aligned}
-\sum_{s'\in\mathcal{S}} \hat{P}_{s,a,s'}\phi^*(-\xi b_{s'} + \psi) &= \sum_{s'=1}^{\hat{S}} \hat{P}_{s,a,s'} - \sum_{s'=\hat{S}+1}^{S} \hat{P}_{s,a,s'} \left(\left(1 + \frac{-\xi b_{s'} + \psi}{2}\right)^2 - 1 \right) \\
&= \sum_{s'=1}^{\hat{S}} \hat{P}_{s,a,s'} - \sum_{s'=\hat{S}+1}^{S} \hat{P}_{s,a,s'} \left((-\xi b_{s'} + \psi) + \frac{(-\xi b_{s'} + \psi)^2}{4} \right).
\end{aligned}$$

Therefore, the subproblem is given by (C.2.1)-(C.2.4).

$$\max_{\xi,\psi} -\beta\xi + \psi + \sum_{s'=1}^{\hat{S}} \hat{P}_{s,a,s'} - \sum_{s'=\hat{S}+1}^{S} \hat{P}_{s,a,s'} \left((-\xi b_{s'} + \psi) + \frac{(-\xi b_{s'} + \psi)^2}{4} \right) \quad \text{(C.2.1)}$$

$$\text{s.t. } -\xi b_{s'} + \psi \leq -2 \;\forall\; s' \in \{1, \ldots, \hat{S}\} \quad \text{(C.2.2)}$$

$$-\xi b_{s'} + \psi \geq -2 \;\forall\; s' \in \{\hat{S}+1, \ldots, S\} \tag{C.2.3}$$

$$\xi \in \mathbb{R}_+, \psi \in \mathbb{R}. \tag{C.2.4}$$

Note that for $\hat{S} = 0$, constraint (C.2.2) is redundant and can be removed. Similarly, for $\hat{S} = S$, constraint (C.2.3) can be removed. Given this formulation, the solution of the subproblem is obtained from solving at most 3 problems, each with an analytical solution. By Ho et al. (2022), for a fixed $\hat{S}$ and ξ, the solution of this subproblem in ψ is given by:

$$\psi^* = \begin{cases} -2 + \xi b_{\hat{S}+1} & \text{if } H(\xi) \leq -2 + \xi b_{\hat{S}+1} \\ -2 + \xi b_{\hat{S}} & \text{if } H(\xi) \geq -2 + \xi b_{\hat{S}} \\ H(\xi) & \text{otherwise,} \end{cases} \tag{C.2.5}$$

where

$$H(\xi) = \frac{2\sum_{s'=1}^{\hat{S}} \hat{P}_{s,a,s'} + \xi \sum_{s'=\hat{S}+1}^{S} b_{s'} \hat{P}_{s,a,s'}}{\sum_{s'=\hat{S}+1}^{S} \hat{P}_{s,a,s'}}.$$

For some border cases, we do not need to solve the problem in all 3 of these cases. In particular, we have the following special cases:

1. $\hat{S} = 0$. In this case, the second case is not defined as $b_{\hat{S}}$ does not exist.

2. $\hat{S} = S - 1$. In this case we have:

$$\begin{aligned} H(\xi) &= \frac{2\sum_{s'=1}^{S-1} \hat{P}_{s,a,s'} + \xi b_S \hat{P}_{s,a,S}}{\hat{P}_{s,a,S}} \\ &= \frac{2(1 - \hat{P}_{s,a,S})}{\hat{P}_{s,a,S}} + b_S \xi \\ &= \frac{2}{\hat{P}_{s,a,S}} + (-2 + b_S \xi) \\ &> -2 + \xi b_S \end{aligned}$$

$$= -2 + \xi b_{\hat{S}+1}.$$

Hence, the first case in (C.2.5) is impossible. In addition, for $\hat{S} = S - 1$ the problem becomes:

$$\max_{\xi,\psi} -\beta\xi + \psi + \sum_{s'=1}^{S-1} \hat{P}_{s,a,s'} - \hat{P}_{s,a,S}\left((-\xi b_S + \psi) + \frac{(-\xi b_S + \psi)^2}{4}\right)$$

$$\text{s.t.} \quad -\xi b_{s'} + \psi \leq -2 \;\forall\; s' \in \{1, \ldots, S-1\}$$

$$-\xi b_S + \psi \geq -2$$

$$\xi \in \mathbb{R}_+, \psi \in \mathbb{R}.$$

In the third case of (C.2.5), we have $\psi = \frac{2}{\hat{P}_{s,a,S}} + (-2 + b_S\xi)$ and so the objective function is given by:

$$-\beta\xi + \frac{2}{\hat{P}_{s,a,S}} + (-2 + b_S\xi) - \hat{P}_{s,a,S}\left(\left(\frac{2}{\hat{P}_{s,a,S}} - 2\right) + \frac{1}{4}\left(\frac{2}{\hat{P}_{s,a,S}} - 2\right)^2\right).$$

Therefore, the derivative of the objective function is $(b_S - \beta)\xi \leq 0 \;\forall\; \xi \geq 0$, since $\beta \geq \min \boldsymbol{b} = b_S$. Hence, ξ should be set at zero if it is unconstrained.

