

# ABSTRACT

BALAN, SRINIVASAN. Exact and Approximate Algorithms for Stochastic Production Inventory Problems Subject to Congestion. (Under the direction of Dr. Reha Uzsoy).

Multi-period capacitated production inventory problems are computationally challenging. These problems have been researched extensively, and various approximations to solve them have been proposed. In this thesis, unlike classic inventory models, we use a concave, monotonically non-decreasing clearing function (CF) to characterize the replenishment process. *Clearing functions* are meta-models that capture the queuing behavior and relationship between the workload and the production in the capacitated production system. The dissertation begins with an introduction and problem statement in Chapter 1. We review the previous related work in Chapter 2. In Chapter 3, we study the single-period, zero lead time, single-item problem (SPP) and propose an exact algorithm to find the optimal planned releases given the initial finished goods (FG) and work-in-progress (WIP) inventory levels. We use Karush Kuhn Tucker (KKT) conditions to derive the optimality conditions for the SPP. We compare the results with the uncapacitated and capacitated production inventory problem without congestion, and as a special case, we discuss the results with an uncapacitated Graves CF. Solving the single-period problem has practical benefits since its solution is easier to compute and helpful in understanding the structure of a multi-period solution.

Chapter 4 proposes two stochastic dynamic programming (SDP) algorithms to solve the multi-period production inventory problem in a finite horizon setting using releases ( $R$ ) and production ( $P$ ) as actions. Since the recursive relation does not preserve the discretization of the state space, we use bilinear interpolation to estimate the expected cost-to-go function from the look-up table. We show that the objective function is convex in  $R$  and  $P$ , and exploiting the structure of the problem, we develop a binary search procedure to accelerate the SDP algorithm. In addition, we deploy a parallel computing framework to speed up the computing using multiple cores. Through numerical experiments, we study the effect of binary search and parallel computing with the original SDP algorithm, the best grid size for state space, and action space to yield a benchmark cost. We discuss possible extensions with other approximate dynamic programming methods and DP-based decomposition methods for multi-item problems in Chapter 5. Finally, we present a summary of our analysis and critical findings, followed by future directions.

© Copyright 2023 by Srinivasan Balan

All Rights Reserved

Exact and Approximate Algorithms for Stochastic  
Production Inventory Problems Subject to Congestion

by  
Srinivasan Balan

A dissertation submitted to the Graduate Faculty of  
North Carolina State University  
in partial fulfillment of the  
requirements for the Degree of  
Doctor of Philosophy

Industrial Engineering

Raleigh, North Carolina  
2023

APPROVED BY:

---

Dr. John Baugh

---

Dr. Ton De Kok

---

Dr. Michael Kay

---

Dr. Russell King

---

Dr. Maria Mayorga

---

Dr. Reha Uzsoy  
Chair of Advisory Committee

## DEDICATION

To my parents who encouraged me during difficult times and motivated for the successful completion of this dissertation.

## BIOGRAPHY

Srinivasan Balan was born in Chennai, Tamilnadu, India. He received his Bachelor of Engineering degree in Industrial Engineering from the College of Engineering, Guindy, Anna University, in 2007. He then did his Master's in Industrial Engineering (specializing in Operations Management) from the National Institute of Industrial Engineering, Mumbai, India, in 2012 and was honored with two gold medals for academic excellence. He joined the Edward P. Fitts Department of Industrial and Systems Engineering at North Carolina State University to pursue his Ph.D. degree in Industrial Engineering and a minor in Operations Research under the guidance of Dr. Reha Uzsoy in 2017. In addition, he received a Master's in Operations Research degree from the same institute in 2022. Prior to NC State, he worked in the industry for eight years in the automotive, manufacturing, healthcare, agribusiness, and consulting sectors.

His last work experience at ITC Infotech and Ernst and Young gave him ample supply chain consulting experience in the areas of Supply Chain Network Design, Inventory Optimization, and Vehicle Routing Problems. He held research assistant positions with Dr. Reha Uzsoy from Edward P Fitts Department of Industrial and Systems Engineering and Dr. Robert Handfield from the Poole College of Management, Supply Chain Resource Cooperative at NC State. His research interests are in Stochastic Production Inventory problems, Stochastic Optimization methods, and Supply Chain engineering focusing on applications of Operations Research in real-world problems. Besides research, he worked as an Analytics and Project intern (2019 Summer) at UPS and as a Data Science graduate intern (Summer and Fall 2021) at Intel Corporation. While at NC State, he served in several leadership positions across various organizations, including President of the IISE LSC division, Staff Editor at the OR/MS Tomorrow magazine, Student chapter representative of INFORMS FORA committee, and President of the INFORMS NCSU student chapter. He authored 10 refereed journal papers and international conferences and is a certified Six Sigma Green Belt. He received the Voice of the Future award and Excellence award during his work at Murugappa Group and Olam International Ltd. He has won various awards and competitions during his Ph.D. program, including 1st place in the case study competition organized by the IISE LSC division, 3rd place in the 10th AIMMS MOFTA Split Delivery Vehicle Routing problems organized by Lehigh University, and a Magna Cum Laude award as President for the INFORMS NCSU Chapter. He can be contacted at [srini.sigma@gmail.com](mailto:srini.sigma@gmail.com).

## ACKNOWLEDGEMENTS

This dissertation would not have been possible without the support of my parents, Mr. C. S. Balan, and Mrs. Geetha, for their unconditional love and confidence in me.

First and foremost, I would like to express my sincere gratitude to my advisor, Dr. Reha Uzsoy. His willingness and guidance helped me to learn new aspects of theory and the application of algorithms, which led to this dissertation. I thank him for his valuable amount of knowledge sharing, countless emails, and funding support during and after COVID-19. I want to thank my committee members for devoting time and sharing valuable advice throughout the research process. Special thanks to Dr. Michael Kay and Dr. Maria Mayorga for helping me with the stochastic dynamic program and its acceleration procedure. I am thankful to Dr. Ton De Kok, for his support, and constructive feedback that brought the dissertation to its present shape. I thank Dr. John Baugh for accepting to be on my committee as a GSA representative. I am grateful to Dr. Russell King, Director of Graduate Programs, ISE department, who also served on my committee and helped me sail through this Ph.D. journey.

I have been fortunate to work with Dr. Robert Handfield from the Supply Chain Resource Cooperative, Poole College of Management. His support throughout the first two and a half years of my Ph.D. program is highly appreciated. During my time at the SCRC, I had the privilege to work with him on various consulting engagements, and I am grateful for his time, motivation, encouragement, and funding support in the first two and a half years of the Ph.D. program.

I am deeply grateful to all the faculty members of NC State for their support and encouragement. Special mention to Dr. Thom Hodgson, Dr. Michael Kay, Dr. Julie Ivy, Dr. Yuan Shin Lee, Dr. Dan Harris, and Dr. John Baugh. I truly enjoyed their classes and interactions. I would also like to thank every non-teaching staff in the ISE department for finding time to help with anything I need. Last but definitely not least, I would like to thank my friends, without whom this Ph.D. journey would not have been possible. I am thankful to a long list of people I have encountered throughout my time at NC State University, while collaborating with the IISE LSC division, INFORMS OR/MS, and INFORMS NCSU chapter. Special mention to my friend Rahman for his valuable time in teaching programming gimmicks and other graduate students who always supported me and made my time in Raleigh a wonderful and fun-filled experience.

# TABLE OF CONTENTS

<b>List of Tables</b> . . . . .	<b>viii</b>
<b>List of Figures</b> . . . . .	<b>xi</b>
<b>Chapter 1 Introduction</b> . . . . .	<b>1</b>
1.1 Problem statement and objective . . . . .	1
1.2 Motivation . . . . .	2
1.3 Contributions . . . . .	3
1.4 Outline of the dissertation . . . . .	4
<b>Chapter 2 Previous Related Work</b> . . . . .	<b>5</b>
2.1 Motivation . . . . .	5
2.2 Classical Inventory Models . . . . .	5
2.2.1 Single-stage uncapacitated models . . . . .	6
2.2.2 Single-stage capacitated models . . . . .	7
2.2.3 Single stage multi-period models . . . . .	8
2.2.4 Single stage capacitated multi-period models . . . . .	9
2.2.5 Multi-stage multi-period models . . . . .	10
2.2.6 Multi-stage multi-period capacitated models . . . . .	15
2.3 Production Planning under Congestion . . . . .	16
2.3.1 Order release in the production planning . . . . .	16
2.3.2 Review of Stochastic Dynamic Programming . . . . .	24
2.3.3 Approximate Dynamic Programming . . . . .	28
2.4 Methods used in PP and Inventory models . . . . .	31
2.4.1 Chance constrained programming . . . . .	31
2.4.2 Robust optimization . . . . .	33
2.4.3 Stochastic programming . . . . .	34
2.4.4 Simulation-based approaches . . . . .	36
2.4.5 Need for a benchmark algorithm . . . . .	37
2.5 Conclusions . . . . .	38
<b>Chapter 3 Single-Stage Zero Lead time Problem</b> . . . . .	<b>40</b>
3.1 Introduction . . . . .	40
3.2 Model assumptions . . . . .	41
3.3 Convexity conditions . . . . .	45
3.4 Optimal solution structure . . . . .	46
3.5 Single period problem - optimality conditions . . . . .	51
3.5.1 Special case of Graves CF . . . . .	51
3.6 Numerical Experimentation and Design . . . . .	52
3.6.1 Using Missbauer CF . . . . .	52
3.6.2 Experimental design . . . . .	53

3.7	Experimental Results and Discussion . . . . .	55
3.7.1	Solution Characterization . . . . .	56
3.7.2	Effect of $C^p$ . . . . .	62
3.7.3	Effect of $C^r$ . . . . .	63
3.7.4	Results of Cost Configuration E3 . . . . .	66
3.8	Principal Results . . . . .	79
3.9	Conclusion . . . . .	79
<b>Chapter 4 Single-Stage Multi-Period Problem . . . . .</b>		<b>81</b>
4.1	Dynamic Programming Formulation . . . . .	81
4.2	Policy optimality . . . . .	84
4.2.1	$P$ as action . . . . .	84
4.2.2	$R$ as action . . . . .	86
4.3	Demand Discretization . . . . .	87
4.4	Stochastic Dynamic Programming Algorithm . . . . .	87
4.4.1	Stochastic Dynamic Programming using $P$ as action . . . . .	87
4.4.2	Stochastic Dynamic Programming using $R$ as action . . . . .	90
4.4.3	A binary search approach to compute optimal actions . . . . .	90
4.5	Value Function Approximation . . . . .	93
4.5.1	Truncation approximation . . . . .	93
4.5.2	Bilinear Interpolation . . . . .	94
4.6	Model Assumptions and Implementation issues . . . . .	95
4.7	Computational Experiments and Results . . . . .	95
4.7.1	Experimental Design . . . . .	95
4.7.2	Effect of Grid Size in Conventional SDP . . . . .	98
4.7.3	Effect of Binary Search in SDP . . . . .	100
4.7.4	Effect of Binary Search and Parallel Computing . . . . .	103
4.7.5	Effect of Actions: $R$ versus $P$ . . . . .	105
4.7.6	Effect of Value Function Approximations . . . . .	106
4.7.7	Effect of W State Space Reduction . . . . .	110
4.8	Principal Results . . . . .	112
4.9	Conclusion . . . . .	113
<b>Chapter 5 Conclusion and Future Directions . . . . .</b>		<b>115</b>
5.1	Summary . . . . .	115
5.2	Future Directions . . . . .	116
<b>Bibliography . . . . .</b>		<b>120</b>
<b>APPENDIX . . . . .</b>		<b>136</b>
Appendix A Acronyms . . . . .		137
A.1	Demand Discretization . . . . .	138
A.2	Original SDP results . . . . .	142
A.3	BS-based SDP results . . . . .	143

A.4	<i>R</i> as action-based SDP results . . . . .	145
A.5	Sensitivity analysis - <i>R</i> and <i>P</i> as action . . . . .	147
A.6	Using Graves CF . . . . .	149
A.6.1	Graves CF - Solution characterization . . . . .	150

## LIST OF TABLES

Table 3.1	Summary of optimal solutions using solution characterization . . .	49
Table 3.2	Optimal structure and solution characterization . . . . .	50
Table 3.3	Optimal structure and solution characterization . . . . .	50
Table 3.4	Complete experimental design for the single period problem . . . .	53
Table 3.5	Cost parameters for the experimental design . . . . .	53
Table 3.6	Results for demand $\sim \mathcal{N}(7, 1)$ . . . . .	58
Table 3.7	Comparison of UCPI solution for demand $\sim \mathcal{N}(9, 1)$ using $R$ as action with (a) $K_2=10$ (b) $K_2=30$ . . . . .	76
Table 3.8	Comparison of UCPI solution for demand $\sim \mathcal{N}(9, 1)$ using $P$ as action with (a) $K_2=10$ (b) $K_2=30$ . . . . .	76
Table 3.9	Comparison of CPI solution for demand $\sim \mathcal{N}(9, 1)$ using $R$ as action with (a) $K_2=10$ (b) $K_2=30$ . . . . .	77
Table 3.10	Comparison of CPI solution for demand $\sim \mathcal{N}(9, 1)$ using $P$ as action with (a) $K_2=10$ (b) $K_2=30$ . . . . .	78
Table 4.1	Demand patterns for an overall mean demand of 5 . . . . .	96
Table 4.2	Demand patterns for an overall mean demand of 8 . . . . .	96
Table 4.3	Parameter settings . . . . .	97
Table 4.4	Experimental design using grid size and VFA . . . . .	97
Table 4.5	Expected cost and computational times (minutes) comparison for various experiments using $P$ -based standard SDP. We show the im- pact of action space discretization $\in \{1, 0.5, 0.1\}$ and state space discretization $\in \{1, 0.5\}$ with mean demand of 5, $K_2=10$ . We use bilinear interpolation as VFA on a 36-core CPU. We denote the ex- pected cost for the grid size in state and action as (state grid, action grid) . . . . .	99
Table 4.6	A comparison of expected cost and computing times between BS- based SDP and the standard SDP methods. We use an integer and 0.5 grid on state and a 0.1 grid on an action-based solution from the standard SDP as a benchmark. We use an integer grid and 0.5 grid on state space for mean demand of 5 and $K_2 = 10$ in the BS- based SDP. We use a 36-core CPU as a computing environment with bilinear interpolation as VFA . . . . .	101

Table 4.7	A comparison of expected cost and computing times between BS-based SDP and the original SDP methods. We use an integer and 0.5 grid on state and a 0.1 grid on an action-based solution from the standard SDP as a benchmark. We use an integer grid and 0.5 grid on state space for mean demand of 5 and $K_2 = 30$ in the BS-based SDP. We use a 36-core CPU as a computing environment with bilinear interpolation as VFA. We use $W_{max} = 99$ (75% of $P_{max}$ ) as $W$ state space truncation in both the BS-based SDP methods instead of 279 units to avoid extra computing times . . . . .	102
Table 4.8	Computational times (seconds) of the 10-period numerical for different demand, SCV, and SL parameters using a Missbauer CF with $K_2 = 10$ . We use an integer grid on state space and 0.1 grid action space using the $P$ -based standard SDP algorithm. We use bilinear interpolation as the VFA . . . . .	104
Table 4.9	Expected cost of the 10-period numerical for different demand, SCV, and SL parameters using $R$ and $P$ as actions for a Missbauer CF with $K_2 = 10$ . We use an integer grid on state and action space in the standard SDP algorithm for mean demand of 5 and bilinear interpolation as VFA. Gap % is calculated as $100\%(Cost_R - Cost_P)/Cost_R$	106
Table 4.10	Comparing the expected cost and computational times (minutes) between the BILSDP and Truncation approximation using $P$ -based standard SDP algorithm. We report results for grid size (1, 1) using mean demand of 5 and $K_2 = 10$ , and for grid size (0.5, 0.1), we report the results for mean demand of 8 using $K_2 = 10$ . We use $W_{max} = 99$ covering 75% of $P_{max}$ as $W$ state space truncation. We calculate the gap% for the expected cost between the two VFAs . .	108
Table 4.11	Expected cost and computational times comparison for various experiments using $P$ -based standard SDP. We use a 36-core CPU and bilinear interpolation as VFA. We show the impact of state space discretization $\in \{1, 0.5\}$ for 0.1 grid in action space using mean demand = 5 and $K_2 = 30$ in a Missbauer CF. We compare the results of $W_{max}$ between 99 and 279 units covering 75% and 90% of maximum production capacity. We denote the expected cost for the grid size in state and action as (state grid, action grid). We calculate the gap % for the expected cost as $100\%(Cost_{99} - Cost_{279})/Cost_{99}$ . . .	111

Table 4.12	Expected cost and computational times comparison for various experiments using $P$ -based standard SDP. We use a 36-core CPU and bilinear interpolation as VFA. We show the impact of state space discretization $\in \{1, 0.5\}$ for 0.1 grid in action space using mean demand = 8 and $K_2 = 30$ in a Missbauer CF. We compare the results of $W_{max}$ between 99 and 279 units covering 75% and 90% of maximum production capacity. We denote the expected cost for the grid size in state and action as (state grid, action grid). We calculate the gap % for the expected cost as $100\%(Cost_{99}-Cost_{279})/Cost_{99}$ . . .	112
Table A.1	A summary of acronyms used in alphabetical order . . . . .	137
Table A.2	Input continuous demand distribution for 10-period problem . . . .	141
Table A.3	Discretized demand distribution for 10 period problem . . . . .	141
Table A.4	Expected cost for 0.5 grid in state and 0.1 grid in action for mean demand of 5 using $K_2 \in \{10, 30\}$ . . . . .	142
Table A.5	Expected cost for 0.5 grid in state and 0.1 grid in action for mean demand of 8 using $K_2 \in \{10, 30\}$ . . . . .	143
Table A.6	We report the expected cost and computation times (minutes) for BS-based SDP using an integer grid and 0.5 grid on state space for mean demand of 5 and $K_2 = 10$ . . . . .	144
Table A.7	We report the expected cost and computation times (minutes) for BS-based SDP using an integer grid and 0.5 grid on state space for mean demand of 5 and $K_2 = 30$ . . . . .	144
Table A.8	Computational times (seconds) of the 10-period numerical for different demand, SCV, and SL parameters using $R$ as action and a Missbauer CF of $K_1 = 10$ . . . . .	146
Table A.9	Comparison of UCPI solution for demand $\sim \mathcal{N}(9, 1)$ using $R$ as action with (a) $K_2=10$ (b) $K_2=30$ . . . . .	147
Table A.10	Comparison of UCPI solution for demand $\sim \mathcal{N}(9, 1)$ using $P$ as action with (a) $K_2=10$ (b) $K_2=30$ . . . . .	148
Table A.11	Comparison of CPI solution for demand $\sim \mathcal{N}(9, 1)$ using $R$ as action with (a) $K_2=10$ (b) $K_2=30$ . . . . .	148
Table A.12	Comparison of CPI solution for demand $\sim \mathcal{N}(9, 1)$ using $P$ as action with (a) $K_2=10$ (b) $K_2=30$ . . . . .	149
Table A.13	Results for demand $\sim \mathcal{N}(7, 1)$ . . . . .	155

## LIST OF FIGURES

Figure 2.1	A simple two-stage network defined by Lesnaia (2004) . . . . .	14
Figure 2.2	Basic clearing function . . . . .	21
Figure 3.1	Overview of a single-stage single-item production system . . . . .	41
Figure 3.2	Effect of $K_2$ for $K_1 = 10$ using Missbauer CF . . . . .	52
Figure 3.3	Optimal solution characterization for demand $\sim \mathcal{N}(7, 1)$ using $b = 30$ (a) For $K_2=10$ (b) For $K_2=30$ . . . . .	56
Figure 3.4	Optimal solution characterization for demand $\sim \mathcal{N}(9, 1)$ using $b = 30$ (a) For $K_2=10$ (b) For $K_2=30$ . . . . .	57
Figure 3.5	Optimal solution characterization for demand $\sim \mathcal{N}(9, 1)$ using $b = 6$ (a) For $K_2=10$ (b) For $K_2=30$ . . . . .	59
Figure 3.6	Optimal solution characterization for $K_1=10, K_2=10$ (a) At $U = 70\%$ Demand $\sim \mathcal{N}(7, 1)$ (b) At $U = 90\%$ Demand $\sim \mathcal{N}(9, 1)$ . . . . .	61
Figure 3.7	Comparing various models for $W_0 = 0$ using $\mathcal{N}(7, 1)$ with $C^r = 1, C^p = C^w = 0$ , and backorder cost = 6. (a) Objective function (b) Optimal production . . . . .	63
Figure 3.8	Comparing $R^*$ from various models for $W_0 = 0$ using $\mathcal{N}(7, 1)$ with $C^r = 1, C^p = C^w = 0$ , and backorder cost = 6 . . . . .	64
Figure 3.9	Expected cost of CF-based model for various WIP levels using a mean demand of 9, $b=6$ , and $K_2 = 10$ . . . . .	65
Figure 3.10	Expected cost of CF-based model for various WIP levels using a mean demand of 9, $b=6$ , and $K_2 = 30$ . . . . .	65
Figure 3.11	Comparing optimal release for various models using mean demand of 9, $b=6$ when (a) $W_0 = 0$ (b) $W_0 = 10$ . . . . .	66
Figure 3.12	Comparing optimal production for various models using mean demand of 9, $b=6$ , when (a) $W_0 \in \{0, 1, 2, 5, 10\}$ (b) $W_0 = 60$ . . . . .	66
Figure 3.13	Comparing optimal production in CF-based model across different COV values when $b = 30$ (a) CF-based model with $K_2 = 10$ (b) CF-based model with $K_2 = 30$ . . . . .	67
Figure 3.14	Comparing optimal release in CF-based model across different COV values when $b = 30$ (a) CF-based model with $K_2 = 10$ (b) CF-based model with $K_2 = 30$ . . . . .	68
Figure 3.15	Comparing expected cost in CF-based model across different COV values when $b = 30$ (a) CF-based model with $K_2 = 10$ (b) CF-based model with $K_2 = 30$ . . . . .	68
Figure 3.16	Comparing the expected cost of the UCPI, CPI, and CF-based models for various models for a given $(X_0, 0)$ using $\mathcal{N}(7, 1)$ when (a) $b/h = 20$ (b) $b/h = 100$ . . . . .	69
Figure 3.17	Comparing the optimal release of the UCPI, CPI, and CF-based models for various models for a given $(X_0, 0)$ using $\mathcal{N}(7, 1)$ when (a) $b/h = 20$ (b) $b/h = 100$ . . . . .	70