3. $\hat{S} = S$. In this case, the problem becomes (C.2.6)-(C.2.8):

$$\max_{\xi,\psi} -\beta\xi + \psi + 1 \tag{C.2.6}$$

$$\text{s.t.} \quad -\xi b_{s'} + \psi \leq -2 \;\forall\; s' \in \{1, \ldots, S\} \tag{C.2.7}$$

$$\xi \in \mathbb{R}_+, \psi \in \mathbb{R}. \tag{C.2.8}$$

Constraint (C.2.7) implies that $\psi \leq \xi b_S - 2$. Since the objective is increasing in ψ, this means $\psi^* = -2 + \xi b_S$. Hence, the second case in (C.2.5) is guaranteed. Furthermore, the objective is given by $\max \xi(b_S - \beta) - 1$. Since the assumption

made by Ho et al. (2022) is that $\min \boldsymbol{b} \leq \beta$ and $b_S = \min \boldsymbol{b}$, the objective is decreasing in ξ and so the optimal solution is $(\xi^*, \psi^*) = (0, -2)$. The optimal objective value is -1.

Now, in each case defined by (C.2.5), the problem can be reformulated as a univariate program with one constraint. In the first case, Ho et al. (2022) show that the model becomes:

$$\max_{\xi} \left\{ -\beta\xi + \xi b_{\hat{S}+1} - 2 + \sum_{s'=1}^{\hat{S}} \hat{P}_{s,a,s'} - \sum_{s'=\hat{S}+1}^{S} \hat{P}_{s,a,s'} \left((-\xi b_{s'} + \xi b_{\hat{S}+1} - 2) + \frac{(-\xi b_{s'} + \xi b_{\hat{S}+1} - 2)^2}{4} \right) \right\}$$
$$\text{s.t. } \xi \geq 2 \left(\sum_{s'=\hat{S}+1}^{S} (b_{\hat{S}+1} - b_{s'}) \hat{P}_{s,a,s} \right)^{-1}.$$

Differentiating the objective function, we find that it's derivative is given by:

$$-\beta + b_{\hat{S}+1} - \sum_{s'=\hat{S}+1}^{S} \hat{P}_{s,a,s'} \left(b_{\hat{S}+1} - b_{s'} + \frac{1}{2}(b_{\hat{S}+1} - b_{s'})(-\xi b_{s'} + \xi b_{\hat{S}+1} - 2) \right).$$

which can be written as:

$$-\beta + b_{\hat{S}+1} - \sum_{s'=\hat{S}+1}^{S} \hat{P}_{s,a,s'} \xi \left(b_{\hat{S}+1} - b_{s'} \right)^2.$$

which means the globally optimal ξ is given by:

$$\xi_1^* = \frac{-\beta + b_{\hat{S}+1}}{\sum_{s'=\hat{S}+1}^{S} \hat{P}_{s,a,s'} \left(b_{\hat{S}+1} - b_{s'} \right)^2}.$$

In the second case, it is easy to see that ξ_2^* is obtained by replacing $b_{\hat{S}+1}$ with $b_{\hat{S}}$. The model is therefore:

$$\max_{\xi} \left\{ -\beta\xi + \xi b_{\hat{S}} - 2 + \sum_{s'=1}^{\hat{S}} \hat{P}_{s,a,s'} - \sum_{s'=\hat{S}+1}^{S} \hat{P}_{s,a,s'} \left((-\xi b_{s'} + \xi b_{\hat{S}} - 2) + \frac{(-\xi b_{s'} + \xi b_{\hat{S}} - 2)^2}{4} \right) \right\}$$
$$\text{s.t. } \xi \leq 2 \left(\sum_{s'=\hat{S}+1}^{S} (b_{\hat{S}} - b_{s'}) \hat{P}_{s,a,s} \right)^{-1}.$$

The corresponding globally optimal solution is given by:

$$\xi_2^* = \frac{-\beta + b_{\hat{S}}}{\sum_{s'=\hat{S}+1}^{S} \hat{P}_{s,a,s'} \left(b_{\hat{S}} - b_{s'}\right)^2}.$$

In the final case, we note that:

$$H'(\xi) = \frac{\sum_{s'=\hat{S}+1}^{S} b_{s'} \hat{P}_{s,a,s'}}{\sum_{s'=\hat{S}+1}^{S} \hat{P}_{s,a,s'}}.$$

The model then becomes:

$$\max_{\xi} \left\{ -\beta\xi + H(\xi) + \sum_{s'=1}^{\hat{S}} \hat{P}_{s,a,s'} - \sum_{s'=\hat{S}+1}^{S} \hat{P}_{s,a,s'} \left((-\xi b_{s'} + H(\xi)) + \frac{(-\xi b_{s'} + H(\xi))^2}{4} \right) \right\} \tag{C.2.9}$$

$$\text{s.t. } \xi \leq 2 \left(\sum_{s'=\hat{S}+1}^{S} (b_{\hat{S}+1} - b_{s'}) \hat{P}_{s,a,s} \right)^{-1} \tag{C.2.10}$$

$$\xi \geq 2 \left(\sum_{s'=\hat{S}+1}^{S} (b_{\hat{S}} - b_{s'}) \hat{P}_{s,a,s} \right)^{-1} \tag{C.2.11}$$

The derivative of the objective is given by:

$$-\beta + H'(\xi) - \sum_{s'=\hat{S}+1}^{S} \hat{P}_{s,a,s'} \left(H'(\xi) - b_{s'} + \frac{1}{2}(H'(\xi) - b_{s'})(-\xi b_{s'} + H(\xi)) \right).$$

From the same steps as for the first case, this leads to:

$$\xi_3^* = \frac{-\beta + H'(\xi)}{\sum_{s'=\hat{S}+1}^{S} \hat{P}_{s,a,s'} \left(H'(\xi) - b_{s'}\right)^2}.$$

Then solving the problem in each case corresponds to checking if the optimal ξ lies within the allowed range, and selecting one of the bounds if it does not.

C.2.2 Reformulation of Projection Problem

As shown by Ho et al. (2022), a general projection problem can be reformulated as:

$$\begin{aligned}
&\max_{\psi,\xi} -\beta\xi + \psi - \sum_{s'\in\mathcal{S}} \hat{P}_{s,a,s'}\phi^*(-\xi b_{s'} + \psi) \\
&\text{s.t. } \xi \in \mathbb{R}_+, \psi \in \mathbb{R}.
\end{aligned}$$

For the modified χ^2-divergence, we have $\phi^*(z) = \max\left\{1 + \frac{z}{2}, 0\right\}^2 - 1$, or equivalently $\phi^*(z) = \frac{1}{4}\max\left\{z+2, 0\right\}^2 - 1$. Hence, we can represent $\phi^*(-\xi b_{s'} + \psi)$ via:

$$\begin{aligned}
&\zeta_{s'} \geq -\xi b_{s'} + \psi + 2 \ \forall \ s' \in \mathcal{S} \\
&\zeta_{s'} \geq 0 \ \forall \ s' \in \mathcal{S} \\
&u_{s'} \geq \frac{1}{4}\zeta_{s'}^2 \ \forall \ s' \in \mathcal{S}.
\end{aligned}$$

Then, the model becomes:

$$\begin{aligned}
&\max_{\xi,\psi,\boldsymbol{\zeta},\boldsymbol{u}} -\beta\xi + \psi - \sum_{s'\in\mathcal{S}} \hat{P}_{s,a,s'}(u_{s'} - 1) \\
&\text{s.t. } \zeta_{s'} \geq -\xi b_{s'} + \psi + 2 \ \forall \ s' \in \mathcal{S} \\
&\quad u_{s'} \geq \frac{1}{4}\zeta_{s'}^2 \ \forall \ s' \in \mathcal{S} \\
&\quad \zeta_{s'} \geq 0 \ \forall \ s' \in \mathcal{S} \\
&\quad \xi \in \mathbb{R}_+, \psi \in \mathbb{R}.
\end{aligned}$$

C.3 A Newsvendor Model Incorporating Backorder Costs

Suppose that action a is taken when in state s and assume that b' now represents a backorder cost per unit of unmet demand. For a given realisation x of the demand random variable $X_{s,a}$, we define the one-period reward incorporating backorder costs as:

$$r'_{s,a,x} = c\min\{x, \bar{s}\} - wa - h(\bar{s} - \min\{x, \bar{s}\}) - b'\max\{x - \bar{s}, 0\}.$$

In addition, let $\boldsymbol{P}'_{s,a} = \left(P'_{s,a,x}\right)_{x\in\mathcal{X}_{s,a}}$ represent a (non-parametric) candidate for the distribution of $X_{s,a}$. We can then formulate the non-parametric robust Bellman update as:

$$v_s^{n+1} = \max_{\boldsymbol{\pi}_s\in\Delta_A} \min_{\boldsymbol{P}'_s\in\mathcal{P}'_s} \sum_{a\in\mathcal{A}} \pi_{s,a} \sum_{x\in\mathcal{X}_{s,a}} P'_{s,a,x}\left(r'_{s,a,x} + \gamma v^n_{g(x|s,a)}\right) \ \forall\ s\in\mathcal{S}, \tag{C.3.1}$$

where $\mathcal{P}'_s$ is an ambiguity set for the true distribution of $X_{s,a}$ (not the true transition distribution). This set can be defined using ϕ-divergences as follows:

$$\mathcal{P}'_s = \left\{ \boldsymbol{P}'_s \in \Delta_{|\mathcal{X}_{s,1}|} \times \ldots \times \Delta_{|\mathcal{X}_{s,A}|} : \sum_{a\in\mathcal{A}} d_a(\boldsymbol{P}'_{s,a}, \hat{\boldsymbol{P}}'_{s,a}) \le \kappa \right\},$$

where $\hat{\boldsymbol{P}}'_{s,a} = \left(f_{X_{s,a}}(x|\hat{\boldsymbol{\theta}})\right)_{x\in\mathcal{X}_{s,a}}$, for example. Similarly, we can formulate the parametric update problem as:

$$v_s^{n+1} = \max_{\boldsymbol{\pi}_s\in\Delta_A} \min_{\boldsymbol{\theta}_s\in\Theta_s} \sum_{a\in\mathcal{A}} \pi_{s,a} \sum_{x\in\mathcal{X}_{s,a}} f_{X_{s,a}}(x|\boldsymbol{\theta}_{s,a})\left(r'_{s,a,x} + \gamma v^n_{g(x|s,a)}\right) \ \forall\ s\in\mathcal{S}.$$