Figure 3.18	Comparison of $P^*$ using $b/h \in \{20, 100\}$ (a) $K_2 = 10$ (b) $K_2 = 30$	70
Figure 3.19	Comparing the expected cost, $P^*$ , and $R^*$ for various models using 70% utilization with E3 cost parameters and $b = 30$ . We report the results for various FG states by keeping $W_0 = 5$ to study the CF's queuing effect (a) Objective cost (b) $P^*$ (c) $R^*$ . . . . .	72
Figure 3.20	Comparing the expected cost, $P^*$ , and $R^*$ to study the effect of the CF shape parameter using $\mathcal{N}(9, 2)$ , $b = 30$ when $W_0 = 0$ (a) Objective cost (b) $P^*$ (c) $R^*$ . . . . .	73
Figure 4.1	$\Theta_t(X_{t-1}, W_{t-1})$ is convex in $P_t$ ; (a) Green color with $P^* = 0$ , (b) Blue curve with $P^*$ given by argmin of (4.3), (c) Orange curve indicates $P^* = P_{cap}$ . . . . .	91
Figure 4.2	Missbauer CF . . . . .	97
Figure A.1	Effect of $\alpha$ using Graves CF . . . . .	150
Figure A.2	For demand $\sim \mathcal{N}(7, 1)$ (a) For $\alpha=0.5$ (b) For $\alpha=0.25$ . . . . .	151
Figure A.3	For demand $\sim \mathcal{N}(7, 2)$ (a) For $\alpha=0.5$ (b) For $\alpha=0.25$ . . . . .	153
Figure A.4	For demand $\sim \mathcal{N}(9, 2)$ (a) For $\alpha=0.5$ (b) For $\alpha=0.25$ . . . . .	154
Figure A.5	$\mathcal{N}(7, 1)$ : Missbauer CF (a) For $K_2=10$ (b) For $K_2=30$ . . . . .	156
Figure A.6	$\mathcal{N}(7, 1)$ (a) UCPI (b) Graves CF: For $\alpha=0.5$ (c) Graves CF: For $\alpha=0.25$ . . . . .	157
Figure A.7	A graphical illustration of the steep and flat Missbauer and Graves CF showing the expected output near the origin is similar that is determined by the shape parameter of the CFs . . . . .	158
Figure A.8	Marginal production: Blue color represents slope near origin requires four units of workload, Green color represents slope near the curvature requiring nine units of workload, and Yellow color represents slope near the horizontal segment requires 51 units of workload . . . . .	159
Figure A.9	Optimal production and objective function using E1 . . . . .	160
Figure A.10	Using E2 when $W_0 = 0$ (a) Objective cost (b) $P^*$ (c) $R^*$ . . . . .	161
Figure A.11	Using E2 when $W_0 = 80$ : $P^*$ and objective cost . . . . .	161
Figure A.12	Using E3 when $W_0 = 0$ (a) Objective cost (b) $P^*$ (c) $R^*$ . . . . .	162
Figure A.13	Using E3 when $W_0 = 80$ : $P^*$ and objective cost . . . . .	163
Figure A.14	When $W_0 = 0$ , (a) Objective cost (b) $P^*$ (c) $R^*$ . . . . .	164
Figure A.15	When $W_0 = 80$ , (a) Objective cost (b) $P^*$ . . . . .	165

# Chapter 1

## Introduction

### 1.1 Problem statement and objective

The objective of a supply chain is to deliver the right product of the right quality at the right time in the right place at best possible total cost. Most supply chain networks are subject to capacitated resources and uncertain demand. Due to capacitated production resources, material flows, and uncertain processing times, production and inventory decisions must account for queuing behavior (Buzacott and Shanthikumar 1993). Planning and coordination in these networks can contribute significantly to competitiveness. In particular, to match supply and demand, we need to consider cycle times, the time between the material being released into the production system and its emergence as a finished product that can be used to meet the demand to release work to production resources at the right time. Queueing theory suggests that the probability distribution of the cycle time is determined, among other factors, by resource utilization, which, in turn, is determined by the work release decisions made by the planning system. The need to produce safety stock places additional demands on production facilities, resulting in longer, more variable cycle time due to the queueing behavior of capacitated production resources (Hopp and Spearman 2011). Three principal research streams address these problems: 1) Inventory models, 2) Queueing models, and 3) Mathematical programming.

Several authors have proposed heuristic and exact approaches to solve inventory problems under capacity restrictions (Bookbinder and Tan 1988; Cheng and Sethi 1999; Federguen and Zipkin 1986a,b; Karlin 1960; Ravindran et al. 2011; Scarf 1959; Schäl 1976; Tarim and Kingsman 2004; Veinott 1966; Veinott Jr 1965). For large problem instances,

exact solution methods suffer from the curse of dimensionality, often rendering the computation of optimal policies impractical.

On the other hand, queuing models (Buzacott and Shanthikumar 1992; Hopp and Spearman 2011) explicitly model the stochastic nature of the production process and the material flow between stages of the system. These models effectively capture the non-linear relationships between crucial system parameters and performance measures such as utilization, average in-process inventory, and cycle time. However, they are primarily descriptive. Their emphasis on steady-state solutions limits their usefulness for situations where the demand is a non-time-stationary stochastic process, which is generally the case in industrial practice.

This dissertation seeks to develop exact solution approaches for both single-period and multi-period production inventory problems whose replenishment process is governed by queuing behavior of the production process. We use a meta-model to represent the queuing behavior, also known as a clearing function. Since production from the current period meets the current period's demand, we consider a zero lead time model in both the single-period and multi-period problems. We aim to develop a solution that can be used as a benchmark to compare other approximate solutions from various other methods using chance-constrained models, stochastic programming, and robust optimization.

## 1.2 Motivation

Several authors use stochastic optimization methods to solve capacitated production inventory problems. Stochastic programming and robust optimization are computationally intensive and suffer from limited scalability. Defining the scenario generation tree in a stochastic programming model and developing the uncertainty set in a robust optimization model remain difficult in solving problems to reasonable solution quality. Chance-constrained programming models address stochastic aspects of a decision problem by allowing some constraints to be violated with a prespecified probability. However, chance-constrained models often fail to consider the cost consequences of constraint violations when they occur, which can lead to anomalous behavior (Blau 1974). The various stochastic approximation methods available in the literature are seldom tested against an exact benchmark solution, which prevents rigorous assessment of the quality of their solutions relative to optimality. Therefore, in this thesis, we aim to develop an exact solution method for both single-period and multi-period versions of the capacitated stochastic

production inventory problem subject to congestion. To gain initial insight, we start by considering the single-period version of the problem, where all production from the period can be used to meet the current period’s demand; yielding a single-period lead time zero model. We then develop an exact solution for the multi-period problem using stochastic dynamic programming to obtain a benchmark that can be used to assess the performance of approximation methods to illustrate the optimality gap and quality of the policy. We accelerate the SDP by exploiting the structure of the problem using a binary search procedure and forming an approximation scheme for solving large instances. Despite the curse of dimensionality, the binary search (BS) procedure helps the SDP algorithm run in pseudo-polynomial time<sup>1</sup>(Arora and Barak 2009; Garey and Johnson 1979).

To the best of our knowledge, these models based on exact solutions have yet to be explored in the context of capacitated stochastic production inventory problems whose replenishment process is governed by a clearing function.

### 1.3 Contributions

We propose an algorithm to solve the single-stage, single-item, single-period, capacitated, stochastic production inventory problems whose replenishment is governed by a clearing function. Using the Karush Kuhn Tucker (KKT) optimality conditions, we devise an approach to find optimal production and release quantities that minimize the sum of expected work in progress, finished inventory, and shortage costs. The derivation of the solutions is independent of the specific clearing function. Therefore, we test our solution methods using two different clearing functions, those of Graves (1986) and Missbauer (2002). We vary the shape parameter of the Missbauer CF (steep and flat), assuming different utilization, and characterize the optimal solution structure under three distinct cases. Second, we develop a stochastic dynamic programming algorithm to solve the finite horizon problem using a backward induction algorithm. The SDP algorithm involves four specific steps:

1. Demand discretization to represent any continuous demand distribution.
2. Cost minimization by solving the Bellman equation using a recursive backward induction algorithm.

---

<sup>1</sup>A pseudo-polynomial algorithm is an algorithm whose worst case complexity is polynomial in the numeric value of the input(not the number of inputs). In other words, it means that the worst-case run time of the algorithm is polynomial in the magnitude of the input, but exponential in the size of the input (which is a base-2 log of the magnitude). .

3. Function approximation using bilinear interpolation.
4. SDP acceleration using binary search procedure and implementation using parallel computing framework

We test the SDP algorithm computationally and study the effects of model assumptions and parameters on solution time quality.

## 1.4 Outline of the dissertation

The rest of the dissertation is organized as follows: Chapter 2 provides a review of previous methods to study inventory problems with finite and infinite horizons. We discuss multi-stage models with limited production capacity and their optimal policies and introduce the concept of the clearing function and its application in production planning problems. We then introduce a finite horizon backward induction dynamic program for the CF-based capacitated production inventory problem. Since value function approximations are an integral part of the DP recursion, we discuss function approximations and approximate dynamic programming methods in stochastic optimization. Lastly, we present a brief review of previous models used in production planning models using chance-constrained models, stochastic programming, and robust optimization, briefly discuss their relative advantages and disadvantages, and explain why we choose to proceed with SDP.

We propose an exact solution to the single-period zero lead time problem in Chapter 3. We characterize the solution using three cases and discuss the effects of clearing function shape, utilization, and cost parameters. Chapter 4 discusses a backward induction stochastic dynamic programming algorithm to solve the multi-period problem. We discuss various model assumptions, implementation, and acceleration methods to derive the optimal policy. We use this optimal policy to benchmark the approximate solutions in the literature. We present the principal results and future work in capacitated production inventory problems and other approximate DP methods in Chapter 5.

# Chapter 2

## Previous Related Work

### 2.1 Motivation

Most supply chain networks are subject to uncertainty in demand and limited production capacity. In such networks, nodes represent the production and inventory locations, while the links represent the transportation and information flows. In the case of a production-inventory system, the combined presence of limited capacity and variance in job arrival and processing times within the production facility lead to queuing behavior. Uncertainty in the external demand adds complexity to the systems, requiring safety stocks to meet service level requirements.

Planning and coordinating these networks can contribute significantly to competitiveness (Missbauer and Uzsoy 2020b). Research has been conducted on these planning and control problems in operations research and supply chain management since the inception of these fields. Three principal research streams have developed to solve such problems in the literature. 1) Mathematical programming, 2) Inventory Models, and 3) Queuing network models. We highlight some of the key contributions of production inventory problems in production planning under congestion and discuss various stochastic optimization techniques in the literature.

### 2.2 Classical Inventory Models

The inventory literature dates back to the early 1910s. Harris (1913) described a simple economic order quantity (EOQ) formula to find the optimal order quantity under a linear

holding cost, a fixed ordering cost, and constant deterministic demand (Erlenkotter 2014; Hadley and Whitin 1963; Harris 1915). Cárdenas-Barrón et al. (2014) and Andriolo et al. (2014) discuss various extensions, including backorders, lost sales, and stochastic demand. We first discuss single-stage models and their solution structures of uncapacitated, capacitated with and without fixed ordering costs. We then discuss the multi-stage models and their extensions. We restrict our literature review to newsvendor models and their extensions covering single-period and multi-period problems using dynamic programming models. We also review the ideas of stochastic dynamic programming as an exact solution method in a finite horizon setting and approximate methods to solve sequential decision-making problems under uncertainty.

### 2.2.1 Single-stage uncapacitated models

The newsvendor problem was first discussed mathematically in 1888 by Edgeworth (1888), who applied the concepts and tradeoffs inherent in the newsvendor model in the banking sector. Morse et al. (1951) were the first to use the term "newsboy" in referring to this problem in their book. The classical single-period newsvendor problem finds the order quantity that minimizes the sum of expected overage and underage costs under stochastic demand (Axsäter 2015) given by

$$g(S) = h \int_0^S (S - D)\phi(D)dD + b \int_S^\infty (D - S)\phi(D)dD \quad (2.1)$$

where  $S$  is the base-stock level,  $D$  a non-negative random variable representing demand with pdf  $\phi(\cdot)$ , CDF  $\Phi(\cdot)$ ,  $h$  the unit overage cost, and  $b$  the unit underage cost. Differentiating with respect to  $S$  yields the optimal value  $S^* = \Phi^{-1}\left(\frac{b}{b+h}\right)$ , where  $\Phi(S) = P(D \leq S)$  denotes the probability of no stockout given by the CDF of the random variable  $D$ . For the classical newsvendor problem, a basestock policy is optimal without fixed ordering cost since the cost function is convex (Zipkin 2000). A base stock policy works as follows: we observe the current inventory position  $X$  and then place an order whose size will bring the inventory position up to  $S$  (order  $Q = S - X$  units, where  $Q$  represents the order quantity). However, if the inventory position exceeds  $S$ , the optimal policy is to do nothing. i.e,  $Q = 0$ .  $S$  is a constant and does not depend on the system's initial state  $X$  (Snyder and Shen 2019). Base stock policies are sometimes referred to  $(S - 1, S)$  policies. The order-up-to level  $y$  is related to the base-stock level  $S$ , but they are not the same. The difference lies in the definition of  $y$  and  $S$ . The order-up-to-level depends on the initial inventory position  $X$ . If  $X < S$ , then  $y = S$  and if  $X \geq S$ , then  $y = X$ . In

contrast,  $S$  is a fixed number independent of  $X$ . Extending the newsvendor model with a fixed ordering cost  $K$ , and current linear holding and shortage costs leads to the  $(s, S)$  policy derived by Karlin (1960). An  $(s, S)$  policy works as follows: we observe the current inventory position  $X$ , and if  $X \leq s$ , we place an order to bring the inventory position to  $S$ . Both  $s$  and  $S$  are constants, and  $s \leq S$ . The quantity  $s$  denotes the reorder point, and  $S$  the order up to level. The total expected cost in the period as a function of  $s$  and  $S$  is given by

$$g(s, S) = \begin{cases} K + g(S), & \text{if } X \leq s \\ g(X), & \text{if } X > s \end{cases} \quad (2.2)$$

Optimizing  $s$  and  $S$  should be done by setting  $S^* = \Phi^{-1}\left(\frac{b}{b+h}\right)$ , and  $s^*$  such that  $s^* \leq S^*$  and  $g(s^*) = g(S^*) + K$ . Unlike a basestock policy, the convexity of the cost function is lost when fixed costs are present in a classical inventory problem. The concept of  $K$ -convexity was developed specifically for proving the optimality of  $(s, S)$  policies as described in Zipkin (2000) and Snyder and Shen (2019).

Interest in the single-period problem remains unabated as it forms a natural building block for analyzing the multi-item and multi-period problems (Choi 2012; De Kok 2018; Khouja 1999; Qin et al. 2011). This problem reflects many real-life situations in decision-making under uncertainty in the perishables, fashion, sports, and retail industries.

### 2.2.2 Single-stage capacitated models

There are many variations of newsvendor problems with infinite capacity. Our research focuses on capacitated production inventory problems by extending the classical newsvendor model with a more complex capacity model considering queuing behavior of the production resources involved in replenishment. When the optimal policy parameters depend on the system's initial state, i.e., the current inventory level, we call this optimal policy a state-dependent basestock policy, unlike a classical newsvendor model, whose optimal policy is state-independent with zero fixed costs. We now discuss relevant literature on classical newsvendor models with limited capacity.

The capacitated single period problem finds the order quantity that minimizes the sum of expected overage and underage costs (2.1) under stochastic demand (Axsäter 2015) when the order quantity cannot exceed a certain production capacity  $P_{max}$ . The production cannot achieve the order up to level ( $S = X_0 + P^*$ ) if  $P^* \geq P_{max}$ . In that case, the optimal order up to level becomes ( $S = X_0 + P_{max}$ ). Applying the standard

Lagrange multiplier procedure gives,

$$L(S, \lambda) = \min_{\lambda \leq 0} \left\{ h \int_0^S (S - D)\phi(D)dD + b \int_S^\infty (D - S)\phi(D)dD - \lambda[S - P_{max}] \right\} \quad (2.3)$$

Differentiating with respect to  $S$  and setting to zero yields the optimal value  $S^* = \Phi^{-1}\left(\frac{b+\lambda}{b+h}\right)$ , where  $\Phi(S) = P(D \leq S)$  denotes the probability of no stockout given by the CDF of the random variable  $D$ . Rewriting  $S$  in terms of  $\lambda$  gives

$$\lambda = h\Phi(S) - b(1 - \Phi(S))$$

which is the expected benefit of the marginal unit (*EBMU*) of product at the optimal stocking level  $S$ . For the capacitated newsvendor problem, a modified basestock policy is optimal without fixed ordering cost since the cost function is convex (Federgruen and Zipkin 1986a). A modified base-stock policy works as follows: Given the initial inventory  $X$ , when the initial stock is below the modified base-stock level, produce enough to bring the stock up to  $S$ , or as close to it as possible, given the limited capacity  $P_{max}$ ; otherwise do not produce. These discussions already exist in standard textbooks of operations research and management science.

There are a wide range of extensions to the basic capacitated model, including additional constraints, multiple products, and product substitution. For additional references on capacitated newsvendor models, we direct the reader to Abdel-Malek et al. (2020); Hadley and Whitin (1963); Lau and Lau (1996, 1997). To the best of our knowledge, the work in this thesis represents the first attempt to study the single-period capacitated production inventory problem subject to congestion using queuing-based replenishment using a non-linear clearing function.

### 2.2.3 Single stage multi-period models

For multiple period problems with zero fixed costs, a basestock policy is optimal in both finite and infinite horizon setting under linear overage and underage costs (Axsäter 2015). For discussions related to proofs in the finite and infinite horizon settings with non-zero fixed costs, we direct the reader to Axsäter (2015); Scarf (2002); Snyder and Shen (2019); Zipkin (2000). We now discuss problems involving multiple periods with non-zero fixed costs that lead to  $(s, S)$  policies. Early research led by Arrow et al. (1951) proposed a dynamic programming model to solve the inventory model using an  $(s, S)$  policy structure. Karlin (1960) obtained a myopic policy for the uncapacitated classical

inventory problem. There is an abundant literature on optimal  $(s, S)$  policies in Cheng and Sethi (1999); Karlin (1960); Scarf (1959); Schäl (1976); Veinott (1966); Veinott Jr (1965); Veinott Jr and Wagner (1965).

Many researchers studied stochastic inventory problems to characterize the structure of their optimal policies. Iglehart (1963) provides upper and lower bounds for the sequences  $s_n$  and  $S_n$  in each period  $n$  where  $(s_n, S_n)$  denotes the reorder point, and the order upto level, and discuss their limiting behavior as  $n$  tends to infinity. The limiting  $(s, S)$  policy characterizes the optimal ordering policy for the infinite horizon problem. Johnson (1968) gives new conditions for optimality of  $(s, S)$  policies and develop a computational method using total discounted cost and the average cost criteria in an infinite horizon single item inventory problem with non-zero fixed costs and purchase costs. Waldmann (1984) presents an optimal state-dependent  $(s, S)$  policy based on two critical numbers. Zheng (1991) provides a simple proof for optimality of  $(s, S)$  policies in an infinite horizon inventory system for the discounted-cost and the average-cost models. See Axsäter (2015); Zipkin (2000) for in-depth coverage of the  $(s, S)$  policy structure. For essentially all multi-period problems, the only means of obtaining an exact optimal policy is some form of stochastic dynamic programming in a finite horizon setting, which suffers from the curse of dimensionality. For infinite horizon problems, authors have used Markov Decision Process (MDP)-based policy iteration and value iteration procedures to find the optimal policy (Howard 1960).

#### **2.2.4 Single stage capacitated multi-period models**

In the last four decades, there have been a number of papers discussing optimal policies for the capacitated production inventory problems. Federgruen and Zipkin (1986a,b) assume a finite production capacity and prove the optimality of a modified base stock policy under both average-cost and discounted-cost criteria in both finite and infinite horizon settings. The authors use stochastic dynamic programs to compute the optimal base-stock levels. Ciarallo et al. (1994) prove the optimality of modified base-stock policies under a stochastic capacity constraint. Tayur (1993) develops methods to compute optimal base-stock levels for capacitated inventory systems that rely on the analysis of shortfalls. Glasserman (1997) develops bounds and approximations for setting base-stock levels in production inventory systems with deterministic and stochastic capacity constraints. Aviv and Federgruen (1997) proved that modified base-stock policies are optimal for the finite horizon planning model of a capacity-restricted production inventory system

under periodically varying demand and cost patterns and for both the infinite-horizon discounted and undiscounted cost criterion. The authors showed that the optimal base-stock levels can be calculated via a simple but efficient value-iteration method. A discussion of these types of policies is given in Zipkin (2000). Kapuściński and Tayur (1998) develop a simulation-based optimization method to compute the optimal parameters of the policy and characterize some of its properties. To prove the optimality result for the average cost case, the authors derive several technical properties of base-stock policies - convexity, regeneration, coupling, and stability. The models developed using a capacity restriction simply ignore the dependency between load and lead times.

In this research, we attempt to study a single-stage, multi-period problem using a stochastic dynamic program to find the optimal policy. Using release and production as actions, we develop two SDP algorithms and study the structure of the optimal policy. To the best of our knowledge, the work in this thesis represents the first attempt to study the single-stage, multi-period capacitated production inventory problem subject to congestion using queuing-based replenishment using a non-linear clearing function. We discuss the details of this model with several empirical experiments in Chapter 4. We discuss the optimal policy with respect to the non-linear clearing function, utilization, demand patterns of varying means across  $T$  periods, and cost parameters.

### 2.2.5 Multi-stage multi-period models

Inventory policies are derived for multi-echelon supply chain systems by extending the classical inventory theory. Optimal inventory policies depend on supply chain network topologies (Snyder and Shen 2019): Serial system, Assembly system, Distribution system, Tree system, and General system. In a serial system, except for the initial and terminal stages, every stage has exactly one predecessor and one successor. In an assembly system, each stage has at most one successor but can have multiple predecessors. A distribution system is the opposite of the assembly system wherein each stage has at most one predecessor but can have multiple successors. On the other hand, Tree systems are typically a hybrid of both assembly and distribution systems - each stage may have multiple predecessors and successors but the network may contain no undirected cycles<sup>1</sup>. Lastly, general systems are the most complicated supply chain topologies that allow any number of predecessors and successors and may contain undirected cycles. We now discuss some of the

---

<sup>1</sup>An undirected cycle in a supply chain network is a cycle that results from removing all the arrows from the source links at each stage so that the movement of materials can go in any direction from upstream to downstream or vice versa.

state-of-the-art research conducted in all these supply chain topologies; see De Kok and Graves (2003) and references therein.

**Serial system:** Two primary models have been developed to handle multi-echelon inventory systems: stochastic-service and guaranteed-service models. In stochastic-service models, each stage in the network sets an echelon base-stock level  $S_i$  and meets demand from stock whenever possible. The actual lead time for an order to be filled in the downstream stages is stochastic since unmet demand will be backordered. Clark and Scarf (1960) proved the optimality of an echelon basestock policy for finite-horizon problems using stochastic service models, whereas Federgruen and Zipkin (1984) proved it for the infinite horizon serial supply chain problems. We define an echelon base-stock policy as follows. Under appropriate assumptions, Clark and Scarf (1960) showed that the optimal policy for their model is characterized by a critical number for each facility: in each period, each facility places an order just large enough to restore its cumulative downstream inventory position to the critical number. These critical numbers are called base-stock levels, and because they refer to cumulative downstream inventory, the policy is called an echelon base-stock policy. Glasserman and Tayur (1994) examine the stability of a multi-echelon system in which each node has a limited production capacity and operates under a base-stock policy. For all stochastic service models, the only means of obtaining an exact optimal policy is some form of stochastic dynamic programming, which suffers from the curse of dimensionality. On the other hand, guaranteed-service models set a committed service time ( $CST$ ) within which the external demand will be met with certainty. To guarantee this service, these models require demand to be bounded above. Simpson Jr (1958) proposed the first guaranteed service model for multi-echelon inventory problems which was later extended by Inderfurth (1991) and Graves and Willems (2000, 2003a,b).