In these formulations, we could simplify the terms relating to backorder costs as follows:

$$\sum_{x\in\mathcal{X}_{s,a}} P'_{s,a,x}\max\{x - \bar{s}, 0\} = \sum_{x=\bar{s}+1}^{|\mathcal{X}_{s,a}|} P'_{s,a,x}(x - \bar{s}) \tag{C.3.2}$$

$$\sum_{x\in\mathcal{X}_{s,a}} f_{X_{s,a}}(x|\boldsymbol{\theta}_{s,a}) \max\{x-\bar{s},0\} = \sum_{x=\bar{s}+1}^{|\mathcal{X}_{s,a}|} f_{X_{s,a}}(x|\boldsymbol{\theta}_{s,a})(x-\bar{s}). \tag{C.3.3}$$

If we have infinite support demands, i.e. $|\mathcal{X}_{s,a}| = \infty$, then this implies that an infinite number of decision variables are required for the non-parametric model. This means that a completely different treatment is required. In many cases, however, the parametric expression can be further simplified. For example, if the demand random variable is $X_{s,a} \sim \text{Pois}(\lambda_{s,a})$, then we have:

$$\begin{aligned}
\sum_{x=\bar{s}}^{|\mathcal{X}_{s,a}|} f_{X_{s,a}}(x|\boldsymbol{\theta}_{s,a})(x-\bar{s}) &= \sum_{x=\bar{s}+1}^{\infty} \frac{\lambda_{s,a}^x \exp(-\lambda_{s,a})}{x!}(x-\bar{s}) \\
&= \lambda_{s,a} \sum_{x=\bar{s}+1}^{\infty} \frac{\lambda_{s,a}^{x-1} \exp(-\lambda_{s,a})}{(x-1)!} - \bar{s}\left(1 - \sum_{x=0}^{\bar{s}} \frac{\lambda_{s,a}^x \exp(-\lambda_{s,a})}{x!}\right) \\
&= \lambda_{s,a} \sum_{x=\bar{s}}^{\infty} \frac{\lambda_{s,a}^x \exp(-\lambda_{s,a})}{x!} - \bar{s}\left(1 - F_{X_{s,a}}(\bar{s}|\lambda_{s,a})\right) \\
&= \lambda_{s,a}\left(1 - F_{X_{s,a}}(\bar{s}-1|\lambda_{s,a})\right) - \bar{s}\left(1 - F_{X_{s,a}}(\bar{s}|\lambda_{s,a})\right),
\end{aligned}$$

which only involves finite sums. Without incorporating further information on the true distribution of $X_{s,a}$ such as its moments, the expression in (C.3.2) cannot be simplified further. The infinite number of variables required means that the algorithms in this paper are not applicable to the robust Bellman update problem in (C.3.1).

Bibliography

Abdel-Malek, L. L. and Montanari, R. (2005). An analysis of the multi-product newsboy problem with a budget constraint. *International Journal of Production Economics*, 97(3):296–307.

Agrawal, N. and Smith, S. A. (1996). Estimating negative binomial demand for retail inventory management with unobservable lost sales. *Naval Research Logistics (NRL)*, 43(6):839–861.

Ahipasaoglu, S. D., Natarajan, K., and Shi, D. (2019). Distributionally robust project crashing with partial or no correlation information. *Networks*, 74(1):79–106.

Ahmed, S., Çakmak, U., and Shapiro, A. (2007). Coherent risk measures in inventory problems. *European Journal of Operational Research*, 182(1):226–238.

Ainslie, R., McCall, J., Shakya, S., and Owusu, G. (2017). Predicting service levels using neural networks. In *International Conference on Innovative Techniques and Applications of Artificial Intelligence*, pages 411–416. Springer.

Ainslie, R., McCall, J., Shakya, S., and Owusu, G. (2018). Tactical plan optimisation

for large multi-skilled workforces using a bi-level model. In *2018 IEEE Congress on Evolutionary Computation (CEC)*, pages 1–8. IEEE.

Ainslie, R., Shakya, S., McCall, J., and Owusu, G. (2015). Optimising skill matching in the service industry for large multi-skilled workforces. In *International Conference on Innovative Techniques and Applications of Artificial Intelligence*, pages 231–243. Springer.

Alfares, H. K. and Elmorra, H. H. (2005). The distribution-free newsboy problem: Extensions to the shortage penalty case. *International Journal of Production Economics*, 93(1):465–477.

Altintas, N., Erhun, F., and Tayur, S. (2008). Quantity discounts under demand uncertainty. *Management Science*, 54(4):777–792.

Angalakudati, M., Balwani, S., Calzada, J., Chatterjee, B., Perakis, G., Raad, N., and Uichanco, J. (2014). Business analytics for flexible resource allocation under random emergencies. *Management Science*, 60(6):1552–1573.

Arrow, K. J., Harris, T., and Marschak, J. (1951). Optimal inventory policy. *Econometrica*, 19(3):250–272.