The main difference between these two approaches lies in the penalty incurred for the excess demand. For stockouts, stochastic-service models incur a penalty proportional to the time the unit is backordered, whereas in guaranteed service models, no penalty is incurred until the backorder has lasted  $CST$  periods, and after that, the penalty is infinite. For multi-echelon models, especially for serial systems, exact DP formulations have been developed for both stochastic service (Chen and Zheng 1994; Clark and Scarf 1960) and guaranteed service approaches (Graves 1988; Graves and Willems 2000; Inderfurth 1991).

**Assembly system:** Rosling (1989) demonstrated that the assembly system could be treated as a serial system under a restriction on the initial stock levels in long-run balance

from the very first period. Hence, simple echelon basestock policies are optimal, and the computational requirements are drastically reduced for solving an assembly system. i.e., once the assembly system is transformed to an equivalent serial system, the exact method by Clark and Scarf (1960)'s approach and heuristic solutions by Shang and Song (2003) can be applied conveniently to yield an echelon base-stock policy. Van Houtum and Zijm (1991) discuss the computational issues in solving multi-echelon production systems, in particular for serial and assembly systems. The authors show that a pure assembly system is equivalent to a serial system. De Kok and Visschers (1999) propose a decomposition method for general assembly systems that decomposes the assembly network into purely divergent multi-echelon systems. The key idea is to pre-allocate common components to end products. This method is inspired by Rosling's analysis of the relation between assembly systems and serial multi-echelon systems. The optimal solution follows an echelon base-stock policy after decomposing into a serial system and the numerical experiments shown by simulation yield promising results.

**Distribution system:** The main difficulty in distribution systems arises when there is insufficient inventory at a node to meet the orders placed by its successors. The simplest form of the distribution system is one warehouse supplying multiple retailers' (OWMR) networks. How to allocate the inventory from upstream to downstream is a challenging question. Two key policies need to be determined. 1) Replenishment policy and 2) Allocation policy. The best-known exact solution for OWMR is the projection algorithm (Graves 1985), which involves iterating over all possible base-stock values for the warehouse. This method is computationally intensive and requires numerical convolution. Several heuristics have been proposed for OWMR and more general distribution systems. The concept of pooling originated with Eppen (1979) and Eppen and Schrage (1981) work on multi-echelon inventory systems with geographically dispersed stocking locations. In Eppen and Schrage (1981), the depot orders from the supplier and allocates products to the warehouse each period or every  $m$  periods. They show that the reduction of total inventory from pooling is greater as the depot's review period increases and is proportional to the number of products (Alfaro and Corbett 2003). The authors derive an explicit expression for the optimal order quantity under the inventory balance allocation assumption with identical warehouses. For example, 8 units at the depot are to be shipped to two warehouses; initial inventory at warehouses 1 and 2 = 10 units each. Let demand at warehouse 1 = 6 units, and demand at warehouse 2 = 4 units in period  $t$ . Then in period  $t + 1$ , net inventory in warehouse 1 becomes  $10 - 6 + 5 = 9$  units; warehouse

2 becomes  $10-4+3 = 9$  units. The allocation is based on the percentage of demand at each warehouse to the total supply units from the depot.

De Kok and Seidel (1990) analyze the stock allocation policy in a two-echelon distribution network using service constraints. The authors propose heuristic algorithms for service level computations and use an empirical optimization procedure to determine the optimal stock under different system settings. Using simulation, the results are validated and demonstrate the adequacy of the methods. Diks and De Kok (1998) prove that under the balance allocation assumption in a divergent system, it is cost-optimal to control every facility by an order-up-to-policy. The optimal order-up-to-level and the allocation functions at each facility can be determined by system decomposition. This decomposition reduces complex multi-dimensional control problems to simple one-dimensional problems. Diks and De Kok (1999) develop a decomposition-based near-optimal heuristic to solve a divergent supply chain network that determines the control parameters of a replenishment policy. A simulation study of a divergent three-echelon system reveals that this algorithm performs well. Several other decomposition-based heuristics have been proposed. See Gallego et al. (2007); Graves (1985); Rong et al. (2017) and references therein. For additional details on multi-echelon divergent systems, we direct the reader to Diks et al. (1996).

**Tree and General systems:** These systems are more complex than the conventional serial and assembly systems. Guaranteed-service models are the most common technique to solve the general network topology. Graves and Willems (2000, 2003a) describe a stochastic dynamic programming algorithm that runs in pseudopolynomial time on a spanning tree network. The time complexity is given by  $\mathcal{O}(NM^2)$ , where  $N$  represents the number of stages in the network and  $M$  denotes the maximum service time and is bounded by  $\sum_{j=1}^N T_j$  where  $T_j$  denotes the deterministic production lead time in stage  $j$ .

Recently algorithms are developed that solve NP-hard problems by relaxing either the constraint on the running time or the constraint of optimality. i.e., they are either not polynomial in the worst case but solve some instances of the given problem fast, or they are polynomial but usually cannot guarantee an optimal solution (Korom 2017). We discuss two of them to solve the general network safety stock problem using the guaranteed service model. Lesnaia (2004) develops a polynomial time algorithm for a general supply chain network. The goal of this dissertation is to describe an algorithm to solve the problem without assuming any particular structure of the underlying supply chain.

The authors show that the general supply chain network problem is NP-hard by reducing the known vertex cover problem to an instance of the general supply chain network and develop several conditions that characterize an optimal solution to the general network problem. Consider a two-stage network given in Figure 2.1 where  $i$ , and  $j$  denote the supply and demand nodes in a network represented by arcs  $\mathcal{A}$ . Let  $\mathcal{C}$  and  $\mathcal{D}$  denote the inbound component nodes and demand end nodes of the network. Let  $SI$  and  $S$  represent the inbound and outbound service times. Let  $T$  and  $s$  denote the production time and customer service times, respectively.

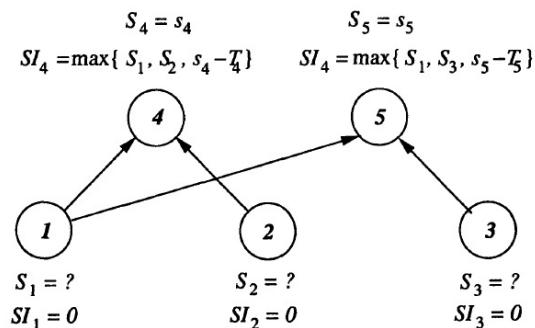


Figure 2.1: A simple two-stage network defined by Lesnaia (2004)

The optimality conditions for the two-stage network are as follows: There always exist an optimal solution such that  $S_j = \min\{s_j, SI_j + T_j, j \in \mathcal{D}\}$ ,  $SI_i = 0, i \in \mathcal{C}$ , and  $SI_j = \max\{s_j - T_j; S_i, \text{ for } i : (i, j) \in \mathcal{A}\}$ ; see Lesnaia (2004) for the proofs illustrated with network examples. The idea is that we can identify all possible candidates for the optimal service times for a stage by constructing paths from one stage to each other stage in the supply chain. Using these constructs (paths) as the basis for a branch and bound (B&B) algorithm, they solve the general network problem to optimality. To generate lower bounds, the authors use the idea of Graves and Willems (2000, 2003a) by solving a spanning tree relaxation of the general network problem and provide a polynomial algorithm to solve these spanning tree problems. In addition to a general network, the authors consider a two-stage network problem and propose a specialized branch and bound algorithm whose time complexity is given by  $\mathcal{O}(N^3)$ , where  $N$  represents the number of stages in the network; see Graves and Lesnaia (2004); Lesnaia (2003); Lesnaia et al. (2005) for details of the proofs and specialized B&B algorithm.

Magnanti et al. (2006) develop a mixed integer programming solution method for a

general network that solves moderately sized instances of the guaranteed service model. Humair and Willems (2011) extend the seminal paper by Graves and Willems (2000) to solve general systems using exact and approximate heuristics. The authors reported less computation time, and the optimality gap for heuristic solutions was close to 1.1%. Recently, Graves and Schoenmeyr (2016) extended the DP approach to solving a tree system under capacity restriction. For further details on topologies, we direct the authors to refer to De Kok and Fransoo (2003); De Kok and Visschers (1999); Snyder and Shen (2019) and reference therein.

### **2.2.6 Multi-stage multi-period capacitated models**

Echelon base-stock policies are optimal for multi-stage inventory systems. Finding the optimal policy using a dynamic programming approach suffers from the curse of dimensionality. Several authors attempt to find simple approximations for multi-stage production inventory systems with limited production capacity. Glasserman and Tayur (1995) consider a capacitated multi-echelon systems operating under base-stock policies and develop estimators of derivatives with respect to base-stock levels. The infinitesimal perturbation analysis(IPA)-based methods for estimating sensitivities of inventory costs with respect to policy parameters converge to the correct value for finite, infinite, and average cost criteria. Glasserman and Tayur (1996) develop a simple approximation for multi-stage production inventory systems with limited capacity. The key step in this approximation is the steady-state distribution of echelon shortfall (base-stock level minus inventory position) by a sum of exponentials. The echelon shortfalls have exponential tails (Glasserman 1997); the parameters of the exponentials are chosen to match asymptotically exact expressions. In a test bed of 72 problems with five production stages, the average relative error for the approximate solution is 1.9%. Huh and Janakiraman (2010) study a capacitated assembly system using an echelon base-stock policy. The authors derive the convexity properties of the shortage costs with respect to the base-stock levels and capacity levels. Recently, Graves and Schoenmeyr (2016) extended the DP approach in solving the single-stage and multi-stage supply networks with capacity restriction. The authors generalize the guaranteed-service (GS) model for safety-stock placement in supply chains to include capacity constraints. The authors defined a new heuristic policy known as the modified constant base-stock policy, unlike the classical constant base-stock policy defined by Graves and Willems (2000) for the serial system. Results indicate that the modified constant base-stock policy performs well in a capacitated serial system in comparison with the best constant base-stock policy in an identical but uncapacitated

system. Multi-echelon inventory models are outside the scope of this thesis. For references related to typology and stochastic multi-echelon inventory models, we direct the reader to De Kok et al. (2018). For additional details on stochastic and guaranteed service models, refer to Graves and Willems (2000, 2003b); Snyder and Shen (2019); Van Houtum et al. (1996); Zipkin (2000) and references therein.

## 2.3 Production Planning under Congestion

The objective of production planning (PP) is to plan the work release into a production system over time to match the output to demand effectively. Production planning decisions determine the amount of work released into the production process in each period. Stochastic optimization methods have been widely applied to solve multi-period production planning problems. Deterministic production planning models were among the earliest applications of optimization models (Holt and Modigliani 1960; Modigliani and Hohn 1955). The extensive literature proposing a wide range of formulations has been discussed by Voß and Woodruff (2006) and Missbauer and Uzsoy (2011), among others.

Mula et al. (2006), and Verderame et al. (2010) present extensive reviews of PP models in the past. Missbauer (2011) and Missbauer and Uzsoy (2020b) review optimization models of production planning problems and analyze new research directions. They focus on models that support production volumes and order release decisions over time and highlight issues related to determining planned lead times. Missbauer and Uzsoy (2020b) discuss additional details of PP models under capacitated resources and congestion.

### 2.3.1 Order release in the production planning

A fundamental concept in production planning is the cycle time (Hopp and Spearman 2011) of each production unit - the time elapsing between a production order being released to the unit and its completion (Missbauer and Uzsoy 2021). Queuing theory shows that the cycle time of a production order through a production unit is a random variable whose probability distribution depends on the resource utilization level (Buzacott and Shanthikumar 1993). In order to release production orders in time to complete when needed, knowledge of cycle times is required. However, the cycle time is a random variable representing an outcome of the release decisions that determine the state of the production facility, constituting a circularity. This circularity, since its inception,

remains a challenging problem in production planning. Since exact analysis of the transient behavior of simple queuing systems remains arduous, computationally tractable approximations of the queues are needed for practical purposes. We now discuss three basic approaches to the circularity issues in the production planning context.

### Exogenous fixed lead time

The first approach to addressing circularity is treating the cycle time as an exogenous planned lead time  $L$  independent of workload, such that material entering the production system at time  $t$  will emerge as a finished product at time  $t+L$ . Let  $R$  denote the quantity released to the production system and  $P$  the production output. Thus, by flow balance equation for multiple products  $i$  in an uncapacitated system, the quantity  $R_{t-L}$  released in period  $t-L$  will equal the production output  $P_{it}$  at period  $t$ . We have  $P_{it} = R_{i,t-L}$ . If we impose a capacity constraint that limits the output of the production resource, the system dynamics change. As long as the total workload to be processed in a certain time remains below the maximum possible production  $P_{max}$ , order releases in period  $t-L$  will emerge in period  $t$ . In contrast, if this workload exceeds the capacity, demand  $D$  must be met by pre-build inventories or back-ordered. The inventory balance equation thus becomes  $I_{it} = I_{i,t-1} + R_{i,t-L} - D_{it}$ ,  $\sum u_i R_{i,t-L} \leq P_{max}$ , where  $u_i$  denotes the processing time of product  $i$  on the resource,  $D_{it}$  the demand for product  $i$  in period  $t$ ,  $I_{it}$  and  $I_{i,t-1}$  the ending and beginning finished goods inventory of product  $i$  in period  $t$ , respectively.

The flexibility of using exogenous lead time in any planning problems makes it easy to explain and leads to computationally tractable models in uncapacitated stochastic inventory optimization (Zipkin 2000), mathematical programming models (De Kok and Fransoo 2003; Hackman and Leachman 1989), materials requirements planning (Vollmann et al. 1997), and production smoothing models (Pochet and Wolsey 2006). In a typical production inventory system, the raw material to FG transition happens at the end of the period, i.e., capacity consumption at the end of the period  $T$ . However, exogenous lead time-related models do not necessarily consume capacity only at the end of period  $t+L$ . Depending on the problem type and the planning horizon, capacity can be consumed either at the end of the period  $t+L$ , at a different point in lead-time  $L$ , or in multiple periods. For instance, Spitter et al. (2005) developed two linear programming models to solve capacity-constrained supply chain operations planning (SCOP) problems where the demand for the end items is assumed to be exogenously determined, but capacity can be consumed in any interval during the planned lead times. The authors point out

that setting cost-optimal planned lead times is extremely complex and requires extensive experimental study, which is left for further research. Refer to Spitter et al. (2003) for some initial issues about planned lead times in SCOP models.

Despite these advantages, assuming that the planned lead time is independent of workload ignores the queuing behavior of the production system (Missbauer and Uzsoy 2021). At low workload levels, the cycle time will be overestimated, while it is likely to be underestimated at high levels. Second, determining the optimal lead time in a production system is not straightforward, especially if the observed cycle times are random variables subject to a probability distribution. Using time-varying lead time parameters for time-varying demand is intuitive. However, studies based on time-varying lead time parameters motivated by traffic modeling show the production orders released earlier cannot execute after those released later (Carey 1987). This means that the resulting mathematical formulation becomes significantly larger or yields to non-convex optimization problems (Carey 1992) that are difficult to solve to global optimality (Carey and Bowers 2012).

### **Iterative multi-model approaches**

This approach seeks to address the circularity by decomposing the order release problem into two subproblems: 1) Determine the optimal releases given a planned lead time for each period, 2) Estimate the cycle time that will be realized under those releases (Bang and Kim 2009; Hung and Leachman 1996; Irдем et al. 2008; Kacar et al. 2011, 2013, 2016; Kim and Kim 2001). The release planning sub-problem is formulated as an LP, while cycle time estimation is performed using simulation, queuing, or statistical prediction. Hence this approach represents a hybrid analytical/simulation approach.

The key advantage of using an iterative multi-model scheme is that this approach captures the cycle time estimation with rigorous system dynamics defined by the simulation models. These cycle time estimates help in solving the planning sub-problem formulated as an LP model. Despite some advantages, these methods suffer high computational time due to the use of the simulation model in the cycle time estimation. In addition, the convergence of these methods is not well understood, exhibiting several variations from one implementation to another (Irдем et al. 2010), with evidence of convergence to local optima depending on the initial lead-time estimates. Therefore, machine learning and data analytics techniques can be deployed to replace the simulation model, significantly reducing the computational burden while retaining good performance. For instance, Li et al. (2016) develop a metamodel-based Monte Carlo simulation (MCS) method to accu-

rately capture the dynamic, stochastic behavior of a manufacturing system and to allow real-time evaluation of a release plan’s performance metrics. This evaluation capability is then embedded in a multi-objective optimization framework to search for near-optimal release plans. The proposed method has been applied to a scaled-down semiconductor fabrication system to demonstrate the quality of the metamodel-based MCS evaluation to quantify the relationship between a release plan of jobs and its resulting performance metrics. Kacar et al. (2011) show that the clearing function-based models(described in the next subsection) frequently outperform multi-model methods, and Missbauer (2020) discuss the shortcomings of multi-model methods in convergence to the optimal global solution.

### Clearing functions

A more recent approach is the use of a non-linear clearing function (CF) that can be viewed as a metamodel of the queuing system representing the production resource. The problem of how to anticipate the releases at the planning level are relatively complex when it involves the following: 1) multiple products, 2) probability distribution of interarrival times between jobs (mean =  $t_a$ ), with mean arrival rate,  $\lambda = \frac{1}{t_a}$ , 3) squared coefficient of variation (SCV)  $c_a^2$ , and 4) the probability distribution of effective processing time with a mean ( $t_e$ ), with the service rate,  $\mu = \frac{1}{t_e}$  and the SCV  $c_e^2$ . The average utilization of the production resource is given by  $u = \frac{t_e}{t_a}$ . For a G/G/1 queue (Kingman 1961), at steady state, the expected cycle time  $T$  is approximated by

$$T = \frac{c_a^2 + c_e^2}{2} \left( \frac{u}{1-u} \right) t_e + t_e \quad (2.4)$$

Two key observations are noted in the above relationship (Missbauer and Uzsoy 2020b): 1)  $T$  increases non-linearly with  $u$ , eventually tending to  $\infty$  as  $u$  approaches 1. This behavior shows that work release decisions affect the average cycle time of the planning level. 2)  $T$  is affected by the variability  $c_a^2$  in the material flow into the production resource and variability  $c_e^2$  in the production process.

To illustrate, it is helpful to represent our production resource as a G/G/1 queue. The number of customers in the queue at a given point corresponds to the amount of WIP at the production facility, which is a random variable,  $W = \mathbb{E}[WIP]$  denoting the expected WIP level in the production system. Following the queuing analysis, in steady-state, we assume the average arrival rate of the system is equal to its average processing rate ( $P$ ),

and according to Little's law, (Little 1961),

$$W = PT = \frac{T}{t_a} \quad (2.5)$$

where  $T$  is the average cycle time, and  $W$  is the average WIP level. This means the production rate  $P$  can be achieved either by controlling the average cycle time to achieve the desired WIP level or by controlling the average WIP level to achieve an average cycle time. Combining (2.4) and (2.5), the expected WIP of the G/G/1 queue (Medhi 2002) is

$$W = \frac{c_a^2 + c_e^2}{2} \left( \frac{u^2}{1-u} \right) + u \quad (2.6)$$

The average utilization  $u$  can be interpreted as the long-run fraction of time the resource will be busy, thus producing output. Solving for  $u$  in terms of  $W$  yields a quadratic equation whose positive solution is given by

$$u = \frac{-(W+1) + \sqrt{(W+1)^2 + 4(C-1)W}}{2(C-1)} \quad (2.7)$$

where  $C = \frac{c_a^2 + c_e^2}{2}$  represents the variance term and  $C \neq 1$ . For an M/M/1 queue,  $C = 1$  and (2.7) simplifies to  $W = \frac{u}{1-u}$  yielding  $u = \frac{W}{W+1}$ . If we plot the relationship between average utilization and average WIP for given values of  $t_e$  and  $C$ ,  $u$  is a monotonically non-decreasing concave function of  $W$ ; as the average WIP increases,  $u$  increases at a decreasing rate. This means the higher the WIP in the system, the lower the probability that the production resource will be idle due to starvation; therefore, maintaining a given average throughput requires maintaining a sufficiently high average WIP in the system.

Several authors obtain empirical relationships between WIP and throughput and WIP and utilization to understand the effective output of the production system under capacity restriction (Missbauer and Uzsoy 2020b). Such relationships can be derived under steady-state, either by closed-form analytical models (Asmundsson et al. 2009, 2006) or by empirical methods (Irdem et al. 2008; Kacar 2012; Kacar et al. 2011, 2016) fitting a functional form or data obtained from the simulation model. We shall refer to these functions as **clearing functions** since they represent the ability of the production facility to convert a fraction of its workload into output in a planning period. For additional resources, we direct the reader to refer to Missbauer and Uzsoy (2020b, 2021).

Graves (1986); Karmarkar (1989); Missbauer (2002); Srinivasan et al. (1988) use queuing theory to derive the relationship between workload and expected output. Despite the differences between various assumptions made by authors, most of the CFs discussed in the literature have the following properties (Gopalswamy 2019).

1. CFs are concave, continuous, monotonically, non-decreasing functions
2.  $\lim_{\Lambda \rightarrow +\infty} f(\Lambda) = P_{max}$ , where  $P_{max}$  is the maximum capacity of the production system and  $\Lambda$  denotes the workload available.
3. The curvature of the CFs decreases with increasing variability of the production system based on the steady-state queuing approximations

Several CFs from the literature are shown in Figure 2.2. Most conventional LP-based models assume the maximum capacity as a fixed quantity independent of workload, as shown in Figure 2.2 below.

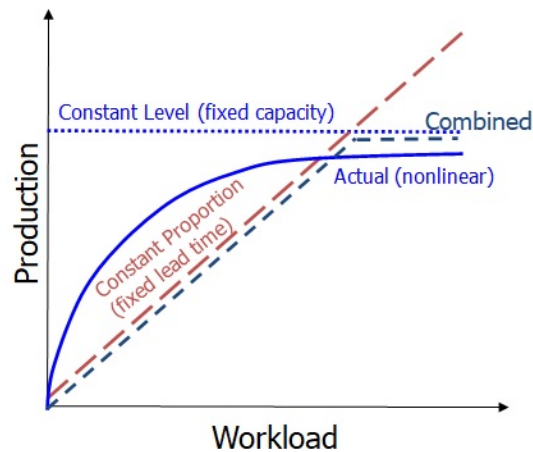


Figure 2.2: Basic clearing function

The constant proportion function proposed by Graves (1986) assumes that production system is managed to maintain a fixed cycle time, and the system’s capacity is not limited, as shown by the red line. This linear function implies that the production resource can convert a constant fraction of its workload into output in a planning period, regardless of the available workload. However, in a typical clearing function, the expected output is limited by a fixed maximum capacity limit shown by the blue dotted line. The clearing function is a concave, non-decreasing function indicated by the blue curve limited by a bounded capacity. The primary reasoning behind this asymptotic behavior is that as

the workload increases, the probability of the resource being idle due to lack of work decreases. Since most classical queuing models assume inter-arrival time and service time distributions with infinite support and long tails, there is always a non-zero probability of the resource being idle in a planning period. Therefore, the expected output approaches the theoretical maximum output only asymptotically.

**Types of CFs** This section will review some of the commonly used clearing functions in the production planning literature (Missbauer and Uzsoy 2020b). Since a CF represents the expected output as a function of workload, it captures the effects of load-dependent lead times (Kacar et al. 2011). At the start of period  $t$ , the resource is assumed to have  $W_{t-1}$  units remaining from the previous period. In period  $t$ ,  $R_t$  units are released to the resource, resulting in a workload of  $\Lambda_t = W_{t-1} + R_t$  units. Thus, the output of the resource in period  $t$  is estimated by the clearing function

$$P_t = f(W_{t-1} + R_t) \quad (2.8)$$

Several authors have proposed various empirical clearing functions based on workload and constant cycle time. These empirical models assume various queuing behavior at a steady state, and the relationship is captured between the workload and the output using appropriate parameters. A list of workload-based and constant cycle time-based CF proposed by Graves (1986); Karmarkar (1989); Missbauer (2002) are discussed in Missbauer and Uzsoy (2020b).

Many authors using single-variable clearing functions in optimization models have chosen to approximate the concave clearing function by outer linearization. These piecewise linear functions allow us to solve multi-period production inventory models using linear programming.

$$P_t = \min \left( a_k(R_t + W_{t-1}) + c_k \right) \quad \forall k \in K$$

Here  $a_k$  is the slope and  $c_k$  the intercept of segment  $k$  of the piecewise linearized clearing function. For multi-item problems, Asmundsson et al. (2009) proposed an allocated clearing function model for a multi-product planning problem, where the workload  $\Lambda_t$  is defined as

$$\Lambda_{it} = \sum_i u_i(R_{it} + W_{i,t-1}) \quad \forall i \in I, t \in T$$

where  $u_i$  is the amount of time required on the resource to produce one unit of item  $i$ . The total output the resource can produce, in units of time, depends on the workload  $\Lambda_t$  in that period, which is then allocated among the different products  $i \in I$ , yielding the constraint set (Missbauer and Uzsoy 2021)

$$I_{it} = I_{i,t-1} + P_{it} - D_{it} \quad \forall i \in I, t \in T$$

$$W_{it} = W_{i,t-1} + R_{it} - P_{it} \quad \forall i \in I, t \in T$$

$$u_i P_{it} \leq Z_{it} f\left(\frac{\Lambda_{it}}{Z_{it}}\right) \quad \forall i \in I, t \in T$$

$$\sum_i Z_{it} = 1 \quad \forall t \in T$$

where  $I_{it}$  denotes the amount of finished goods inventory of product  $i$  at the end of period  $t$ ,  $W_{it}$  the amount of WIP at the end of period  $t$ ,  $P_{it}$  the number of units of item  $i$  produced in period  $t$ , and  $Z_{it}$  the fraction of the period's total output in units of time allocated to producing item  $i$ . The formulation yields a convex optimization problem as long as the CF is concave. Using a piecewise linearized CF, the resulting model approximates the capacity constraint into a linear constraint, as shown below.

$$u_i P_{it} \leq Z_{it} \left( a^k \frac{\Lambda_{it}}{Z_{it}} + c^k \right) = a^k \Lambda_{it} + Z_{it} c^k \quad \forall i \in I, k \in K, t \in T$$

An important benefit of using clearing functions is that releases and lead times are jointly optimized. The shape of the clearing function defines the lead time requirements from workload conversion to the expected output. Asmundsson et al. (2009) give extensive discussions on CFs based modeling, and we refer to Asmundsson et al. (2009, 2006); Kacar (2012) for additional details. Kacar et al. (2011) study an iterative linear programming simulation algorithm to solve production planning problems and compare the results with a simulation model. The authors find that the CF-based models outperformed the conventional LP-based simulation models. Kacar et al. (2013) study CF-based models using wafer fabrication as a testbed and conclude that the clearing function model yields substantial improvements in profit over conventional linear programming models. These allocated clearing function-based models proposed by Asmundsson et al. (2009) outperform capacity-constrained static LP models (fixed lead time models) that do not consider queuing behavior. Recent work by Gopalswamy et al. (2019) suggests that directly fitting the piecewise linear functions using convex regression can yield considerably better results. See Missbauer and Uzsoy (2020b) and references therein. For additional resources

on the application of clearing functions, we refer the reader to Missbauer and Uzsoy (2020a).

Despite its advantages, there are some limitations and open questions on the application of CF in a production planning context. First, the shape of the CF depends on the mean and variance of the processing time, which relies heavily on the queuing approximation for the production resource. In a multistage production system, this also depends on the interarrival times of order releases from previous stations and processing decisions of upstream processes (Missbauer and Uzsoy 2020b). Theoretically, the queuing approximation is based on steady-state analysis, whereas in practice, we need a dynamic interpretation of changes in the queuing behavior in transient states. Fitting a CF using industrial data may not be straightforward. The same considerations and difficulties hold using industrial data similar to data from a simulation model. When used in production planning models to represent production, the performance of the planning models can be susceptible to parameter estimates. The primary approach simulates the production process over time for different releases, capturing the workload and output in each planning period. In this process, the workload of each item in each period depends on the specific releases used, which will affect the CF estimation. Kacar (2012) suggests that instead of fitting the CF to optimize fit to the data, fitting it to obtain the best performance from the production system yields promising results. Gopalswamy et al. (2019) propose a mixed-integer linear programming formulation to estimate a piecewise linear CF directly from the data.

### **2.3.2 Review of Stochastic Dynamic Programming**

Stochastic dynamic programming (SDP) deals with sequential decision problems in which the current period reward and the subsequent period state are random (Bertsekas 1995), i.e., with multistage or multi-period stochastic systems. The decision maker’s goal is to maximize the expected (discounted) reward over a given planning horizon (Ross 2014). Bellman (1957) uses the typical structure underlying sequential decision-making problems and dynamic programming (DP) to solve a wide range of sequential decision problems. In DP problems involving sequential decision-making over discrete periods, as in the production-inventory problems considered in this dissertation, two sets of variables associated with each planning period  $t$  define the total cost of the system: state variables and action variables. A policy is a mapping from each state to a feasible action. An optimal policy specifies the action to be taken in each state to minimize the objective function

over a given time horizon.

DP works based on the principle of optimality described in (Bellman 1957). It states, "An optimal policy has the property that whatever the initial state and the decision are, the remaining decisions must constitute an optimal policy concerning the state resulting from the first decision." Thus, an optimal policy can be determined recursively by solving subproblems for each state in each period. Hence dynamic programming is enumerative, requiring the specification of an optimal action for all possible system states in all planning periods. Computationally efficient dynamic programming algorithms are thus possible only if the sizes of the state and action spaces do not grow too rapidly with the size of the problem instance.

Stochastic DP addresses sequential decision problems in which a transition probability matrix governs the evolution of system states from one period to the next. We define the following notation to describe the DP approach. Let  $X$  denote the state space and  $A(x_t)$  the action space corresponding to each state  $x_t \in X$ . At each planning period,  $t \in 1, 2, \dots$  the sequence of events is as follows:

1. The current state,  $x_t \in X$  is observed
2. An action,  $a_t \in A(x_t)$  is selected
3. A reward,  $r(x_t, a_t)$  is earned
4. The system transitions to state  $x_{t+1}$  with probability  $p(x_{t+1} | x_t, a_t)$ ; note that the transition probability may depend on both the state  $x_t$  and the selected action  $a_t$ .

**Definition 2.3.1.** A policy  $\pi$  is a mapping from states to actions that specifies what action should be taken at each state in a given decision-making process. In some cases, the policy may be a simple function or lookup table, whereas, in others, it may involve extensive computation, such as a search process (Sutton and Barto 2018). An optimal policy  $\pi \in \Pi$  specifies the optimal actions for each state in each period, where  $\Pi$  denotes the set of all policies  $\pi$ .

In their most general form, stochastic dynamic programs use functional equations of the form (Bellman 1957)

$$f_t(x_t) = \max_{a_t \in A_t(x_t)} \left\{ r(x_t, a_t) + \gamma \sum_{x_{t+1}} P(x_{t+1} | x_t, a_t) f_{t+1}(x_{t+1}) \right\}$$

where  $f_t(x_t)$  is the maximum expected reward that can be attained over periods  $t, t+1, \dots, n$ , given the state  $x_t$  at the beginning of stage  $t$ .  $a_t$  belongs to the set  $A_t(x_t)$  of feasible actions at stage  $t$  given the initial state  $x_t$ ;  $\gamma$  is a non-negative discount factor strictly less than 1 accounting for the time value of rewards. The optimal policy can then be found by solving the Bellman equations backwards, starting from the terminal states and working backwards to the initial state.

### Finite horizon problem

In general, decision-making in inventory planning considers a finite number of periods in a finite horizon setting with the end-of-horizon impact on ending raw material, in-process inventory, and FG inventory. The terminal cost function denoted by  $V_{T+1}(x) = R_{T+1}(x)$  is used to specify the cost of reaching a terminal state  $x$  at the end of the horizon  $T$  where  $R_{T+1}(x)$  denotes the terminal reward. The terminal cost function plays a key role in deciding the optimal policy in a finite horizon setting. On the other hand, in an infinite horizon setting, there is no well-defined ending time or cost function. These models find an optimal solution at long-term or steady-state behavior of the system (Puterman 2014). Inventory problems are solved using both finite and infinite horizon settings; in particular, infinite horizon problems have been solved using discounted and average cost criteria (Howard 1960).

In this thesis, we solve a production planning problem to find the optimal planned release and production under a finite horizon setting with a fixed planning horizon of  $T$  periods. The expected reward gained through  $T$  periods in a finite horizon setting is given by

$$V_\pi(x) = \min_{\pi \in \Pi} \mathbb{E} \left\{ \sum_{t=1}^T r(x_t, a_t) \right\} \quad (2.9)$$

We want to find a policy  $\pi \in \Pi$  that minimizes (2.9). The horizon is defined for  $t = 1$  to  $T$ , where  $T$  is finite,  $x_t$  is the state process with initial state  $x_0$ , and  $a_t$  the action variable ( $x_t$  and  $a_t$  can be vectors),  $r(x_t, a_t)$  is a reward function defined for time periods  $t < T$  and  $R_T(x)$  is the terminal cost function,  $\gamma$  is the discount factor ( $0 < \gamma \leq 1$ ),  $A_t$  the set of all possible actions  $a_t$ , and the right side of (2.9) denotes the expected sum of all rewards over all periods  $t = 1$  to  $T$ . The DP model for the finite-horizon problem with discrete periods is given by

$$V_t(x_t) = \min_{a_t \in A(x_t)} \left\{ r_t(x_t, a_t) + \sum_{j \in X} P(j|x_t, a_t) V_{t+1}(j) \right\} \quad (2.10)$$

where  $V_t(x)$  is called the value function of state  $x$  at stage  $t$  with terminal cost function  $V_{T+1}(x) = R_{T+1}(x)$  and  $j$  is the next-stage state conditional on the current-stage state  $x_t$  and the action  $a_t$ . The optimal policy is the argmin of (2.10), which specifies an optimal action for every state  $x_t$  encountered over the planning horizon  $t = 1, \dots, T$ .

### Backward induction algorithm

The finite horizon stochastic dynamic programming algorithm uses a backward induction algorithm with a terminal cost to solve the sequential decision-making problem. We solve the Bellman equation from period  $T$  recursively until period  $t = 1$  with the initial boundary condition for the period  $T + 1$ . The collection of optimal actions for each state constitutes the optimal policy. The algorithm is as follows:

---

#### Algorithm 1 SDP for finite horizon problem

---

```

1: procedure MAIN LOOP
2:   Set  $t = T$ 
3:    $V_T(x_T) = R_T(x_T)$  for all  $x_N \in X$ 
4:   loop :
5:   Let  $t = t - 1$ , calculate  $V_t^*(x_t)$  for all  $x_t \in X$ 
6:    $V_t^*(x_t) \leftarrow \min_{a_t \in A(x_t)} \left\{ r_t(x_t, a_t) + \sum_{j \in X} P(j|x, a) V_{t+1}^*(j) \right\}$ 
7:    $A_t^*(x_t) \leftarrow \operatorname{argmin}_{a_t \in A(x_t)} \left\{ r_t(x_t, a_t) + \sum_{j \in X} P(j|x, a) V_{t+1}^*(j) \right\}$ 
8:   if  $t = 1$ ; stop. Otherwise, return to the loop
9: end procedure

```

---

The main drawback of SDP algorithms is the rapid growth of the number of state-action pairs. It takes much memory to store the calculated result of every subproblem, whether the stored value will be utilized or not. For a finite horizon multi-period problem of  $T$  periods with  $n$  states and  $m$  actions using a backward induction algorithm; in each iteration, we enumerate all possible values of actions for each state and evaluate the expected cost, giving  $\mathcal{O}(nmT)$  arithmetic operations. These large number of operations give rise to *curse of dimensionality*.

Furthermore, the expectation operator requires discrete states to calculate the optimal expected cost-to-go function. Thus the discretization of the state space affects the number of state-action pairs that must be considered in each period. The classical results (a base-stock policy for a classical multi-period problem without fixed costs and an  $(s, S)$  policy with fixed costs) emerge from proving structural results about the form of the optimal policy but usually do not allow the optimal policy to be computed efficiently. The

only known approach to obtaining exact solutions to multi-period stochastic inventory problems remains stochastic dynamic programming - which is subject to the curse of dimensionality.

It is worth noting that the inventory systems studied in many research papers have nice problem structures for their optimal policies. However, the structure of the optimal policies for capacitated stochastic production inventory models using the queuing behavior of production resources has yet to be fully characterized. Therefore, this dissertation develops an SDP algorithm for a single-stage single-item finite horizon capacitated production inventory problem subject to congestion due to the queuing effect of releases and in-process inventory under demand uncertainty. In the next section, we discuss approximate dynamic programming algorithms and value function approximations used to solve stochastic sequential decision-making problems relevant to stochastic production inventory problems.

### **2.3.3 Approximate Dynamic Programming**

Approximate dynamic programming (ADP) methods (Werbos 1987) are helpful when the problem size is too big for exact methods to work (Powell 2007). ADP methods are also useful when the transition probabilities are unknown, and the problem has many states and actions (Sutton and Barto 2018). ADP algorithms are classified as model-free and model-based algorithms (Keerthi and Ravindran 1994). Since ADP algorithms and reinforcement learning methods incorporate dynamic programming and the Bellman equation, ADP algorithms are sometimes referred to as modern reinforcement learning (RL) algorithms (Powell et al. 2022). The most commonly used RL algorithms are Monte Carlo policy iteration, temporal difference learning, Q-learning, and State Action Reward State Action (SARSA). These algorithms aim to solve large-scale stochastic optimization problems in a reasonable run time compared to exact methods. Recent works on RL algorithms, in conjunction with other machine learning algorithms have been used for solving inventory problems. RL-based models are outside the scope of this thesis. For additional details on RL-based and deep learning-based models, we direct the readers to Boute et al. (2021); Clausen and Li (2022); Giannoccaro and Pontrandolfo (2002); Kemmer et al. (2018); Oroojlooyjadid et al. (2020); Panzer and Bender (2021); Powell (2021, 2022); Sui et al. (2010); Sutton and Barto (2018).

ADP methods use value function approximation methods (such as regression and

neural networks) to estimate the expected cost-to-go function for large state spaces (Das et al. 1999). Powell (2014) discusses three classes of approximating functions: look-up tables, parametric (linear regression, neural network, Fourier basis), and non-parametric (for approximating convex functions using piecewise linear, separable approximations, and multi-dimensional Benders cuts). VFA is needed in any of the following cases described by Bertsekas et al. (2011); Brunskill (2019); Cai (2009); Ravindran (2017); Silver (2015): 1) Memory constraints due to large state spaces, 2) Memory outrun due to multiple dimensions in the state (e.g., Fleet optimization problem by Powell (2021) where state space has 25 dimensions), 3) Continuous states that do not match directly the discrete states in the state space definition, 4) Need for a parameterized representation to make the model learn from the actual data (one way of solving this VFA is by Monte Carlo methods where we use a sample path to find the reward and then create a data set to find the parameterized value function by replacing the actual value function) and 5) Need for a generalization method to calculate a complex expected cost to-go function. This generalization is essential in ADP because certain data-driven VFA methods like neural networks (NN) or regression methods use mean square error as a performance measure to find the optimal weights (in the case of NN) or coefficients (in the case of regression methods). We need the sample path containing the states and corresponding rewards to be noted to train the parameterized models. So VFA methods are selected carefully, whether it is based on any of the following methods: 1) A parameterized method using a machine learning model, 2) Finding a state aggregation, 3) Finding an interpolation using nearest neighbors of the states, and 4) An interpolation technique for approximating the functions with fine grid size. The ADP model for the finite-horizon problem with discrete periods is given below,

$$\bar{V}_t(x_t) = \min_{a_t \in A(x_t)} \left\{ r_t(x_t, a_t) + \sum_{j \in X} P(j|x_t, a_t) \bar{V}_{t+1}(j) \right\} \quad (2.11)$$

where  $\bar{V}_t(x)$  is called the value function of state  $x$  at stage  $t$  with the terminal cost function  $\bar{V}_{T+1}(x) = R_{T+1}(x)$ ,  $j$  is the next-stage state conditional on the current-stage state  $x_t$  and the action  $a_t$ . Assuming we come up with a reasonable approximation (for instance, a linear approximation method)  $\bar{V}_{t+1}(j)$  to find the expected cost-to-go function,

the optimal policy is given by

$$A_t(x_t) = \operatorname{argmin}_{a_t \in A(x_t)} \left\{ r_t(x_t, a_t) + \sum_{j \in X} P(j|x_t, a_t) \bar{V}_{t+1}(j) \right\} \quad (2.12)$$

which specifies an optimal action for every state  $x_t$  encountered over the planning horizon  $t = 1, \dots, T$ . By definition, (2.12) should yield an approximately optimal policy.

In this dissertation, we use bilinear interpolation, considering the neighboring states (defined by discretizing the continuous state to discrete grids), to evaluate the cost for a particular state that is not part of the discretized state space (Bertsekas 2012, 1995; Bertsekas et al. 2011). The discretization of the continuous states allows the DP algorithm to calculate the expected cost-to-go functions, and find an optimal policy for the given problem under study. The optimal policy for the original continuous problem is calculated using the extended approximate cost function through some form of interpolation, such as the bilinear interpolation defined in Hadley (1965). Other interpolation methods have been proposed by several authors in the numerical computation of DP algorithms. See Cai and Judd (2013, 2015); Judd and Judd (1998) and references therein. Value function approximation can significantly reduce the computational complexity of solving a Bellman equation, especially in large state spaces. However, the approximation introduces a degree of error into the solution, which may affect the quality of the optimal policy found. Therefore, the choice of approximation given by  $\bar{V}_{t+1}(x)$  is critical to ensure accurate and efficient approximation of the value function. We characterize bilinear interpolation as an approximation, i.e., when the grid size in the discretization gets smaller, the expected cost computed from the DP approach is the exact value of the true cost. However, this leads to an increased computational burden. Therefore the trade-off is between the computation run time versus the accuracy of the value function.

The main reason for VFA is the continuous state space (fractional ending FG inventory and WIP) given by the CF-based replenishment; thus, VFA is necessary to find the expected cost-to-go functions in the Bellman equation. Unlike backward approximate dynamic programming, where we work with a sampling of states  $x$ , in this work, we enumerate all the states to establish the optimal policy for each state. The system becomes complex because of the multi-dimensional state space  $(X_{t-1}, W_{t-1})$  and actions based on both release and production variables. We use a lookup table for the value function approximations, thus implementing a classical backward dynamic programming approach

that provides an approximate solution whose accuracy depends on the discretization of the state space. We explain the details of the VFA used in Chapter 4 and discuss the impact of the expected cost through a list of computational experiments.

## 2.4 Methods used in PP and Inventory models

The stochastic dynamic programming algorithm gives an exact solution for the capacitated production inventory problem subject to congestion and demand uncertainty, governed by a CF. However, SDP suffers from the curse of dimensionality due to the large number of state-action pairs that must be evaluated. Several authors have developed solution methods using stochastic programming, robust optimization, simulation-based optimization, and chance-constrained models (Albey et al. 2019; Aouam and Uzsoy 2012a, 2015; Irdem et al. 2008; Kacar et al. 2011; Lin and Uzsoy 2016; Norouzi 2013; Ziarnetzky et al. 2020). While these methods show promising results, the difficulty of finding an exact solution using SDP has prevented the rigorous assessment of solution quality and the optimality gap of these methods. Therefore, we now discuss stochastic optimization methods to solve capacitated production inventory problems under stochastic demand.

### 2.4.1 Chance constrained programming

Charnes and Cooper (1959) introduced chance-constrained (CC) programming as a tool to solve optimization problems under uncertainty. They create a deterministic equivalent of the stochastic optimization problem by allowing certain constraints to be violated with a specified probability (Bookbinder and Tan 1988; Johnson et al. 1974; Tarim and Kingsman 2004). Blau (1974) described the limitations of chance-constrained mathematical programming formulations, showing an example where the expected value of perfect information is less than the expected value of sample information. Without proper knowledge of the demand distributions, the formulation becomes challenging to solve, and may become non-convex unless the distributions satisfy some assumptions. Despite its limitations (mainly associated with structure, convexity, and stability), the CC formulation leads to a convex linear programming model under specific assumptions; see Houda (2007); Lejeune and Prékopa (2018); Prékopa et al. (2011); Wets (1998), that is intuitive to explain and easy to interpret under appropriate conditions.

Several authors have used CC to address production planning problems under demand uncertainty (Aouam and Uzsoy 2012a, 2015; Bookbinder and Tan 1988; Norouzi

2013; Tarim and Kingsman 2004). Ravindran et al. (2011) proposed a PP model with chance constraints considering workload-dependent lead times using clearing functions. Their model assumes a policy that maintains inventory position at a percentile of the lead time demand and treats the lead times used to establish inventory levels as an exogenous parameter. The authors compare different CC models to model the production planning problems under congestion that the effects of capacity loading on WIP and lead times, and the contribution of WIP towards safety stock and non-stationary stochastic demands. These models achieve a good tradeoff between cost and customer service compared to other mathematical models. Aouam and Uzsoy (2012a) developed CC-based heuristic solutions for production planning in the face of stochastic demand and workload-dependent lead times. Aouam and Uzsoy (2012b) examine a multistage stochastic programming (MSP) model and a two-stage stochastic program (2SP) and compare the solution with CC models, including Dynamic Inventory Position (DYNIP) and Zero Order Inventory Position (ZOIP) proposed by Ravindran et al. (2011). Results indicate that the planning procedures with recourse (MSP, DYNIP) consistently outperform those without recourse (2SP, ZOIP). Under the right parameterization the DYNIP model can compute near-optimal solutions in a reasonable CPU time. Albey et al. (2016) developed chance-constrained models in conjunction with simulation optimization to solve multiple capacity allocation problems.