Arrow, K. J., Karlin, S., and Scarf, H. E. (1958). Studies in the mathematical theory of inventory and production. *The Mathematical Gazette*, 44(348):156–156.

Bagnell, J. A., Ng, A. Y., and Schneider, J. G. (2001). Solving uncertain Markov decision processes. *Technical Report*.

Bai, Q. and Chen, M. (2016). The distributionally robust newsvendor problem with dual sourcing under carbon tax and cap-and-trade regulations. *Computers & Industrial Engineering*, 98(1):260–274.

Bansal, M., Huang, K.-L., and Mehrotra, S. (2018). Decomposition algorithms for two-stage distributionally robust mixed binary programs. *SIAM Journal on Optimization*, 28(3):2360–2383.

Barr, D. R. and Sherrill, E. T. (1999). Mean and variance of truncated normal distributions. *The American Statistician*, 53(4):357–361.

Bastian, N. D., Lunday, B. J., Fisher, C. B., and Hall, A. O. (2020). Models and methods for workforce planning under uncertainty: Optimizing U.S. army cyber branch readiness and manning. *Omega*, 92. Article 102171. Online publication.

Bayraksan, G. and Love, D. K. (2015). Data-driven stochastic programming using phi-divergences. In *The operations research revolution*, pages 1–19. INFORMS.

Behzadian, B., Petrik, M., and Ho, C. P. (2021). Fast algorithms for L_∞-constrained s-rectangular robust MDPs. In *Advances in Neural Information Processing Systems*, volume 34, pages 25982–25992. Curran Associates, Inc.

Bellman, R. (1957). A Markovian decision process. *Journal of Mathematics and Mechanics*, 6(5):679–684.

Bellman, R. (1966). Dynamic programming. *Science*, 153(3731):34–37.

Bellman, R., Glicksberg, I., and Gross, O. (1955). On the optimal inventory equation. *Management Science*, 2(1):83–104.

Ben-Tal, A., den Hertog, D., De Waegenaere, A., Melenberg, B., and Rennen, G. (2013). Robust solutions of optimization problems affected by uncertain probabilities. *Management Science*, 59(2):341–357.

Ben-Tal, A. and Nemirovski, A. (1998). Robust convex optimization. *Mathematics of Operations Research*, 23(4):769–805.

Benders, J. (1962). Partitioning procedures for solving mixed-variables programming problems. *Numerische mathematik*, 4(1):238–252.

Bensoussan, A., Çakanyildirim, M., Royal, A., and Sethi, S. (2009). Bayesian and adaptive controls for a newsvendor facing exponential demand. *Risk and Decision Analysis*, 1(4):197–210.

Bensoussan, A., Çakanyildirim, M., and Sethi, S. P. (2007). A multiperiod newsvendor problem with partially observed demand. *Mathematics of Operations Research*, 32(2):322–344.

Bertsimas, D., Natarajan, K., and Teo, C.-P. (2004). Probabilistic combinatorial optimization: Moments, semidefinite programming, and asymptotic bounds. *SIAM Journal on Optimization*, 15(1):185–209.

Black, B., Ainslie, R., Dokka, T., and Kirkbride, C. (2022a). Distributionally robust resource planning under binomial demand intakes. *European Journal of Operational Research*. Advance online publication.

Black, B., Dokka, T., and Kirkbride, C. (2022b). Robust Markov decision processes under parametric transition distributions. *arXiv preprint: arXiv.2211.07488*.

Bouakiz, M. and Sobel, M. J. (1992). Inventory control with an exponential utility criterion. *Operations Research*, 40(3):603–608.

Breton, M. L. and Hachem, S. E. (1995). Algorithms for the solution of stochastic dynamic minimax problems. *Computational Optimization and Applications*, 4(4):317–345.

Chen, D., Qi, J., Meng, F., Ang, J., Chu, S., and Sim, M. (2015). A robust optimization model for managing elective admission in hospital. *Operations Research*, 63(6):1452–1467.

Chen, L. (2010). Bounds and heuristics for optimal Bayesian inventory control with unobserved lost sales. *Operations Research*, 58(2):396–413.

Chen, L., Ma, W., Natarajan, K., Simchi-Levi, D., and Yan, Z. (2022). Distributionally robust linear and discrete optimization with marginals. *Operations Research*, 70(3):1822–1834.

Chen, X. A., Wang, Z., and Yuan, H. (2017). Optimal pricing for selling to a static multi-period newsvendor. *Operations Research Letters*, 45(5):415–420.

Collins, R. A. (2004). The behavior of the risk-averse newsvendor for uniform, truncated normal, negative binomial and gamma distributions of demand. In *Department of Operations and Management Information Systems, Leavey School of Business Santa Clara University Working paper*. Citeseer.

Conn, A. R., Gould, N. I. M., and Toint, P. L. (2000). *Trust Region Methods*. Society for Industrial and Applied Mathematics.

Dantzig, G. B. (1960). Inductive proof of the simplex method. *IBM Journal of Research and Development*, 4(5):505–506.

Deng, T., Shen, Z.-J. M., and Shanthikumar, J. G. (2014). Statistical learning of service-dependent demand in a multiperiod newsvendor setting. *Operations Research*, 62(5):1064–1076.