Norouzi (2013) presented a shortfall-based chance-constrained formulation using approximations proposed by Glasserman (1997) for capacitated inventory models. The author assumes the system operates under a base-stock policy, with base-stock level  $S_t$  and defines the shortfall with initial FG inventory  $X_{t-1}$  and production as:

$$Y_t = \max[S_t - X_{t-1} - P_t, 0]$$

This model assumes independent demand in each period and derives a chance constraint whose satisfaction will maintain a specified probability of no stockouts, and extends to stochastic demand with dynamic and correlated demands. Norouzi (2013) finds the shortfall-based CC model outperforms the earlier versions suggested by Ravindran et al. (2011) and discussed by Aouam and Uzsoy (2012a).

Lin and Uzsoy (2016) develop two chance-constraint-based production planning models with stochastic demand implemented in a rolling horizon environment. They extend the shortfall-based formulation proposed by Norouzi (2013) and find that the formula-

tions reduce planned release changes using shortfall-based CC without affecting the service level and improving cost. Ziarnetzky et al. (2018) apply the shortfall-based chance constraint formulations to solve a rolling horizon problem under the Martingale Model of Forecast Evolution (MMFE) (Heath and Jackson 1994). The results indicate that considering forecast evolution in production planning models can improve performance by exploiting the advanced demand information provided by the forecast updates. Similarly, Ziarnetzky et al. (2020) study the CC-based models for multi-period, multi-product production planning models in a rolling horizon planning context. They find that the rolling horizon models yield promising results and extend for future developments.

### 2.4.2 Robust optimization

Robust optimization typically involves formulating mathematical optimization problems that consider uncertain parameters. Unlike stochastic programming models and chance-constrained models, which incorporate probability distributions of the uncertain parameters, robust optimization models consider possible realizations of the uncertain parameters in an uncertainty set and then optimize against worst-case realizations within this set (Bertsimas et al. 2018). Stochastic programming and SDP methods assume complete knowledge of the probability density function to cover the randomness present in stochastic optimization problems. However, RO methods assume that the uncertain data resides in an uncertainty set (Gorissen et al. 2015) and develop solutions to protect against a worst-case scenario that can be defined differently. With the recent development in solver technology, RO methods are quite popular in solving stochastic optimization problems. Bertsimas and Sim (2004) propose an approach to solving linear programming problems under uncertainty. By adjusting the level of conservatism of the robust solutions in probabilistic bounds of constraint violations, the RO approach solves discrete optimization problems, and the resultant problem is also a linear programming problem.

Ben-Tal and Nemirovski (2002) survey the main results of RO models that combine the computational tools with process optimization problems in which the data are uncertain and are only known to belong to some uncertainty set. Bertsimas et al. (2011) recently survey the theory and applications of RO for multistage decision-making problems. They share various uncertainty sets used in RO formulations for any decision-making problems under uncertainty. We direct the reader to the tutorial on robust and data-driven optimization under uncertainty by Bertsimas and Thiele (2006a).

Bertsimas and Thiele (2006b) applied RO to solve inventory problems. They used a cardinality-constrained uncertainty set and showed that the robust problem has an optimal solution in the form of an  $(s, S)$  policy. See and Sim (2010) developed a RO approach to address multi-period inventory control problems under ambiguous demands (limited information on mean, std. dev.). Unlike a dynamic programming algorithm, a tractable deterministic optimization problem is solved in polynomial run time using a second-order conic program. Computational results suggest that this method performs better than DP-based heuristic approaches. Qiu et al. (2017) formulated the finite horizon single-product periodic review inventory problem as a DP. The authors transformed the problem into a second-order cone program using box and ellipsoid uncertainty sets. The authors proved the existence of optimal  $(s, S)$  policy structure for robust DP approaches and illustrated the effectiveness and practicality of the approach.

In the case of production inventory problems subject to congestion, Aouam and Uzsoy (2015) compare the performance of the stochastic programming model with CC models and RO models suggested by Bertsimas and Thiele (2006b). Given their model’s different assumptions to address uncertainty, the models behave quite differently, and RO models perform better than scenario-based stochastic programming. i.e., the solution quality is completely determined by the level of uncertainty set of the random variables. Li et al. (2011) discuss the RO models in a chemical production planning setup and find the RO models perform better than some of the robust mixed-integer linear programming models (Li and Floudas 2014). Albey et al. (2019) presented a distributionally robust release planning model that allows planned lead time probability estimates to vary over a specified ambiguity set. The authors evaluate the performance of non-robust and robust approaches using a simulation model of a scaled-down wafer fabrication facility and found that the RO model produces good results with the highest uncertainty level.

### 2.4.3 Stochastic programming

Earlier studies on the application of stochastic programming (SP) date back to Dantzig (1955). Similarly, Ermol’ev and Mirzakhmedov (1976) developed SP models for various production and inventory problems. The purpose of these models is to convert a stochastic optimization problem into a deterministic equivalent problem. Two approaches are proposed to solve stochastic optimization problems (Birge and Louveaux 2011) using SP (Prekopa 1973): 1) Recourse models, and 2) Probabilistic constrained models.

Recourse models divide the problem into two stages. A deterministic problem is solved in the first stage, yielding one decision (adoptive variables). In the subsequent stage, we solve a stochastic problem that minimizes the expected cost (maximize rewards) of the consequences of that decision (anticipative variables). Recourse models (Birge 1997) are further classified into two types: 1) Two-stage stochastic programming (2S-SP) and 2) Multistage stochastic programming (MS-SP). A complete description of SP formulation and methods are discussed in Birge and Louveaux (2011); Prékopa (2013).

Probabilistic-constrained models are a combination of recourse models and probabilistic constrained (chance-constraints) SP models (Prékopa et al. 1998; Shapiro et al. 2021). We discussed chance-constrained models in the previous section relevant to our work, and combination models are outside the scope of this thesis. For additional details, we direct the reader to Luedtke et al. (2010).

Higle and Kempf (2010) introduced an SP model for production planning under uncertainty. This uncertainty model extends to supply via uncertainties in the production process and demand via probabilistic descriptors of quantities and due dates even after orders have been received. The resulting model is a stochastic linear program incorporating Markov chains within the probabilistic models. Aouam and Uzsoy (2012a) applied chance-constrained programming models to a single-stage multi-period capacitated production inventory problem replenished by a CF. The author used two-stage and multistage SP (MSP) models as a benchmark to compare CC solutions and discussed the results using an experimental design. Results suggest that MSP models outperform CC models, although with high computation time; the main reason for high computation time is the large number of scenarios generated to approximate the true distribution of the random variables.

While the CC models provide a reasonable approximation allowing violation of the probabilistic constraint with a prespecified  $\alpha$ , SP models rely on the scenarios generated from the random variables, and multi-stage models require a large number of scenarios for a good solution quality. MS-SP is considered the best available approach for an exact solution other than the SDP and MDP formulations. However, the scalability of this approach in generating scenarios and recourse strategy is a major concern (Shapiro 2006). In the past few decades, several studies have determined how many scenarios need to be considered to provide a reasonably good solution. The number of scenarios necessary for a stochastic programming problem depends on the complexity of the problem, the

level of uncertainty involved, and the desired accuracy of the solution. In general, a larger number of scenarios will provide a more accurate representation of the underlying probability distribution, and lead to more accurate solutions, but increase computational complexity (Birge and Louveaux 2011; Prékopa 2013; Shapiro et al. 2021).

Aouam and Uzsoy (2015) compared two-stage SP formulation with the CC and RO models. Results indicate that the RO method yields promising results, but different models face challenging issues and difficulty addressing this problem. Careful attention is needed while defining the uncertainty sets in the case of RO and generating multiple scenarios that describe the demand uncertainty and explicit holding and backorder costs in the case of two-stage SP. In both cases, an exact solution is not available to compare the solution quality of SP models, which is addressed in this dissertation using a stochastic dynamic programming algorithm.

#### **2.4.4 Simulation-based approaches**

Simulation based optimization (simopt) algorithms are popular in solving optimization problems under uncertainty. Fu (1994) describe simulation based optimization approaches to solve problems in the field of operations research and management science (Fu 2002; Fu et al. 2015). Several authors have used simulation models to study the production inventory system due to the stochastic behavior of production resources and the non-linear relationship between the cycle time and resource utilization. Their high computational burden has limited their application as a stand-alone method. While modeling a large production facility, simulation-based methods require many time-consuming simulation runs and a large number of iterations yielding a local optimum.

Despite its limitations, simulation models help derive insights into optimization problems with uncertainty. Liu et al. (2011) developed a genetic algorithm (GA) based multi-objective simulation optimization model to solve a production planning model in a wafer fabrication setup. Albey and Uzsoy (2015) develop two mathematical models to represent the dependency of workload and release times. They compared the solution with a reference solution using a gradient-based simulation optimization algorithm and found three models that yielded good results. Albey et al. (2016) proposed a GA-based capacity allocation to solve a multi-item multi-period production inventory problem in the face of stochastic demand. Computational results reveal that the proposed model shows promising results. Li et al. (2016) implement simulation optimization using a pre-fitted

meta-model, avoiding the need for time-consuming simulation replications during the optimization process. Ziarnetzky and Mönch (2016) present a PP model based on the clearing function. The optimal release schedules are assessed using a discrete event simulation, and the production resource utilization and capacity decisions are evaluated using simulated annealing. Results indicate an increase in the profit without violating the cycle time. Zhang et al. (2022) import a theoretical solution from the mathematical planning model into the discrete event simulation model to perform simulation iterative optimization. The authors devised two strategies to reflect the physical production system data to update production plans. Experimental results show that the proposed method saves at least 26% of the production cost compared with the load-independent lead time-based model with discrete event simulation technology.

### 2.4.5 Need for a benchmark algorithm

Stochastic optimization methods such as RO, CC, SP, and simulation optimization have significantly impacted solving capacitated production inventory problems subject to congestion. Bertsimas and Thiele (2006b) propose a general methodology based on robust optimization to address the problem of optimally controlling a supply chain subject to stochastic demand in discrete time. The authors pointed out that dynamic programming, which suffers from the curse of dimensionality, assumes complete knowledge of the demand distribution. In contrast, the RO method considers the uncertainty of the demand without assuming a specific distribution. Another significant benefit of using the RO method is that this method yields tractable solutions, especially when compared to the multi-dimensional dynamic programming method. Despite these benefits, the performance of these models depends heavily on the accurate specification of the uncertainty sets used and the uncertainty budget.

SP models depend heavily on selecting and weighting the discrete scenarios used to meet stochastic demand. They also lead to large formulations, especially when implemented in a multi-stage setting with recourse (Aouam and Uzsoy 2012b). Therefore, stochastic programming methods do not scale to large problem instances. SP models may result in solutions that are difficult to interpret because they depend on probabilities and scenarios. Additionally, the complexity of SP models increases significantly with the number of stages and scenarios, and computational methods such as dynamic programming and stochastic decomposition may be required to solve them efficiently to find the exact solution.

CC programming models are easy to implement, scalable to large instances, easy to explain to production managers, and solved via commercial solvers without the need for tailored algorithms. Aouam and Uzsoy (2012a, 2015) compare the performance of CC models with SP and RO methods in a single-stage, single-item production planning problem with workload-dependent lead times. They find that the CC models provide comparable solutions in quality to those from the other formulations. In addition, based on the analysis given by Aouam and Uzsoy (2012a), the CC and SP models make quite different assumptions in formulating the models. For instance, in SP modeling, the decision maker can assign probabilities to different scenarios and can choose a decision that optimizes some objective function across these scenarios. However, in CC modeling, the decision maker wants to ensure that the probability of violating some constraint(s) does not exceed a certain level denoted by  $\alpha$ . While these models' behavior is well documented, it is assumed that they perform well when appropriately formulated and parameterized, which is an open research question. Chance-constrained models may result in conservative solutions that overestimate the risk of violating constraints, leading to suboptimal decisions. Finally, simulation optimization approaches have been used in various problem domains, but their computational burden has limited their application to production planning problems.

## 2.5 Conclusions

This chapter gives a brief literature review of production planning models and solution methods. In this thesis, we solve capacitated production inventory problems under congestion subject to uncertain demand. We start with motivation, classical inventory models, newsvendor models, and their extensions in a broad sense. We then discuss the optimal policies and structures of the optimal policies in single-echelon and multi-echelon systems. We then review exact solution methods using stochastic dynamic programming and approximate dynamic programming algorithms and motivate the need for function approximations. We discuss approaches to finding order releases and production quantities based on a non-linear clearing function-based replenishment that gives the queuing behavior of production resources. We briefly share the relevant literature using chance-constrained models, robust optimization, stochastic programming, and simulation-based optimization models. Finally, we briefly discuss their relative advantages and disadvantages and why we choose to proceed with SDP.

Based on the discussions, exact solutions to these complex problems are computa-

tionally burdensome to obtain. Other stochastic approximation methods like CC, SP, RO, and simulation-based procedures affect solution quality and convergence bounds. We need an exact solution to benchmark the approximation methods using CC, RO, SP, and simulation optimization-based methods. The only solution method to get an exact solution for production inventory models is stochastic dynamic programming which suffers from the curse of dimensionality.

Therefore, we first develop a single-stage, single-item, single-period mathematical program to find the optimal releases and production. Using the Karush Kuhn Tucker conditions, we give the optimal conditions and propose optimal solution characterization using three cases of release and production. We then develop a single-stage multi-period finite horizon solution using a stochastic dynamic program as an exact solution and discuss the policy structure in terms of cost and clearing function parameters. We use value function approximation and discretize both state and action space such that the optimal policy will act as a benchmark that can be helpful to compare the solution quality and optimality gap with other approximation methods. We study the single period and multiple periods problem using an extensive test bed and discuss the summary of the principal results. Finally, we present some extensions and future directions toward completing the dissertation using modern reinforcement learning algorithms.

# Chapter 3

## Single-Stage Zero Lead time Problem

### 3.1 Introduction

This chapter considers the single-stage, single-item, zero lead time, capacitated stochastic production inventory problem (SPP) with replenishment governed by a concave, non-decreasing clearing function. We assume there is no time lag in delivering finished products from production resources to finished goods inventory. Hence the zero lead time in the single-period production inventory problem. A single stage represents a processing activity that converts raw materials into finished goods. There are two types of inventory present in this single-stage production system. At the start of the period, raw materials are released into the system for processing. Any unprocessed raw material remains in the system at the end of the period as ending work-in-process inventory. We store the output of the production stage as finished goods inventory that can be used to meet customer demand. In this chapter, we consider a single-period problem where costs are incurred, and uncertainty is realized at the end of the period. Assuming unlimited raw material availability, the production stage has a finite capacity  $P_{max}$  and let  $D$  represent the stochastic demand with mean  $\mu$  and standard deviation  $\sigma$  where  $\mu < P_{max}$ . At the beginning of the period, we observe the finished goods (FG) and the work-in-process (WIP) inventory. Given the initial levels of FG and WIP inventories, the SPP seeks to determine the production and release quantities that minimize the sum of expected WIP and finished goods inventory holding costs, shortage costs, release costs, and production

costs. The concave non-linear CF governing output differs significantly from conventional capacitated and uncapacitated newsvendor-based production inventory problems. Figure 3.1 shows a schematic representation of the single-stage production inventory system.

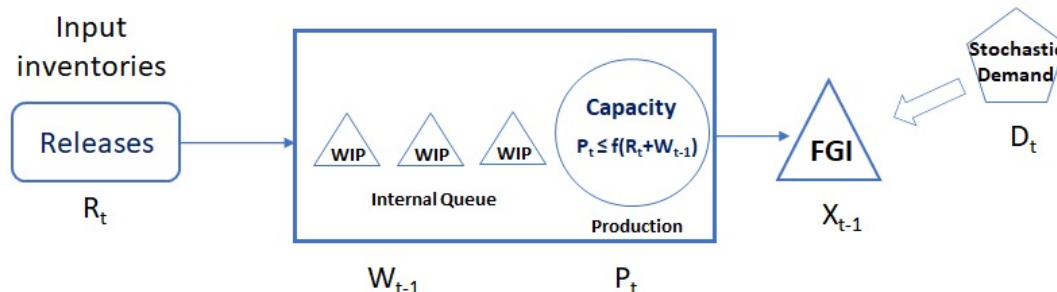


Figure 3.1: Overview of a single-stage single-item production system

## 3.2 Model assumptions

At the beginning of the period, the system's state, denoted by  $(X_0, W_0)$ , is described by the finished goods (FG) inventory level  $X_0$  and the work in process (WIP) inventory level  $W_0$ . Let  $X$  and  $W$  denote the finished goods and work-in-process inventory levels at the end of the period. A negative value of  $X$  represents a backorder due to excess demand in a period. We seek a general solution to find the optimal production and release quantities for any initial state  $(X_0, W_0)$ . Production  $P$  and releases  $R$  are determined before the uncertain demand is observed. Demand is stochastic and follows a given probability distribution function. Only one order can be placed in every period. i.e.,  $R$  units of material are released into the system, and optimal production takes place with the updated workload  $(R + W_0)$  using a clearing function.

The details of the CF-based replenishment and workload are as follows. We define the workload, the amount of work made available to the resource at the beginning of the period, as  $\Lambda = W_0 + R$ . A workload-based CF  $f(\Lambda)$  assumes the expected output  $P$  of the production resource cannot exceed the available workload  $\Lambda$ , implying  $\frac{\partial f}{\partial \Lambda} \leq 1$  for all  $\Lambda \geq 0$ , and hence  $P \leq \Lambda$  (Missbauer and Uzsoy 2020b). We assume  $f(0) = 0$  and that production is bounded above by the production capacity  $P_{max}$ , where  $\lim_{\Lambda \rightarrow +\infty} f(\Lambda) = P_{max}$ . Excess inventory at the end of the period incurs holding costs for FG and WIP, and stockouts are back-ordered. All holding and backorder costs are linear, and there are no fixed ordering costs. We define the following notation.

## Parameters

$h$	Unit finished goods holding cost
$b$	Unit backorder cost
$C^p$	Unit production cost
$C^w$	Unit work in process (WIP) holding cost
$C^r$	Unit release (raw material) cost
$D$	Non-negative random variable representing demand
$\phi(\cdot)$	Probability density function of demand $D$
$\Phi(\cdot)$	Cumulative distribution function of demand $D$
$X_0$	Initial FG inventory
$W_0$	Initial WIP

## Decision variables

$R$	Raw material release quantity at the beginning of period
$P$	Production quantity, governed by clearing function (CF), $P \leq f(W_0 + R)$

## Clearing function parameters

$P_{max}$	Maximum production capacity such that $\lim_{\Lambda \rightarrow \infty} f(\Lambda) = P_{max}$
$\Lambda$	Workload, given by $\Lambda = W_0 + R$
$f(\Lambda)$	Non-decreasing concave clearing function

The expected cost of the system for a given initial state  $(X_0, W_0)$  with release  $R$  and production  $P$  is  $\Theta(X_0, W_0, R, P)$ . The following assumptions are made regarding costs.

- $b > h$ : Stocking out is more expensive than holding extra inventory
- $b > C^p + C^r$ : Otherwise, it is optimal not to release or produce in the face of unmet demand
- $b > C^w$ : Ensures it is never optimal not to produce in the face of unmet demand when WIP is available
- $h > C^w$ : We consider value addition from WIP to the FG. Hence, FGI holding cost is higher than WIP holding cost
- The production process is deterministic and described by a workload-based non-decreasing concave clearing function, leading to the constraint  $P \leq f(W_0 + R)$

instead of  $P = f(W_0 + R)$ , which leads to a non-convex feasible region (Merchant and Nemhauser 1978). Since releases are also deterministic, ending WIP is deterministic.

The sequence of events is as follows:

1. At the start of the period, current FG inventory  $X_0$  and WIP  $W_0$  are observed.
2.  $R \geq 0$  units of material are released into the system, instantaneously raising the workload level to  $W_0 + R$ .
3. Production of  $P \leq f(W_0 + R)$  units are ordered and instantaneously received, raising the finished goods inventory level to  $X_0 + P$  by reducing the workload.
4. The uncertain demand  $\mathbf{D}$  is observed and fulfilled from the finished goods inventory as much as possible.
5. The ending state of the system is determined by the material balance equations

$$X = X_0 + P - \mathbf{D}$$

$$W = W_0 + R - P$$

$W$  is deterministic, whereas  $X$  is a random variable because the demand  $D$  is a random variable.

6. Costs of release, production, FG holding and backlogging, and WIP holding are charged at the end of the period.

The single-period production inventory problem can be stated as follows:

$$\min_{R, P \geq 0} \Theta(X_0, W_0, R, P) = g(X_0 + P) + C^r R + C^w(W_0 + R - P) + C^p P \quad (3.1a)$$

$$\mathbf{s.t.} \quad W_0 + R - P \geq 0 \quad (3.1b)$$

$$P \leq f(W_0 + R) \quad (3.1c)$$

$$R, P \geq 0 \quad (3.1d)$$

where

$$g(X_0 + P) = h \int_0^{X_0 + P} (X_0 + P - \mathbf{D}) \phi(\mathbf{D}) d\mathbf{D} + b \int_{X_0 + P}^{\infty} (\mathbf{D} - X_0 - P) \phi(\mathbf{D}) d\mathbf{D} \quad (3.2)$$

represents the newsvendor equation with holding cost  $h$  and backorder cost  $b$ . The objective function (3.1a) minimizes the sum of the expected cost of release (raw materials), ending work in progress, cost of production, and the newsvendor objective function based on the ending FGI. Constraint (3.1b) requires the ending WIP to be non-negative, and (3.1c) limits the production output by the CF based on the workload ( $W_0 + R$ ). By assumption, (3.1b) is redundant due to the concave non-decreasing shape of the CF with  $\frac{\partial f}{\partial \lambda} \leq 1$  at the origin. Therefore, the problem can be written as

$$\min_{R, P \geq 0} \Theta(X_0, W_0, R, P) = g(X_0 + P) + C^r R + C^w(W_0 + R - P) + C^p P \quad (3.3a)$$

$$\text{s.t.} \quad P \leq f(W_0 + R) \quad (3.3b)$$

$$R, P \geq 0 \quad (3.3c)$$

In the case of a CF-based production inventory model, the production is governed by a clearing function that captures the queuing behavior in the production system. i.e.,  $0 \leq P \leq f(R + W_0)$  and the optimal release is given by  $R^* = \max\{f^{-1}(P^*) - W_0, 0\}$ . This is different from the capacitated production inventory (CPI) problem without congestion shown in (3.4), where production is bounded by the maximum production capacity  $P_{max}$ , i.e.,  $P^* \leq P_{max}$  but there is no relation between workload and production. In addition, the CPI model assumes that the initial WIP behaves like raw material that is consumed by the production resource to produce  $P^*$ . Any initial WIP is present at the beginning of the period is consumed first before releasing any additional raw material to the production resource, so the optimal release depends on the initial WIP. Under the given cost assumptions, when  $W_0 = 0$ ,  $R^* = P^*$ . However, when  $W_0 > 0$ , we have two cases: 1) if  $P^* > W_0$ , then  $R^* = P^* - W_0$ , where  $P^* \leq P_{max}$ . In other words, we get  $P^* = \min\{P_{max}, R^* + W_0\}$ . Otherwise, 2) when  $P^* < W_0$ ,  $R^* = 0$  and the ending WIP is calculated as  $W = W_0 - P^*$ . The expected cost is calculated by the objective function shown in (3.4)a.

$$\min_{R, P \geq 0} \Theta(X_0, W_0, R, P) = g(X_0 + P) + C^r R + C^w(W_0 + R - P) + C^p P \quad (3.4a)$$

$$\text{s.t.} \quad P \leq P_{max} \quad (3.4b)$$

$$R, P \geq 0 \quad (3.4c)$$

When there is unlimited production capacity, we have an uncapacitated production inventory (UCPI) problem, as shown in (3.5). Therefore, similar to the CPI model, when  $W_0 = 0$ ,  $R^* = P^*$  for the UCPI model. However, when  $W_0 > 0$ , the optimal release is cal-

culated depending upon  $P^*$  with respect to  $W_0$ . i.e., when  $P^* > W_0$ , then  $R^* = P^* - W_0$ . Otherwise,  $R^* = 0$  and the ending WIP is calculated by  $W = W_0 - P^*$ . The expected cost is calculated by the objective function given in (3.5)a.

$$\min_{R, P \geq 0} \Theta(X_0, W_0, R, P) = g(X_0 + P) + C^r R + C^w(W_0 + R - P) + C^p P \quad (3.5a)$$

$$\text{s.t.} \quad R, P \geq 0 \quad (3.5b)$$

From (3.5), when  $W_0, C^r, C^w = 0$ , the UCPI model becomes a classical newsvendor model; the optimal production is given by  $\max\{0, S^* - X_0\}$ , when  $X_0 \leq S^*$ . Otherwise,  $P^* = 0$  when  $X_0 > S^*$ . The order upto level  $S^*$  is calculated as  $\Phi^{-1}(\alpha)$ , where  $\alpha = \left(\frac{b-C^p}{b+h}\right)$  represents critical fractile and  $\Phi^{-1}(\cdot)$  denotes the cumulative demand distribution. For the classical newsvendor models, with and without fixed costs, we direct the reader to Zipkin (2000) and references there.

### 3.3 Convexity conditions

**Lemma 3.3.1.** 1.  $g(X_0 + P)$  and the objective function are convex in  $R$ , and  $P$ .

2.  $P \leq f(W_0 + R)$  is a convex feasible region in  $R, P$ .

3.  $\Theta(X_0, W_0, R, P)$  is convex in  $R, P$ .

*Proof.* 1. The objective function terms  $C^p P, C^r R, C^w(R - P)$  are linear. Since  $h$  and  $b > 0$ , the second derivative of  $h \mathbb{E}[X_0 + P - \mathbf{D}]^+ + b \mathbb{E}[\mathbf{D} - X_0 - P]^+$  is  $(h + b)\phi(X_0 + P) \geq 0$ . Thus,  $g(\cdot)$  is convex in  $X_0 + P$ . Since the sum of convex functions is convex,  $\Theta(R, P)$  is convex.

2. The constraint takes the form  $P - f(W_0 + R) \leq 0$ . Since  $f(\cdot)$  is concave,  $-f(\cdot)$  will be convex in  $R$ ; Since  $P$  is linear, this constraint is convex in both  $P$  and  $R$  (Bazaraa et al. 2013).

3. For a convex optimization problem, any local minimum is a global minimum. Since the objective function  $\Theta(R, P)$  and the feasible region defined by the constraint,  $P \leq f(\Lambda)$  are convex in the range  $R, P \geq 0$ , as long as the feasible region is nonempty, the optimal solution to (3.3) yields global minimum. ■

### 3.4 Optimal solution structure

Intuitively, the optimal solution for (3.3) must consider three cases:

- **Case 1:** The initial FGI  $X_0$  is insufficient to meet the external demand, and the initial WIP  $W_0$  does not provide sufficient workload for production to raise the finished inventory to the optimal level. Raw material must be released to raise the workload to the appropriate level. Production must then take place from the workload to replenish the finished inventory to the optimal level. Thus,  $R^* > 0$ ,  $P^* > 0$ .
- **Case 2:** Initial FGI  $X_0$  is insufficient to meet the demand at minimum cost. However, the existing WIP  $W_0$  is sufficient to support the production needed to raise the FGI to the necessary level. Thus,  $R^* = 0$ ,  $P^* > 0$ .
- **Case 3:** We have sufficient FGI to meet the external demand. Therefore, neither releases nor production are necessary, and hence  $R^* = P^* = 0$ .
- Under our cost assumptions, it is never optimal to raise the workload through releases without also producing to raise the FGI; hence  $R^* > 0$ ,  $P^* = 0$  is never optimal.
- If  $W_0 > 0$ , the initial WIP is consumed first before adding any additional release to the workload to avoid excess release cost.

We now compute the optimal releases and production using the Karush Kuhn Tucker (KKT) conditions for the nonlinear program (3.3a) - (3.3c). We have a convex objective and a convex constraint; therefore, any solution satisfying the KKT conditions will be a global minimum. Defining a Lagrange multiplier  $\lambda$  associated with (3.3b), we obtain the Lagrangian function

$$L(R, P, \lambda) = \min_{R \geq 0, P \geq 0} \left\{ C^r R + C^w (W_0 + R - P) + C^p P + g(X_0 + P) - \lambda [P - f(W_0 + R)] \right\} \quad (3.6)$$

Taking partial derivatives with respect to  $R$  and  $P$  gives

$$\frac{\partial L(\cdot)}{\partial R} \geq 0 \implies C^r + C^w + \lambda \frac{\partial f(W_0 + R)}{\partial R} \geq 0 \quad (3.7)$$

$$\frac{\partial L(\cdot)}{\partial P} \geq 0 \implies C^p - C^w + (h + b)\Phi(X_0 + P) - b - \lambda \geq 0 \quad (3.8)$$

**Feasibility conditions:**

The CF constraint sets the feasibility requirement as

$$P \leq f(W_0 + R)$$

**Complementary slackness:**

The idea is to determine whether the optimum is at a boundary point of the feasible region given by the CF, or an extreme point of the objective function.

$$\lambda[P - f(W_0 + R)] = 0 \quad (3.9)$$

$$R \left[ C^r + C^w + \lambda \frac{\partial f(W_0 + R)}{\partial R} \right] = 0 \quad (3.10)$$

$$P \left[ C^p - C^w + (h + b)\Phi(X_0 + P) - b - \lambda \right] = 0 \quad (3.11)$$

**Non-negativity constraints:**  $R \geq 0, P \geq 0, \lambda \leq 0$

From the above non-negativity constraints, we derive three cases based on the optimal releases and production using  $\lambda \leq 0$  as follows:

1. When  $\lambda = 0$ , i.e., the clearing function is not tight, i.e.,  $P < f(W_0 + R)$ , and since  $C^r$  and  $C^w > 0$ , (3.7) cannot hold at equality, implying by complementary slackness  $R^* = 0$ . Therefore,  $P^* < f(W_0)$ . We have two subcases based on the values of  $P^*$  at the optimum solution.

- (a)  $\lambda = 0$  implies that  $P^* < f(W_0)$ , which does not hold when  $W_0 = 0$ , so a solution with  $P^* = 0$  is not compatible with  $W_0 = 0$ . If  $W_0 > 0$ ,  $R^* = P^* = 0$  implies (3.7) and (3.8) are both satisfied as inequalities.

- (b)  $P^* > 0$  implies  $P^* < f(W_0)$  for all  $W_0 > 0$ . The optimal production level  $P$  is then obtained from (3.11) as

$$X_0 + P^* = \Phi^{-1} \left( \frac{b - (C^p - C^w)}{b + h} \right) \quad (3.12)$$

The marginal cost of moving an item from WIP to FGI is equal to the marginal benefit of adding a unit to reduce FGI cost. We have a feasible candidate solution, (Case 2:  $R^* = 0, P^* > 0, \lambda^* = 0$ ).

2. When  $\lambda < 0$ , i.e., there is no slack in the CF, i.e.,  $P^* = f(W_0 + R^*)$ . We have three subcases that  $R^*$  takes at the optimum solution.

- (a)  $R^* = 0$  implies  $P^* = f(W_0)$  allowing  $P^* = 0$  only when  $W_0 = 0$  which is a degenerate solution since at the origin  $P^* = 0$ . The optimal  $\lambda$  is calculated using (3.8), (Case 3:  $R^* = P^* = 0, \lambda^* < 0$ ). Similar to 1a, when  $R^* = P^* = 0$  for a sufficiently large FG supporting a do-nothing policy, the effect of CF in the CF-based PI problem vanishes. Under such a situation,  $\lambda^* \leq 0$  irrespective of any amount of initial WIP.
- (b) When  $R^* = 0$  and  $W_0 > 0$ , we have  $P^* = f(W_0)$ . The initial WIP is just sufficient to produce FG without any additional releases. (3.7) implies

$$\left. C^r + C^w + \lambda \frac{\partial f(W_0 + R)}{\partial R} \right]_{R^*=0} > 0$$

which in turn gives

$$0 > \lambda > \frac{-(C^r + C^w)}{\frac{\partial f(W_0 + R)}{\partial R}}$$

We observe that the slope of the CF is positive for  $W_0 > 0$ . Therefore, the optimal  $\lambda$  satisfies the inequality (3.7). From (3.8) and (3.7), we have

$$0 > C^p - C^w + (h + b)\Phi(X_0 + P) - b \geq \left. \frac{-(C^r + C^w)}{\frac{\partial f(W_0 + R)}{\partial R}} \right]_{R^*=0}$$

i.e., the sum of the net cost of producing one unit and the marginal savings of adding one FG unit to the expected cost fails to offset the costs of the associated releases needed to obtain that additional unit of output. (Case 2:  $R^* = 0, P^* > 0, \lambda^* < 0$ ).

(c) When  $R^* > 0$ , initial WIP is not sufficient to raise the FG implies

$$C^p - C^w + (h + b)\Phi(X_0 + P) - b = \left. \frac{-(C^r + C^w)}{\frac{\partial f(W_0 + R)}{\partial R}} \right]_{R^*} \quad (3.13)$$

To have  $\lambda < 0$  requires

$$C^w - C^p + b > (h + b)\Phi(X_0 + P^*)$$

Using  $P^* = f(W_0 + R^*)$  in (3.13) gives

$$C^p - C^w + (h + b)\Phi(X_0 + f(W_0 + R^*)) - b = \left. \frac{-(C^r + C^w)}{\frac{\partial f(W_0 + R^*)}{\partial R}} \right]_{R^*} \quad (3.14)$$

Thus, the optimal release  $R^*$  is derived by numerically solving (3.14) and the optimal production quantity is given by  $P^* = f(W_0 + R^*)$  yielding (Case 1:  $R^*, P^* > 0, \lambda^* < 0$ ). Alternatively, we solve the system of equations (3.7) and (3.8), to obtain  $\lambda^*$  and  $R^*$ . The solution is valid only when  $\lambda^* < 0$  and  $R^* > 0$  and the optimal production is  $P^* = f(W_0 + R^*)$ . Given  $R^*$  and  $P^*$ , we calculate the ending WIP level as  $W = W_0 + R^* - P^*$ . The ending WIP calculated from the WIP balance equation illustrates the queuing effect in the production system, requiring it to carry that much workload to produce  $P^*$  units, resulting in ending WIP  $W$ . This means a sufficient workload is needed to produce and thus replenish the FGI. When  $C^r = C^w = 0$ , and initial states  $(X_0, 0)$ , we recover the classical newsvendor model without fixed costs.

In summary, we have five possible optimal situations describing three cases as solution characterization. Table 3.1 illustrates the summary of optimal solutions, solution characterization cases, and subcases as listed in the KKT conditions with inference.

Table 3.1: Summary of optimal solutions using solution characterization

Solution	$R^*$	$P^*$	$\lambda^*$	Cases	Classification	Notes
1	$R^* > 0$	$P^* > 0$	$\lambda^* < 0$	1	2c	$P^* = f(W_0 + R^*)$
2	$R^* = 0$	$P^* > 0$	$\lambda^* < 0$	2	2b	$P^* = f(W_0), W_0 > 0$
3	$R^* = 0$	$P^* > 0$	$\lambda^* = 0$	2	1b	$P^* < f(W_0), W_0 > 0$
4	$R^* = 0$	$P^* = 0$	$\lambda^* < 0$	3	2a	$P^* = f(W_0), W_0 = 0$
5	$R^* = 0$	$P^* = 0$	$\lambda^* = 0$	3	1a	$P^* < f(W_0), W_0 > 0$

**Numerical Example 1.** Given the initial WIP and FG inventory levels, the objective is to find the optimal production and corresponding releases. The relevant data are as follows:  $C^r = 1$ ,  $C^w = 0.25$ ,  $C^p=1.25$ ,  $h=0.3$  and  $b=6$  (implying a 95% service level in an uncapacitated newsvendor model). Assuming Normally distributed demand with mean 5 and standard deviation 1. We use a Missbauer clearing function with parameters  $K_1=10$ , and  $K_2=10$ , respectively. Using the SPP-NLP, the optimal production, planned releases, and the Lagrange multiplier are calculated as shown in Table 3.2. The optimality condition from Section 3.4 is shown in 'Opt. Case' column and the classification based on each subcase are shown in the 'Classification' column.

Table 3.2: Optimal structure and solution characterization

$X_0$	$W_0$	$\lambda$	$R^*$	$P^*$	Cost	Opt. Case	Classification
-5	0	-5.00	11.55	4.23	53.30	1	2c
0	0	-4.40	9.56	3.70	23.75	1	2c
1	10	-2.30	0.00	3.82	9.39	2	2b
0	10	-4.25	0.00	3.82	13.77	2	2b
5	2	0.00	0.00	0.82	2.30	2	1b
5	0	-1.85	0.00	0.00	2.51	3	2a
20	1	0.00	0.00	0.00	4.75	3	1a

**Numerical Example 2.** We modified the relevant data as follows:  $C^r = 0.3$ ,  $C^w = 0.7$ ,  $C^p=0.5$ ,  $h=0.1$  and  $b=6$  (implying a 98% service level in an uncapacitated newsvendor model). Assuming Normally distributed demand with mean 7 and standard deviation 1 in a Missbauer CF with parameters  $K_1=10$ , and  $K_2=10$ , respectively. Table 3.3 shows the optimality condition for each initial state from Section 3.4 in 'Opt. Case' column and the subcases in the 'Classification' column. The reasons for the absence of 1a and 1b are due to the cost parameters: Since  $C^w > C^p$ , it is always optimal to produce when  $W_0 > 0$ .

Table 3.3: Optimal structure and solution characterization

$X_0$	$W_0$	$\lambda$	$R^*$	$P^*$	Cost	Opt. Case	Classification
-7	0	-6.20	18.42	5.61	67.61	1	2c
0	0	-5.83	17.43	5.45	25.80	1	2c
2	10	-4.31	2.70	4.50	12.99	1	2c
0	20	-5.43	0.00	5.86	20.07	2	2b
6	4	-1.39	0.00	1.80	3.25	2	2b
10	1	-0.10	0.00	0.49	0.95	2	2b
8	0	-1.07	0.00	0.00	0.61	3	2a

## 3.5 Single period problem - optimality conditions

With the three possible conditions derived in Section 3.4, we use a non-linear programming formulation that solves the single-stage, single-item zero lead time stochastic production inventory problem where a clearing function governs the replenishment. We give two examples using Missbauer and Graves CF to illustrate the application of NLP and the structure of the optimal solution. We compare the CF-based solution with the capacitated and uncapacitated production inventory system without CF and conduct sensitivity analysis for cost parameters.

### 3.5.1 Special case of Graves CF

Suppose we use Graves CF given by  $P = \alpha\Lambda$  in (3.3b), where  $\alpha$  denotes the fraction of the workload converted into output during the period. Using  $P = \alpha(W_0 + R)$  in (3.3b), the optimal production is then obtained as

$$X_0 + P^* = \Phi^{-1} \left[ \frac{b + C^w - C^p - (C^r + C^w) \frac{1}{\alpha}}{b + h} \right] \quad (3.15)$$

allowing the optimal production to be calculated in closed form. Using (3.15) in  $P^* = \alpha(W_0 + R)$ , we get

$$W_0 + R^* = \frac{\Phi^{-1} \left[ \frac{b + C^w - C^p - (C^r + C^w) \frac{1}{\alpha}}{b + h} \right] - X_0}{\alpha} \quad (3.16)$$

$$R^* = \frac{P^*}{\alpha} - W_0 \quad (3.17)$$

From (3.17), when  $W_0 > \frac{P^*}{\alpha}$ , we get  $R^* = 0$  and the ending WIP becomes  $W = W_0 - P^*$  yielding Case 2 as the optimal condition. On the other hand, from (3.16), when  $X_0 > P^*$ , we get Case 3 as the optimal condition irrespective of the initial WIP condition. Lastly, when  $P^* > X_0$  and  $\frac{P^*}{\alpha} > W_0$ , we get  $R^*, P^* > 0$  yielding Case 1 as the optimal condition. We discuss the solution characterization, effect of cost inputs, demand parameters, and utilization with Graves CF for values of  $\alpha \in \{0.5, 0.25\}$  in Appendix A.6.

## 3.6 Numerical Experimentation and Design

### 3.6.1 Using Missbauer CF

Missbauer (2002) derived a clearing function using a steady state  $M/G/1$  queue as

$$P = \frac{1}{2} \left( K_1 + K_2 + \Lambda - \sqrt{K_1^2 + 2K_1K_2 + K_2^2 - 2K_1\Lambda + 2K_2\Lambda + \Lambda^2} \right) \quad (3.18)$$

where  $\Lambda = W_0 + R$ ,  $K_1$  is the maximum production capacity, and  $K_2$  a parameter determining the shape of the clearing function. The domain and the range of  $P$  are  $[0, \infty)$  and  $[0, K_1)$ , respectively. (3.18) is a continuous, concave, and non-decreasing function, and its inverse is given by

$$f^{-1}(\Lambda) = \frac{P^2 - K_1P - K_2P}{P - K_1} \quad (3.19)$$

The domain of  $f^{-1}(\Lambda) = [0, K_1)$  and the range of  $f^{-1}(\Lambda) = [0, \infty)$ . The inverse function is useful when calculating the optimal release given the initial WIP and the optimal production. i.e., the optimal release is given by  $R = \max\{f^{-1}(P) - W_0, 0\}$ .



Figure 3.2: Effect of  $K_2$  for  $K_1 = 10$  using Missbauer CF

Figure 3.2 shows the shape of the Missbauer CF with its parameters  $K_1$  and  $K_2$ . For illustration, we show the relationship between the workload and expected output by fixing  $K_1 = 10$  and varying  $K_2 \in \{5, 10, 15, 20\}$ . The horizontal line segment represents the maximum production capacity denoted by  $K_1$ . Since this CF is derived assuming an  $M/G/1$  queue in a steady state, the variance of the processing time affects the shape of

the CF. i.e., the higher the variance, the larger the  $K_2$  value. By reducing variability in the processing time, we get a steep function that requires less workload to yield a given amount of production. For instance, to produce 6 units of output, we need 14 units of workload when  $K_2 = 5$ , 21 units of workload when  $K_2 = 10$ , 29 units of workload when  $K_2 = 15$ , and 36 units of workload when  $K_2 = 20$ . We observe even a small difference in the expected production affects the required workload, and hence the costs of the system by a significant amount.

### 3.6.2 Experimental design

The experimental design consists of Missbauer CF with a capacity of  $K_1=10$ , two levels of the CF shape parameter (steep  $K_2 = 10$  and flat  $K_2 = 30$ ), and mean demand selected relative to the capacity to study the impact on low utilization 70% and high utilization 90%. We use Normally distributed demand of  $\mathcal{N}(7, 1)$  and  $\mathcal{N}(9, 2)$ . We consider three combinations of cost parameters to capture the effect of releases, WIP, production, and FGI cost with various initial states  $(X_0, W_0)$ . Table 3.4 shows the complete experimental design for the single period problem.

Table 3.4: Complete experimental design for the single period problem

Parameters	Factors	Levels
Demand	Mean demand = 7, 9 and standard deviation = 1, 2	4
Cost pattern	E1, E2, E3 (shown in Table 3.5 below)	3
CF using Missbauer CF	$K_1 = 10, K_2 = 10, 30$	2
Initial FGI state $X_0$	$X_0 \in \{-\mathbb{E}[D], \dots, 2\mathbb{E}[D]\}$	For $\mathcal{N}(7, 1)$ , 22 levels
Initial WIP state $W_0$	$W_0 \in \{0, 1, 2, 5, 10, 20, 40, 60, 80\}$	9

We consider a range of initial states  $(X_0, W_0)$  to capture the effect of initial states with other input parameters. For  $\mathcal{N}(7, 1)$ , we have  $(22*9=198)$  initial states, and for  $\mathcal{N}(9, 1)$ , we have  $(28*9 = 252)$  initial states in the design.

Table 3.5: Cost parameters for the experimental design

Experiments	$C^r$	$C^w$	$C^p$	$h$	$b$
E1	0	0	2	0.3	6, 30
E2	1	0	0	0.3	6, 30
E3	1	0.25	2	0.3	6, 30

The objective function is common to all the models, including UCPI and CPI models. The only difference between the CF-based models, the UCPI, and the CPI models is

that in the CF-based models, production is constrained by the CF-based replenishment that includes a congestion effect in the production inventory system. In the CPI model, production is limited to the maximum production capacity; in the UCPI model, there is no capacity constraint to the output. Recall the relationship between  $P^*$  and  $R^*$  when  $W_0 = 0$  and  $W_0 > 0$  in Section 3.2. The cost calculation for release, ending WIP, and production is the following. 1) The cost of ending WIP is calculated as  $C^w W$ , where  $W = W_0 - P^*$  when  $W_0 > P^*$ , and  $W = 0$ , when  $P^* > W_0$ . 2) The cost of release is calculated as  $C^r R^*$ , where  $R^* = 0$ , when  $W_0 > P^*$ , and  $R^* = P^* - W_0$ , when  $P^* > W_0$ . Finally, the cost of production is given by  $C^p P^*$ , where  $P^* = R^*$  when  $W_0 = 0$  in the case of UCPI and CPI models, and  $P^* = \min\{P_{max}, R^* + W_0\}$  when  $W_0 > 0$  in the case of CPI model. The expected cost of the production inventory model is shown in (3.3)a.

We consider three sets of cost parameters to study the impact of cost and compare them with capacitated and uncapacitated single-period problems. Table 3.5 summarizes the list of cost patterns. The cost structure  $C^r = 0$ ,  $C^w = 0$  in Experiment 1 allows comparison with uncapacitated and capacitated newsvendor models that do not consider clearing functions in replenishment. In Experiment 2, we study the impact of ending WIP and FGI using  $C^w = 0$ ,  $C^p = 0$ , and focus on  $C^r$ ,  $h$ , and  $b$ . Finally, *E3* considers all the costs, including the material, WIP, and production cost, to study the optimal solution for the clearing function-based replenishment.

We solve 10800 instances with three cost patterns, four demand patterns, and two CF parameters with the initial states (22x9) and (28x9), respectively. To capture the following outcomes, we compare the results with the capacitated and uncapacitated production inventory problem of 3600 instances each. 1) We compare the clearing function-based production replenishment with capacitated newsvendor and uncapacitated newsvendor models. We compare the cost and production of the CF-based model with the classical newsvendor model with zero fixed costs, and the initial  $W_0$ ,  $C^r$ ,  $C^w$  are zero; 2) Discuss the impact of cost parameters, demand distribution parameters, capacity restriction vs. utilization and how initial states affect the optimal cost and solution based on the shape of the CF,

The SPP-NLP is coded in MATLAB R2020 and solved on an Intel Core i7-7500U CPU @ 2.9GHz, 16GB RAM. We use MATLAB's `fmincon` solver as the NLP engine to find the optimal solution. We report the computation time on an average basis. We show the summary of run times for a corresponding cost input and type of NLP problem solved

between CF-based PI problem, capacitated PI problem, and uncapacitated PI problem in the following. The computational time varies between the type of the cost inputs listed as per Table 3.5.