Derman, E., Geist, M., and Mannor, S. (2021). Twice regularized MDPs and the equivalence between robustness and regularization. In *Advances in Neural Information Processing Systems*, volume 34, pages 22274–22287. NIPS.

Djelassi, H., Mitsos, A., and Stein, O. (2021). Recent advances in nonconvex semi-infinite programming: Applications and algorithms. *EURO Journal on Computational Optimization*, 9:100006.

Dolgui, A. and Pashkevich, M. (2008). On the performance of binomial and beta-binomial models of demand forecasting for multiple slow-moving inventory items. *Computers & Operations Research*, 35(3):893–905.

Duchi, J. C., Glynn, P. W., and Namkoong, H. (2021). Statistics of robust optimization: A generalized empirical likelihood approach. *Mathematics of Operations Research*, 46(3):946–969.

Fetter, R. B. (1961). A linear programming model for long range capacity planning. *Management Science*, 7(4):372–378.

Floudas, C., Aggarwal, A., and Ciric, A. (1989). Global optimum search for nonconvex nlp and minlp problems. *Computers & Chemical Engineering*, 13(10):1117–1132.

Gallego, G., Katircioglu, K., and Ramachandran, B. (2007). Inventory management under highly uncertain demand. *Operations Research Letters*, 35(3):281–289.

Gallego, G. and Moon, I. (1993). The distribution free newsboy problem: Review and extensions. *The Journal of the Operational Research Society*, 44(8):825–834.

Gao, R. and Kleywegt, A. J. (2017a). Distributionally robust stochastic optimization with dependence structure. *ArXiv Preprint: 1701.04200*.

Gao, R. and Kleywegt, A. J. (2017b). Distributionally robust stochastic optimization with dependence structure. *arXiv preprint. arXiv.1701.04200*.

Ghaoui, L. E., Oks, M., and Oustry, F. (2003). Worst-case value-at-risk and robust portfolio optimization: A conic programming approach. *Operations Research*, 51(4):543–556.

Givan, R., Leach, S., and Dean, T. (2000). Bounded-parameter Markov decision processes. *Artificial Intelligence*, 122(1):71–109.

Goyal, V. and Grand-Clement, J. (2022). Robust Markov decision processes: Beyond rectangularity. *Mathematics of Operations Research*. Advance online publication.

Grand-Clément, J. and Kroer, C. (2021). Scalable first-order methods for robust MDPs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 12086–12094.

Grothey, A. (2001). *Decomposition methods for nonlinear nonconvex optimization problems*. PhD thesis, Citeseer.

Gurobi Optimization, LLC (2022). Gurobi Optimizer Reference Manual.

Han, Q., Du, D., and Zuluaga, L. F. (2014). Technical note—a risk- and ambiguity-averse extension of the max-min newsvendor order formula. *Operations Research*, 62(3):535–542.

Hanasusanto, G. A. and Kuhn, D. (2013). Robust data-driven dynamic programming. In *Advances in Neural Information Processing Systems*, volume 26, page 827–835. Curran Associates, Inc.

Hanssmann, F. and Hess, S. W. (1960). A linear programming approach to production and employment scheduling. *Management Technology*, 1(1):46–51.

Hettich, R. and Kortanek, K. O. (1993). Semi-infinite programming: Theory, methods, and applications. *SIAM Review*, 35(3):380–429.

Hill, R. M. (1997). Applying Bayesian methodology with a uniform prior to the single period inventory model. *European Journal of Operational Research*, 98(3):555–562.

Ho, C. P., Petrik, M., and Wiesemann, W. (2021). Partial policy iteration for L_1-robust Markov decision processes. *Journal of Machine Learning Research*, 22(275):1–46.

Ho, C. P., Petrik, M., and Wiesemann, W. (2022). Robust phi-divergence MDPs. *arXiv preprint arXiv:2205.14202*.

Holt, C. C., Modigliani, F., and Simon, H. A. (1955). A linear decision rule for production and employment scheduling. *Management Science*, 2(1):1–30.

Holte, M. and Mannino, C. (2013). The implementor/adversary algorithm for the cyclic and robust scheduling problem in health-care. *European Journal of Operational Research*, 226(3):551–559.

Hu, Z. and Hong, L. J. (2013). Kullback-Leibler divergence constrained distributionally robust optimization. *Available at Optimization Online: http://www.optimization-online.org/DB˙HTML/2012/11/3677.html*.

Hu, Z., Hong, L. J., and So, A. M.-C. (2013). Ambiguous probabilistic programs. *Available at Optimization Online: http://www.optimization-online.org/DB˙HTML/2013/09/4039.html*.

Hulst, D., den Hertog, D., and Nuijten, W. (2017). Robust shift generation in workforce planning. *Computational Management Science*, 14(1):115–134.

Iyengar, G. N. (2005). Robust dynamic programming. *Mathematics of Operations Research*, 30(2):257–280.

J. Abernathy, W., Baloff, N., Hershey, J., and Wandel, S. (1973). A three-stage manpower planning and scheduling model – a service-sector example. *Operations Research*, 21(3):693–711.

Jiang, R. and Guan, Y. (2016). Data-driven chance constrained stochastic program. *Math. Program.*, 158(1–2):291–327.