Using E1 cost, the uncapacitated PI problem assumes an unlimited raw material availability. The reason is that maximum production is achieved at an infinite workload. So the fmincon solver tries to achieve the optimal production by extremely high releases. It takes 105 seconds for 22 initial states and 127 seconds for 28 initial states for the uncapacitated version. For the capacitated version, we get nearly 300 seconds for  $WIP < 20$  and 90 seconds for  $WIP > 20$  in 22 states. Similarly, we get 350 seconds for  $WIP < 20$  and 100 seconds for  $WIP > 20$  in 28 states. It is evident that the runtime reduces when WIP is sufficiently large, and this is because of the more significant releases due to zero release cost and sufficient WIP necessary to produce the maximum production required to meet the demand at minimum objective cost. The CF-based version takes 132 seconds for 22 states and 165 seconds for 28 states.

Using E2, we keep the cost of production and WIP as zero. Therefore the computation time changes concerning the experimental design. We get 28 seconds for 22 states and 35 seconds for 28 states in a CF-based PI problem. For capacitated version, we get close to 35 seconds for 22 states and 50 seconds for 28 states. Like the previous example, the uncapacitated version of the PI problem using E2 takes 32 seconds for 22 states and close to 45 seconds for 28 states.

Using E3, for the uncapacitated PI problem, the average run time for each experimental run with 22 initial states took close to 50 seconds; however, for 28 states, it took 70 seconds on average. For the capacitated PI problem, the average run time for each experimental run with 22 initial states took close to 35 seconds; however, for 28 states, it took 40 seconds on average. For the CF-based PI problem, the average run time for each experimental run with 22 initial states took close to 21 seconds; however, for 28 states, it took 26 seconds on average.

### **3.7 Experimental Results and Discussion**

In this section, we discuss the solution characterization using Missbauer CF. In the subsequent sections, we analyze the impact of input parameters, the shape of the CF, and the utilization.

### 3.7.1 Solution Characterization

Using the SPP-NLP formulation, we obtain the optimal releases and the production given an initial state  $(X_0, W_0)$  with  $C^r = 1$ ,  $C^p = 2$ ,  $C^w = 0.25$ ,  $h = 0.3$ ,  $b \in \{6, 30\}$  with  $K_2 = 10$ , respectively. We share the corresponding optimality condition for each initial state in Figures 3.3 and 3.4 corresponding to the mean demand 7 and 9 with a standard deviation of 1, respectively. We compare the results for  $K_2 \in \{10, 30\}$ ,  $b = 30$  and share the corresponding optimal solution described in Section 3.5. We use blue to represent Case 1, yellow for Case 2, and green for Case 3 to showcase the optimal solution for the SPP-NLP. For instance, in Figure 3.3(a), for mean demand = 7, (6, 5) corresponds to Case 2, and (10, 0) corresponds to Case 3, respectively. Similarly, in Figure 3.4(b), (6, 5) corresponds to Case 1, and (8, 10) corresponds to Case 2, respectively.

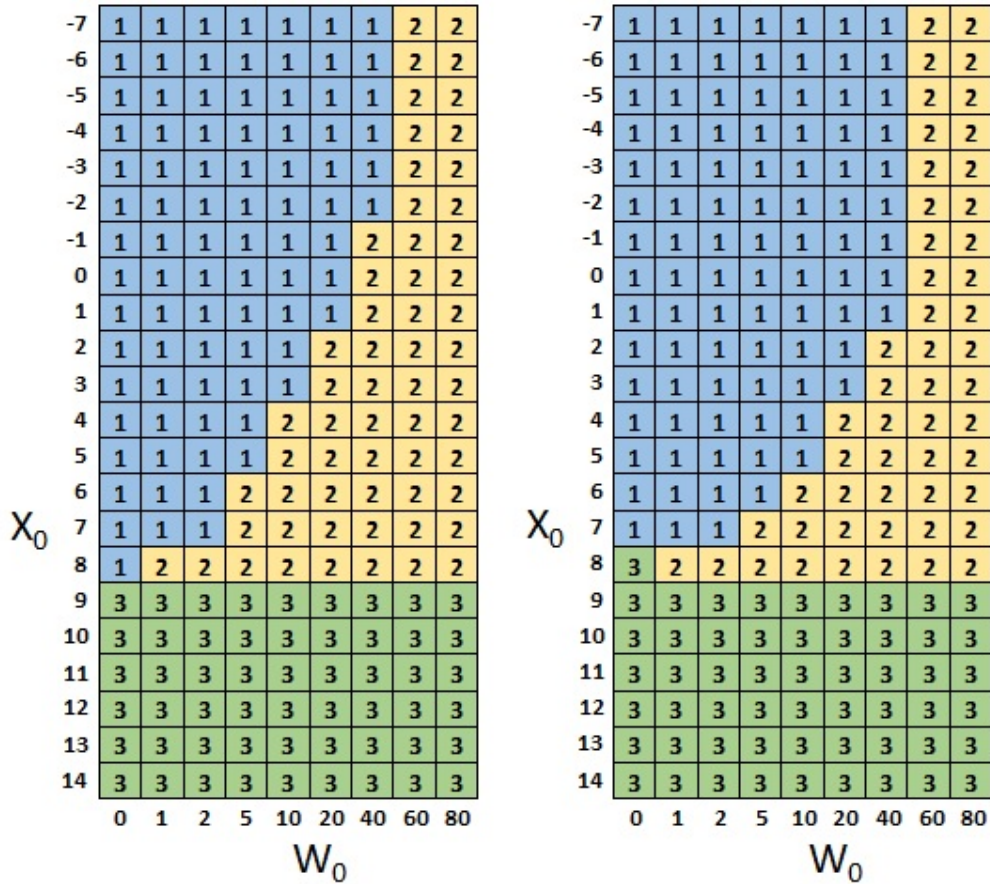


Figure 3.3: Optimal solution characterization for demand  $\sim \mathcal{N}(7, 1)$  using  $b = 30$  (a) For  $K_2=10$  (b) For  $K_2=30$

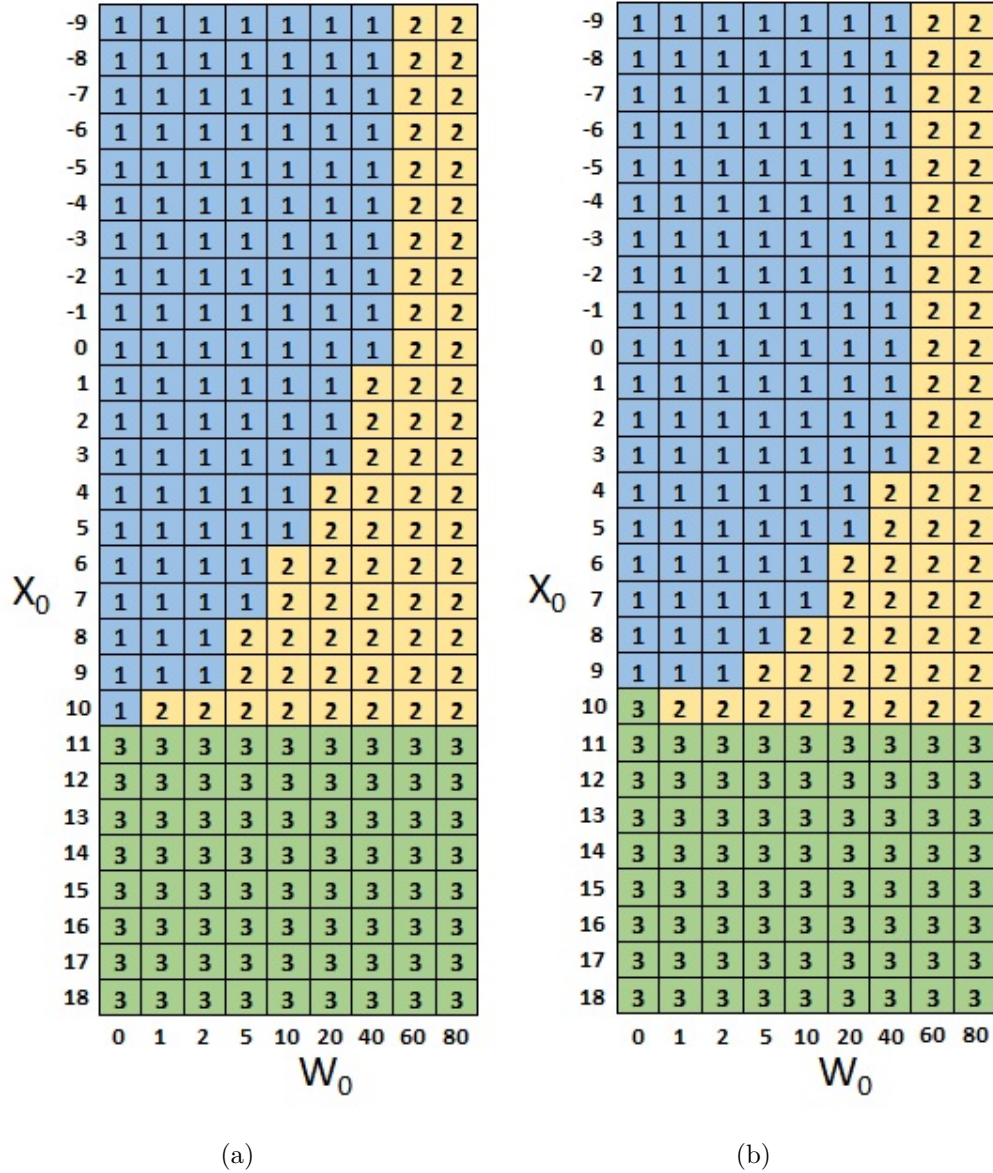


Figure 3.4: Optimal solution characterization for demand  $\sim \mathcal{N}(9, 1)$  using  $b = 30$  (a) For  $K_2=10$  (b) For  $K_2=30$

Figures 3.3 and 3.4 show that the optimal solutions follow the three cases discussed in Section 3.5. We compare the optimality case for states from  $(-7,0)$  to  $(14,80)$  by solving 198 instances in one experiment. We find Case 2 ( $R^* = 0, P^* > 0$ ) and 3 ( $R^* = 0, P^* = 0$ ) as cases when sufficient WIP and FGI are present in the system. These require only production from the existing WIP and do not require any releases to meet the initial FGI. However, when the variance of the processing time increases, the shape of the CF becomes flat. Thus, the existing WIP is insufficient, and additional release is necessary

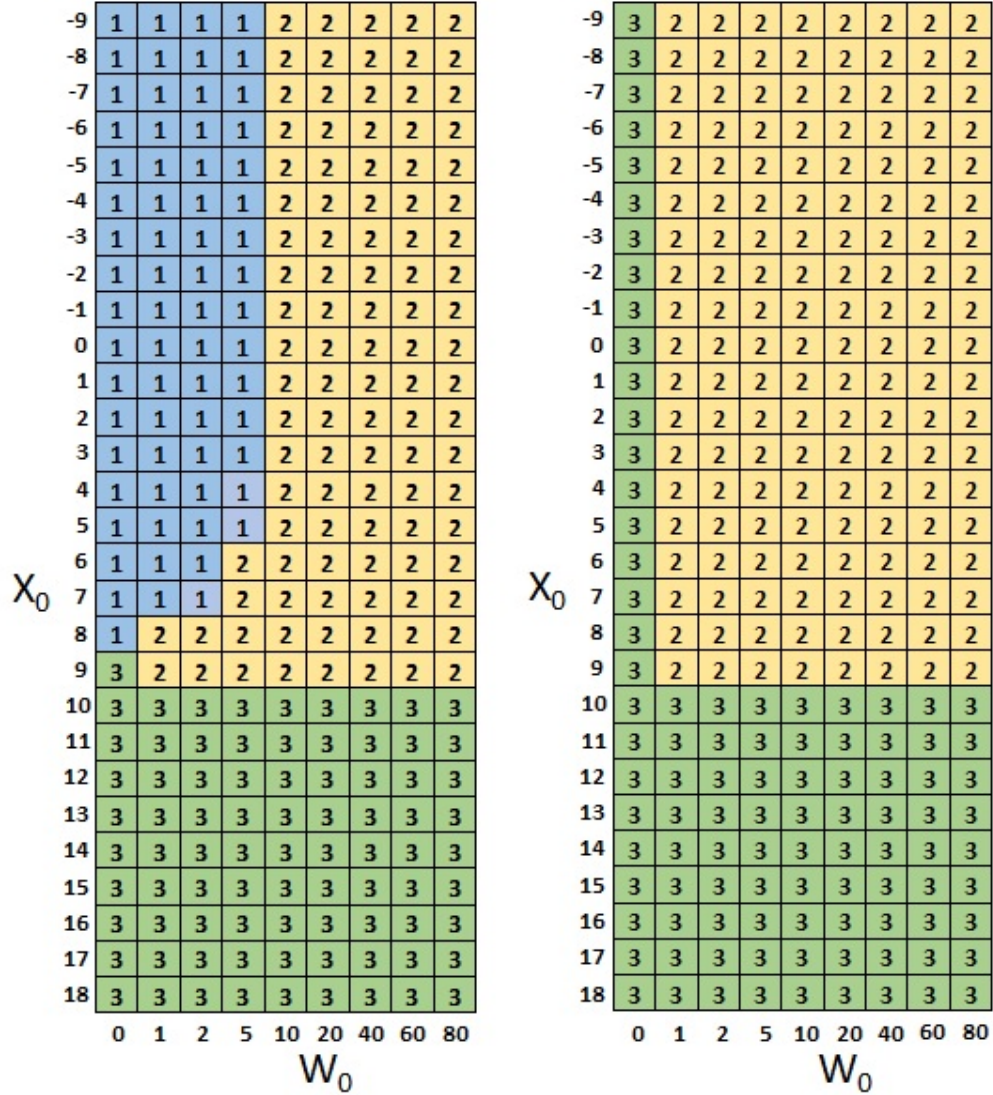
for the production to meet the external demand. We find this situation in Figure 3.3(a) and (b) for states represented in blue as optimal conditions to Case 1. On the other hand, the optimal policy follows Case 2 in cases when  $W_0$  is sufficient to produce  $P^*$  shown in Figure 3.4(a) and (b) represented by yellow color. At high utilization, the maximum production capacity restricts the expected production, and it is necessary to use the existing WIP to produce up to the capacity to meet demand. Depending upon the shape of the CF, the optimal workload  $R + W_0$  determines whether the optimal solution follows Case 1 or 2 based on the initial WIP.

Lastly, the optimal solution is to do nothing for initial states between (9,0) and (14,80) when  $K_2 = 10$  at 70% utilization ( $U$ ) and between (11,0) and (18,80) at 90% utilization, where the FGI is sufficiently large enough to meet the demand as shown in Figure 3.3 and 3.4 (a). i.e., by following Case 3, the optimal policy is  $R^* = P^* = 0$ .

Table 3.6: Results for demand  $\sim \mathcal{N}(7, 1)$

		$K_2=10$			$K_2=30$		
$X_0$	$W_0$	$R^*$	$P^*$	Cost	$R^*$	$P^*$	Cost
-7	20	24.32	7.85	233.69	36.77	6.27	293.75
-6	20	24.32	7.85	203.69	36.77	6.27	263.75
-5	20	24.32	7.85	173.69	36.77	6.27	233.75
-4	20	24.30	7.85	143.70	36.77	6.27	203.75
-3	20	23.91	7.83	113.85	36.77	6.27	173.75
-2	20	21.52	7.72	85.32	36.62	6.27	143.78
-1	20	15.99	7.41	60.89	35.07	6.20	114.22
0	20	9.19	6.90	42.16	29.69	5.93	86.73
1	20	2.90	6.25	28.33	21.44	5.45	63.31
2	20	0.00	5.86	18.79	12.76	4.82	44.31
3	20	0.00	5.49	15.96	4.87	4.09	29.02
4	20	0.00	4.49	14.21	0.00	3.54	16.97
5	20	0.00	3.49	12.46	0.00	3.49	12.46
6	20	0.00	2.49	10.71	0.00	2.49	10.71
7	20	0.00	1.49	8.96	0.00	1.49	8.96
8	20	0.00	0.49	7.21	0.00	0.49	7.21
9	20	0.00	0.00	5.86	0.00	0.00	5.86
10	20	0.00	0.00	5.91	0.00	0.00	5.91

Figures 3.3(a) and (b) and 3.4(a) and (b) compare the area of the blue shaded region between the two shape parameters of the Missbauer CF. When the CF is flat, additional releases are required to meet the workload requirements and the area of the shaded region when  $K_2 = 30$  is greater than the area of the blue shaded region when  $K_2 = 10$  regardless of the mean demand and similar cost parameters. The boundary line that separates the optimality condition between Case 1 and 2 is critical that determines the optimal  $R$  and  $P$  concerning the shape of the CF and the initial states.



(a)

(b)

Figure 3.5: Optimal solution characterization for demand  $\sim \mathcal{N}(9, 1)$  using  $b = 6$  (a) For  $K_2=10$  (b) For  $K_2=30$

Table 3.6 illustrates the optimal  $R$  and  $P$  and expected cost for various initial FG states when  $W_0 = 20$  and  $b = 30$ . It is interesting to observe how the shape of the CF affects the optimal solution. For instance, the optimal solution follows Case 2 when  $K_2 = 10$ , whereas, the optimal solution follows Case 1 when  $K_2 = 30$  for  $(2, 20)$ , and  $(3, 20)$ , respectively. For states between  $X_0 = 5$  and 8, the optimal solution follows Case 2 and yields similar results regarding  $P^*$  and the expected cost. This behavior is due to the sufficient workload being available at the beginning of the period, and  $P^*$  is given by the workload when the slope  $\frac{\partial f}{\partial \lambda}$  of the CF is near the origin.

Finally, we analyze the solution characterization for varying backorder costs. Figure 3.5 shows the optimality conditions for  $K_2 \in \{10, 30\}$  using  $b = 6$ . Comparing Figures 3.4 (a) and 3.5 (a), we see that the backorder cost for the other cost parameters determines the additional release requirement for an initial state. When  $b = 30$ , we get a larger area of the blue-shaded region (Case 1) in comparison with the area of the blue-shaded region when  $b = 6$ . From Figure 3.5 (b), we conclude that the 'do nothing' policy is optimal when initial WIP = 0 for all FG states when  $K_2 = 30$ . This is because it is not optimal to incur high release cost for additional release when the initial WIP is zero in a flat CF. The cost structure plays a critical role in deciding the optimal solution. For  $W_0 \in \{1, \dots, 80\}$ , the optimal policy follows Case 2 when  $K_2 = 30$ , in most FG states between -9 and 9. This phenomenon is because the backorder cost is less than the cost of a large amount of additional release required to produce FG in case of  $K_2 = 30$ . Therefore, a do-nothing policy is optimal in the case of  $W_0 = 0$ , and based on the available WIP, the PI model follows Case 2 for WIP states between 1 and 80.

Figure 3.6 refers to the experimental run for utilization levels with demand  $\mathcal{N}(7, 1)$  and  $\mathcal{N}(9, 1)$ . In this experiment, we set the maximum production capacity  $K_1 = 10$  and  $K_2 = 10$  to study the impact of utilization on the optimality conditions. For low utilization, the policy follows Case 3 when the initial FGI is larger than the mean demand, irrespective of the in-process inventory. It is evident that at low utilization, the system has sufficient capacity. The initial state determines how much workload is necessary to produce FGI to minimize the objective cost function. We also find a similar trend between Case 1 and 2 when the initial FGI state ranges between  $\{-7, \dots, 8\}$  for mean demand 7 and  $\{-7, \dots, 10\}$  for mean demand 9 with WIP state ranges between  $\{0, 1, 2, 5, 10, 20, 40\}$  respectively. We observe Case 2 as optimal when  $W_0 > 40$ . This behavior is due to the system following a conservative pattern (no additional releases) using the existing

WIP to produce sufficient FG to minimize the expected cost. Lastly, we see Case 1 (blue-shaded region) as the optimality condition when sufficient FGI and WIP are unavailable. The system requires additional releases to raise the workload to the required quantity, and production takes place to meet the external demand, resulting in  $R^*, P^* > 0$ .

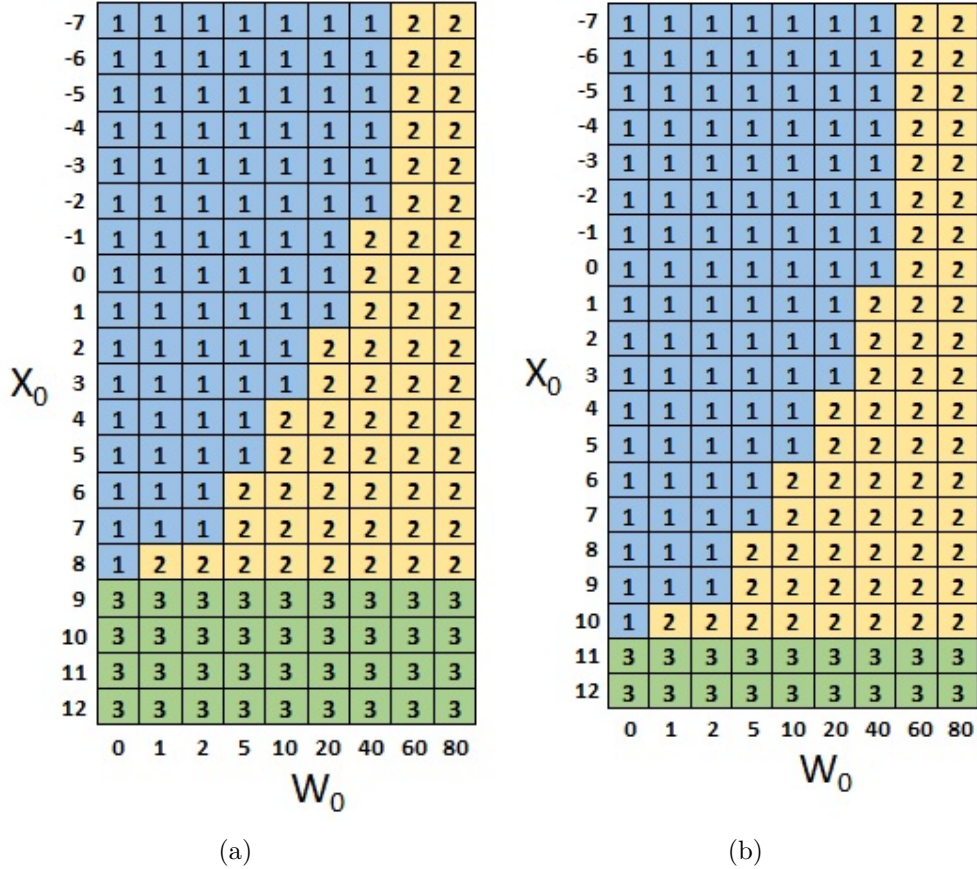


Figure 3.6: Optimal solution characterization for  $K_1=10$ ,  $K_2=10$  (a) At  $U = 70\%$  Demand  $\sim \mathcal{N}(7, 1)$  (b) At  $U = 90\%$  Demand  $\sim \mathcal{N}(9, 1)$

We now discuss the optimal solution based on the expected cost and shape of the CF. The results regarding other combinations of  $K_2$ , and initial states are similar regarding the cost changes and policy parameters; hence these results are not given here. We find the expected cost of the system is higher when  $K_2 = 30$ . When the initial FGI is sufficiently large to meet the external demand, we observe Case 3 as optimal as expected. Otherwise, production takes place from the existing WIP instead of additional releases to the system yielding Case 2.