Kall, P., Wallace, S. W., and Kall, P. (1994). *Stochastic Programming*, volume 6. Springer.

Kim, G., Wu, K., and Huang, E. (2015). Optimal inventory control in a multi-period newsvendor problem with non-stationary demand. *Advanced Engineering Informatics*, 29(1):139–145.

Kogan, K. and Lou, S. (2003). Multi-stage newsboy problem: A dynamic model. *European Journal of Operational Research*, 149(2):448–458. Sequencing and Scheduling.

Kortanek, K. O. and No, H. (1993). A central cutting plane algorithm for convex semi-infinite programming problems. *SIAM Journal on Optimization*, 3(4):901–918.

Kraft, D. (1988). *A software package for sequential quadratic programming*. Technical Report. DFVLR-FB 88-28, DLR German Aerospace Center – Institute for Flight Mechanics, Koln, Germany.

Kuhn, H. W. and Tucker, A. W. (1951). Nonlinear programming. In *Traces and emergence of nonlinear programming*, pages 247–258. Springer.

Lam, H. (2019). Recovering best statistical guarantees via the empirical divergence-based distributionally robust optimization. *Operations Research*, 67(4):1090–1105.

Land, A. H. and Doig, A. G. (1960). An automatic method of solving discrete programming problems. *Econometrica*, 28(3):497–520.

Lau, H.-S. and Lau, A. H. L. (1996). The newsstand problem: A capacitated multiple-product single-period inventory problem. *European Journal of Operational Research*, 94(1):29–42.

Le Tallec, Y. (2007). *Robust, risk-sensitive, and data-driven control of Markov decision processes*. PhD thesis, Massachusetts Institute of Technology.

Lee, C. and Mehrotra, S. (2015). A distributionally-robust approach for finding support vector machines. *Available from Optimization Online: https://optimization-online.org/2015/06/4965/*.

Lee, C.-M. and Hsu, S.-L. (2011). The effect of advertising on the distribution-free newsboy problem. *International Journal of Production Economics*, 129(1):217–224.

Lee, J. and Raginsky, M. (2018). Minimax statistical learning with Wasserstein distances. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, page 2692–2701. Curran Associates Inc.

Lee, S., Kim, H., and Moon, I. (2021). A data-driven distributionally robust newsvendor model with a Wasserstein ambiguity set. *Journal of the Operational Research Society*, 72(8):1879–1897.

Levi, R., Roundy, R. O., and Shmoys, D. B. (2007). Provably near-optimal sampling-based policies for stochastic inventory control models. *Mathematics of Operations Research*, 32(4):821–839.

Levina, T., Levin, Y., McGill, J., Nediak, M., and Vovk, V. (2010). Weak aggre-

gating algorithm for the distribution-free perishable inventory problem. *Operations Research Letters*, 38(6):516–521.

Liao, S., van Delft, C., and Vial, J.-P. (2013). Distributionally robust workforce scheduling in call centres with uncertain arrival rates. *Optimization Methods and Software*, 28(3):501–522.

Liu, B., Holmbom, M., Segerstedt, A., and Chen, W. (2015). Effects of carbon emission regulations on remanufacturing decisions with limited information of demand distribution. *International Journal of Production Research*, 53(2):532–548.

Liyanage, L. H. and Shanthikumar, J. (2005). A practical inventory control policy using operational statistics. *Operations Research Letters*, 33(4):341–348.

Lotfi, S. and Zenios, S. A. (2018). Robust VaR and CVaR optimization under joint ambiguity in distributions, means, and covariances. *European Journal of Operational Research*, 269(2):556–576.

Luo, F. and Mehrotra, S. (2019). Decomposition algorithm for distributionally robust optimization using Wasserstein metric with an application to a class of regression models. *European Journal of Operational Research*, 278(1):20–35.

López, M. and Still, G. (2007). Semi-infinite programming. *European Journal of Operational Research*, 180(2):491–518.

Mannor, S., Simester, D., Sun, P., and Tsitsiklis, J. N. (2007). Bias and variance approximation in value function estimates. *Management Science*, 53(2):308–322.

Martel, A. and Price, W. (1978). A normative model for manpower planning under risk. In *Manpower Planning and Organization Design*, pages 291–305. Springer.

Matsuyama, K. (2006). The multi-period newsboy problem. *European Journal of Operational Research*, 171(1):170–188.

Mehrotra, S. and Papp, D. (2014). A cutting surface algorithm for semi-infinite convex programming with an application to moment robust optimization. *arXiv preprint. arXiv:1306.3437*.

Mehrotra, S. and Zhang, H. (2013). Models and algorithms for distributionally robust least squares problems. *Mathematical Programming*, 146(1):123–141.

Meilijson, I. and Nádas, A. (1979). Convex majorization with an application to the length of critical paths. *Journal of Applied Probability*, 16(3):671–677.

Mersereau, A. J. (2015). Demand estimation from censored observations with inventory record inaccuracy. *Manufacturing & Service Operations Management*, 17(3):335–349.

Millar, R. B. (2011). *Maximum Likelihood Estimation and Inference: With Examples in R, SAS and ADMB*, volume 112. Wiley, New York.

Mohajerin Esfahani, P. and Kuhn, D. (2018). Data-driven distributionally robust optimization using the Wasserstein metric: performance guarantees and tractable reformulations. *Mathematical Programming*, 171(1–2):115–166.

Moon, I. and Choi, S. (1995). The distribution free newsboy problem with balking. *The Journal of the Operational Research Society*, 46(4):537–542.

Nahmias, S. (1994). Demand estimation in lost sales inventory systems. *Naval Research Logistics*, 41(6):739 – 757.

Natarajan, K., Song, M., and Teo, C.-P. (2009). Persistency model and its applications in choice modeling. *Management Science*, 55(3):453–469.

Natarajan, K., Teo, C. P., and Zheng, Z. (2011). Mixed 0-1 linear programs under objective uncertainty: A completely positive representation. *Operations Research*, 59(3):713–728.

Nilim, A. and El Ghaoui, L. (2005). Robust control of Markov decision processes with uncertain transition matrices. *Operations Research*, 53(5):780–798.

Ouyang, L.-Y. and Chang, H.-C. (2002). A minimax distribution free procedure for mixed inventory models involving variable lead time with fuzzy lost sales. *International Journal of Production Economics*, 76(1):1–12.

Pflug, G. and Wozabal, D. (2007). Ambiguity in portfolio selection. *Quantitative Finance*, 7(4):435–442.

Potra, F. A. and Wright, S. J. (2000). Interior-point methods. *Journal of Computational and Applied Mathematics*, 124(1):281–302. Numerical Analysis 2000. Vol. IV: Optimization and Nonlinear Equations.

Powell, W. B. (2007). *Approximate Dynamic Programming: Solving the Curses of Dimensionality*. John Wiley & Sons.

Rahimian, H., Bayraksan, G., and Homem-De-Mello, T. (2019). Identifying effective scenarios in distributionally robust stochastic programs with total variation distance. *Mathematical Programming*, 173(1–2):393–430.

Rahimian, H. and Mehrotra, S. (2019). Distributionally robust optimization: A review. *arXiv preprint. arXiv:1908.05659*.

Ross, E. (2016). *Cross-trained Workforce Planning Models*. PhD thesis, University of Lancaster.

Rossi, R., Prestwich, S., Tarim, S. A., and Hnich, B. (2014). Confidence-based optimisation for the newsvendor problem under binomial, Poisson and exponential demand. *European Journal of Operational Research*, 239(3):674–684.

Samudra, M., Van Riet, C., Demeulemeester, E., Cardoen, B., Vansteenkiste, N., and Rademakers, F. E. (2016). Scheduling operating rooms: achievements, challenges and pitfalls. *Journal of Scheduling*, 19(5):493–525.

Satia, J. K. and Lave, R. E. (1973). Markovian decision processes with uncertain transition probabilities. *Operations Research*, 21(3):728–740.

Scarf, H. E. (1957). *A Min-Max Solution of an Inventory Problem*. RAND Corporation, Santa Monica, CA.

Shapiro, A. and Kleywegt, A. (2002). Minimax analysis of stochastic problems. *Optimization Methods and Software*, 17(3):523–542.

Siegel, A. F. and Wagner, M. R. (2021). Profit estimation error in the newsvendor model under a parametric demand distribution. *Management Science*, 67(8):4863–4879.

Tirinzoni, A., Petrik, M., Chen, X., and Ziebart, B. (2018). Policy-conditioned uncertainty sets for robust Markov decision processes. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc.

Ullah, M., Khan, I., and Sarkar, B. (2019). Dynamic pricing in a multi-period newsvendor under stochastic price-dependent demand. *Mathematics*, 7(6). Article 570. Online publication.

Virtanen, P., Gommers, R., Oliphant, T. E., and SciPy 1.0 Contributors (2020). SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17(3):261–272.

Wang, C., Gao, R., Qiu, F., Wang, J., and Xin, L. (2018). Risk-based distributionally robust optimal power flow with dynamic line rating. *IEEE Transactions on Power Systems*, 33(6):6074–6086.

Wang, Z., Glynn, P. W., and Ye, Y. (2016). Likelihood robust optimization for data-driven problems. *Computational Management Science*, 13(2):241–261.

Wiesemann, W., Kuhn, D., and Rustem, B. (2013). Robust Markov decision processes. *Mathematics of Operations Research*, 38(1):153–183.

Yanıkoğlu, I. and den Hertog, D. (2013). Safe approximations of ambiguous chance constraints using historical data. *INFORMS Journal on Computing*, 25(4):666–681.

Zhang, Y. and Yang, X. (2016). Online ordering policies for a two-product, multi-period stationary newsvendor problem. *Computers & Operations Research*, 74(1):143–151.

Zhang, Y., Yang, X., and Li, B. (2017). Distribution-free solutions to the extended multi-period newsboy problem. *Journal of Industrial and Management Optimization*, 13(2):633–647.

Zhao, C. and Guan, Y. (2015). Data-driven risk-averse two-stage stochastic program with ζ-structure probability metrics. *Technical Report. Available on Optimization Online*, 2(5):1–40.

Zhu, X. and Sherali, H. D. (2009). Two-stage workforce planning under demand fluctuations and uncertainty. *Journal of the Operational Research Society*, 60(1):94–103.