On the other hand, based on the experimental results shown in Figures 3.5, we observe all three optimality cases when  $K_2 = 10$  and  $b = 6$ . This is because of the high release cost required to produce FG compared relative to the backorder cost. However, we observe only Cases 2 and 3 when  $K_2 = 30$ . We get Case 2 as optimal when initial FGI is insufficient to meet the demand; hence existing WIP is utilized to meet the desired production quantity. Irrespective of the shape of the CF and the costs, when the initial FGI is sufficiently high, we observe "do nothing" as the optimal condition.

### 3.7.2 Effect of $C^p$

In the first experiment, we use  $\mathcal{N}(7, 1)$ , E1 cost parameters with  $b = 6$  to study the objective function costs between CF-based, CPI, and UCPI models for various  $(X_0, W_0)$ . Using E1, for  $X_0$  and  $W_0=0$ , the UCPI model becomes a classical newsvendor model, and CPI becomes capacitated newsvendor model. All CF-based (steep and flat) models and CPI model costs remain the same for any given initial state and are unaffected by the initial WIP. Without any queuing present in the production system, there is no congestion effect in the UCPI and CPI models compared to a CF-based replenishment. All models converge to that of the UCPI model's  $P^*$  and expected cost when we get  $P^* < P_{max}$ . i.e., the objective function and  $P^*$  converge to the classical newsvendor model when the initial state becomes  $(-2,0)$  and higher in FGI.

All models give the same results (objective function and  $P^*$ ) when sufficient  $X_0$  is present in the system at the beginning of the period. i.e., this case follows 'Do nothing' as the optimal strategy, in which production does not take place, so the CF is not active. The rise in the objective function is significant when the shape of the CF changes. For instance, for  $K_2=10$ , the expected cost for  $b = 6$  is 44 for  $(-7,0)$ . However, the expected cost becomes 144 when  $b = 30$ . We conclude that the CF-based PI model requires more releases to produce one output.

Comparing objective function values using E1 cost is a trivial case for UCPI and CPI models because the CPI and UCPI models, by their operation, will never hold any WIP in a zero-lead time situation. Furthermore, the cost inputs  $C^r = C^w = 0$  allow infinite releases into the PI models; thereby, the effect of initial WIP on the objective function value is negligible by the CF-based models. It is interesting to observe that if we substitute the optimal  $P^*$  in the CPI and CF-based models, we get an infeasible solution due to capacity restriction for states where  $P^* > P_{max}$ . Hence, we cannot simply use

classical newsvendor-based models in a production environment where congestion due to queuing effect plays a crucial role in finding the optimal release and production. We will study this sensitivity analysis by comparing the objective function cost using E3 cost configuration in subsection 3.7.4 under performance analysis of the UCPI solution.

### 3.7.3 Effect of $C^r$

In the first set of experiments, we study the effect of  $C^r$  by comparing the objective function value, optimal release, and production using E2 cost parameters. We use E2 to find the effect of  $C^r > 0$  concerning  $h$ ,  $b$ , the shape of the CF, and how it compares against the capacitated and uncapacitated production inventory models in the absence of  $C^p$  and  $C^w$ . This study will help us avoid the potential double counting of the WIP holding cost at the end of the period. First, we study the optimal  $R$  and  $P$  impact for  $W_0 = 0$ . For illustration, we use  $\mathcal{N}(7, 1)$  with backorder cost = 6. The CF-based PI model requires a large workload to produce output compared to the CPI and UCPI models due to the congestion effect captured in the shape of the CF. Thus, the optimal  $R$  and  $P$  vary significantly between the different initial states.

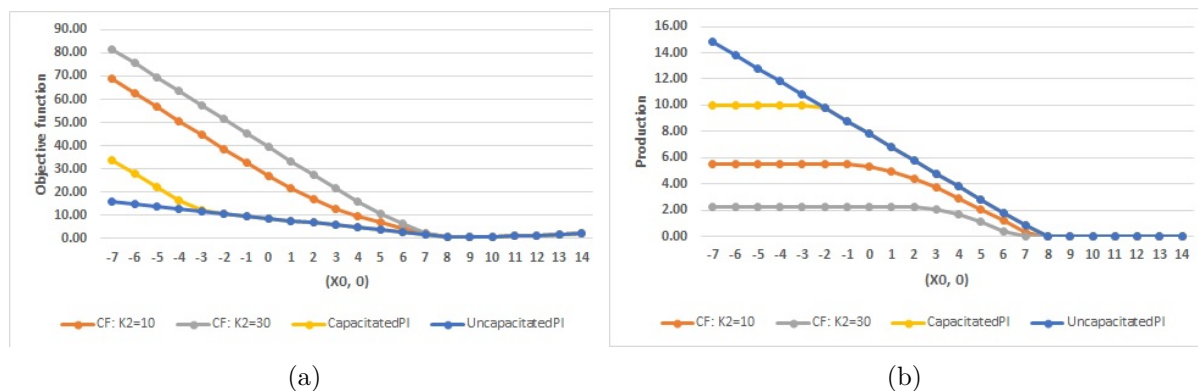


Figure 3.7: Comparing various models for  $W_0 = 0$  using  $\mathcal{N}(7, 1)$  with  $C^r = 1$ ,  $C^p = C^w = 0$ , and backorder cost = 6. (a) Objective function (b) Optimal production

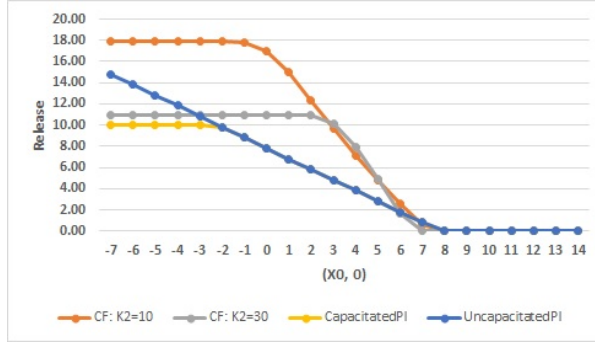


Figure 3.8: Comparing  $R^*$  from various models for  $W_0 = 0$  using  $\mathcal{N}(7, 1)$  with  $C^r = 1$ ,  $C^p = C^w = 0$ , and backorder cost = 6

Figures 3.7 and 3.8 show the objective function, production, and release for various  $X_0$  and  $W_0=0$ . From the release plot, it is interesting to notice that when  $W_0=0$ , CF-based models require additional releases despite increasing the release cost in the objective function. The cost tradeoff is between the cost of additional release and the backorder cost of not fulfilling the demand.

In the second set of experiments, we study the impact of initial WIP when  $W_0 > 0$ . For comparison, we take  $\mathcal{N}(9, 1)$  with  $b = 6$  in E2 cost inputs. We plot the optimal release, production, and expected cost across all the models to evaluate the model behavior for  $W_0 \in \{0, 1, 2, 5, 10, 20, 40, 60, 80\}$ , respectively. Figures 3.9, and 3.10 compare the objective function values using different WIP levels. These graphs indicate that the initial WIP is critical to the optimal cost. Based on the experimental results, it is evident that the expected cost is minimum with  $W_0 > 0$  in a CF-based PI model. We observe similar evidence in other models; hence these results are not given here. In a CF-based model, for a fixed  $X_0$ , when  $W_0$  increases, the objective function will decrease by  $C^r$  times  $\max\{0, f^{-1}(P^*) - W_0\}$  since  $C^w$ , and  $C^p$  are zero. Similarly, we witness an increase in the objective function for a flat CF when the initial FGI falls below zero due to an increase in the release cost. However, when there is sufficient FGI at the beginning of the period, the CF-based models converge to the UCPI model, supporting ‘do nothing’ as the optimal solution for the release and production, which is a trivial case.

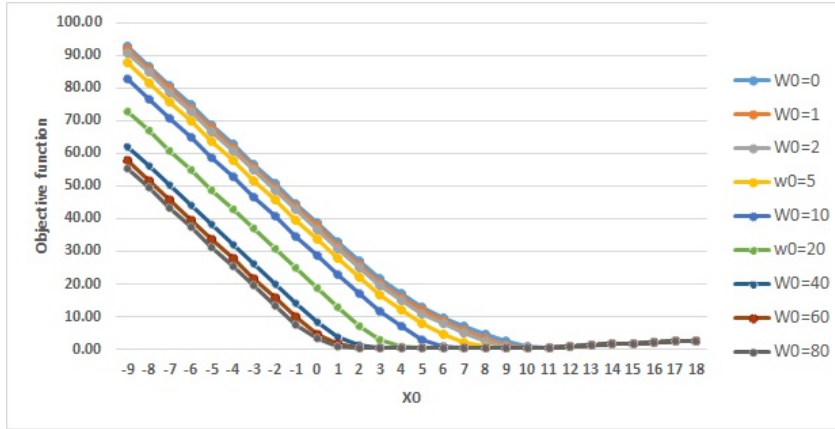


Figure 3.9: Expected cost of CF-based model for various WIP levels using a mean demand of 9,  $b=6$ , and  $K_2 = 10$

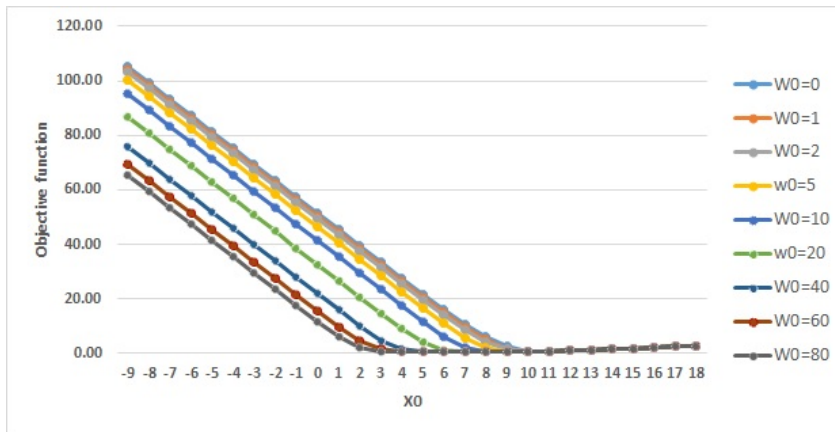


Figure 3.10: Expected cost of CF-based model for various WIP levels using a mean demand of 9,  $b=6$ , and  $K_2 = 30$

In all the models, no additional release is required to produce output when sufficient initial WIP is present in the system. Therefore, when  $W_0 > 20$ , the models use the initial WIP to produce  $P^*$  following Case 2 of the optimality conditions. Thus, we get zero release when  $W_0 \in \{20, 40, 60, 80\}$ . We observe this behavior across CF-based, CPI, and UCPI models. Figures 3.11 compare the release requirement when  $W_0 \in \{0, 10\}$  across the UCPI, CPI, and CF-based models with steep and flat CFs.

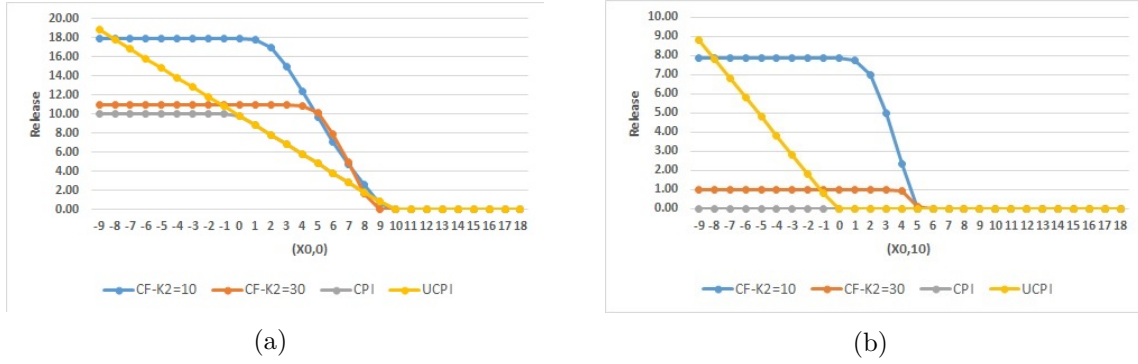


Figure 3.11: Comparing optimal release for various models using mean demand of 9,  $b=6$  when (a)  $W_0 = 0$  (b)  $W_0 = 10$

Figures 3.12 show the optimal production obtained by varying the initial WIP. The optimal production remains constant across models when the initial WIP  $\in \{0,1,2,5,10,60\}$ . We observe various  $P^*$  for the models when there is sufficient  $W_0$  in the system. In the case of a CF-based PI model, the capacity restriction and initial WIP play a critical role in deciding the maximum output possible concerning the workload requirements. The CF-based PI model requires an additional workload to produce output when the CF becomes flat. The initial WIP plays a significant role in finding the optimal production.

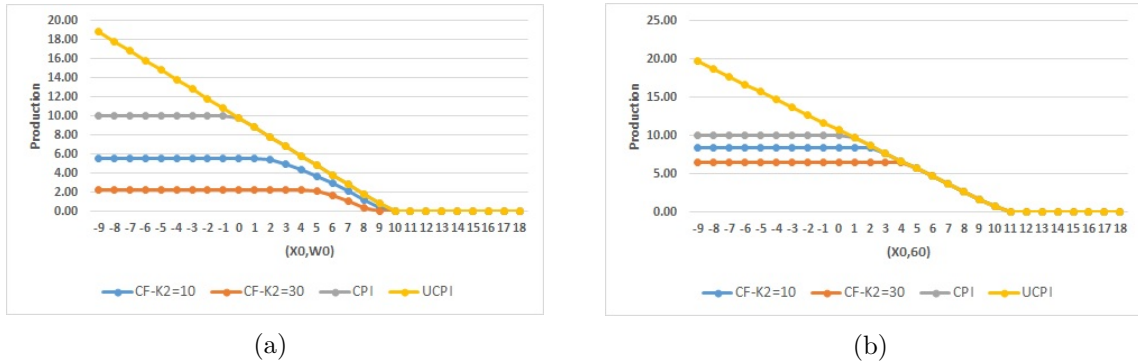


Figure 3.12: Comparing optimal production for various models using mean demand of 9,  $b=6$ , when (a)  $W_0 \in \{0, 1, 2, 5, 10\}$  (b)  $W_0 = 60$

### 3.7.4 Results of Cost Configuration E3

In this study, we take all non-zero cost values to study the effect of the demand coefficient of variation (COV),  $b/h$  ratio, capacity restriction, utilization, the impact of the shape parameter of the CF, and the sensitivity analysis using the UCPI solution in the CF-based models using E3 cost configuration values. This experiment gives the impact of all the cost parameters on the production inventory system and how input costs affect the

optimal production and release for various initial states and the CF parameters.

### Effect of Demand Coefficient of Variation

We present the results for various coefficient of variations (COV) of demand parameters and compare the optimal  $R$ ,  $P$ , and objective functions between the Missbauer CF-based models with two levels of  $K_2$ . In this experiment, we use backorder cost = 30 with an initial WIP of 0. We present the results of the four demand distributions in terms of  $\text{COV} \in \{\frac{1}{9}, \frac{1}{7}, \frac{2}{9}, \frac{2}{7}\}$  with the four different models: UCPI, CPI, CF-based with steep curvature, and CF-based with flat curvature, respectively.

Comparing the production quantity with all four COV values and models, we observe that the CF-based models yield less production than its UCPI counterparts. Figure 3.13 shows the optimal production for various models using four different COV values. In most initial states, when  $X_0 < 0$ , the optimal production remains constant. Due to the capacity restriction, the CPI model produces as much as possible and drops production when the initial FGI is higher. Compared with the COV values, we conclude the relationship as ‘the larger the COV, the larger the production.’ In contrast, we observe production values slightly differ while using a CF-based model with a steep CF. This behavior is due to the initial WIP levels; hence, a slight workload difference gives a small variation in the output that depends on the shape of the CF.

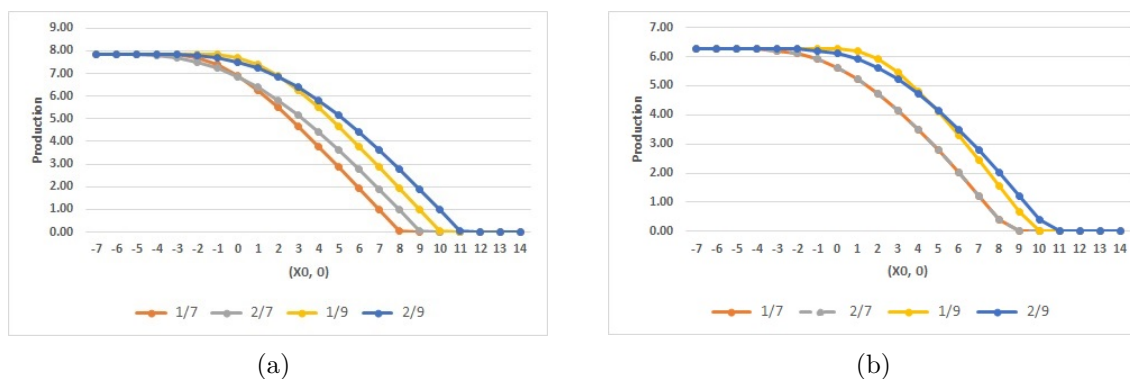


Figure 3.13: Comparing optimal production in CF-based model across different COV values when  $b = 30$  (a) CF-based model with  $K_2 = 10$  (b) CF-based model with  $K_2 = 30$

Figure 3.14 shows the optimal release plot for various models using four different COV values when  $W_0 = 0$ . Except for the UCPI and CPI models, all require relatively high release when  $X_0 < 0$ . The combined effect of the CF shape, COV, and the initial state determines the optimal release. We find a drastic variation between various demand

distributions when  $X_0$  ranges between 0 and  $\mathbb{E}[D]$ . Similarly, we observe that a large amount of release is required when the shape of the CF becomes flat. All the models converge to a single value  $R^* = 0$  when  $X_0 \gg \mathbb{E}[D]$  and follow a do-nothing policy.

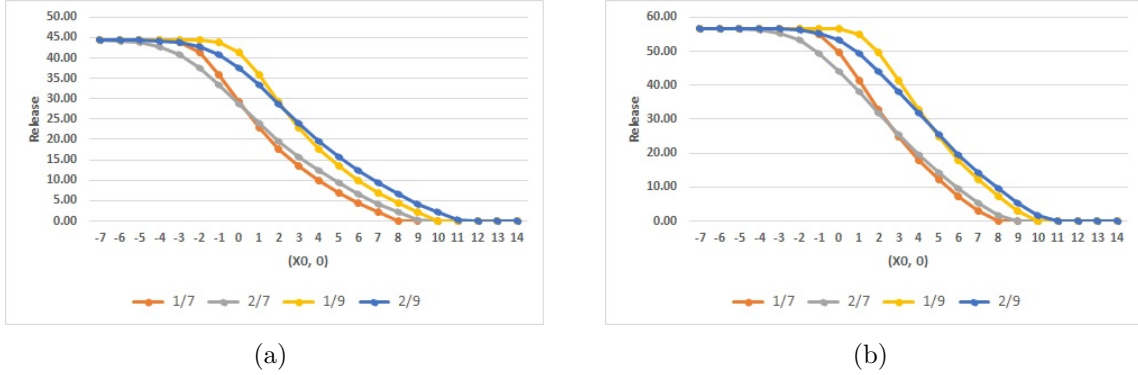


Figure 3.14: Comparing optimal release in CF-based model across different COV values when  $b = 30$  (a) CF-based model with  $K_2 = 10$  (b) CF-based model with  $K_2 = 30$

We observe that the CF-based models with high COV result in high objective function compared to the UCPI and CPI of the same initial states. This is due to the high release requirements to fulfill production requirements. This scenario is shared across all the models listed under study. Comparing different backorder costs between  $b=6$  and 30, we find similar patterns; hence they are not given here as illustrations. Figure 3.15 shows the expected cost for each model using four different COV values. Similar to the production plots, the larger the COV, the higher the expected cost of the PI model.

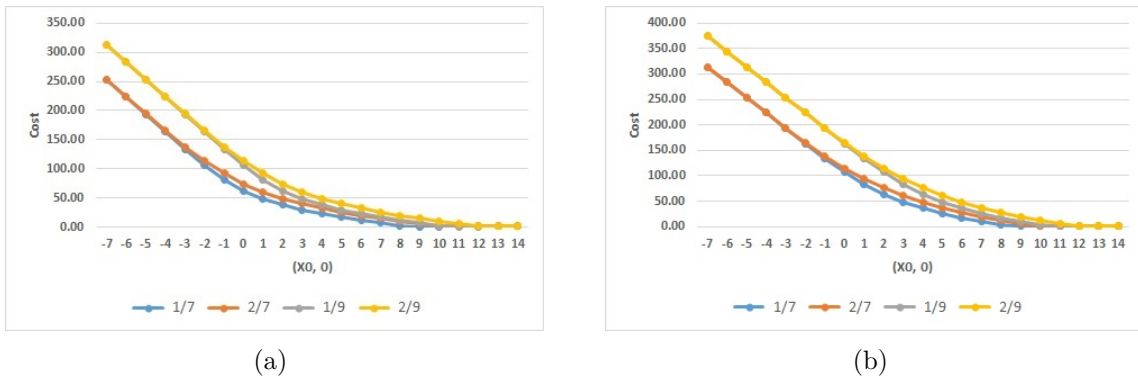


Figure 3.15: Comparing expected cost in CF-based model across different COV values when  $b = 30$  (a) CF-based model with  $K_2 = 10$  (b) CF-based model with  $K_2 = 30$

### Effect of $b/h$ ratio

In this experiment, we study the effect of the backorder and holding cost with the UCPI, CPI, and CF-based models to study the impact of the ending FGI cost using the optimal release, production, and objective function. We keep the holding cost 0.3 constant by varying backorder cost between 6 and 30 to get the  $b/h$  ratio  $\in \{20, 100\}$ . We report the results for all initial FGI and consider  $W_0 = 0$  to get the complete effect of the maximum release required to process the workload into production output. We use  $\mathcal{N}(7, 1)$  demand for convenience and report the results in the following.

Figure 3.16 shows the expected cost of the various models by varying the  $b/h$  ratio. The expected cost increases substantially when the shape of the CF becomes flat. The increase in the expected cost is due to the increase in the release cost when the  $b/h$  ratio is large. The cost tradeoff between the additional release cost and FGI cost is substantial in the case of  $b/h = 100$ . Therefore, under the cost assumptions, it is optimal to add additional releases to produce output which will reduce the FGI cost by avoiding any backorders.

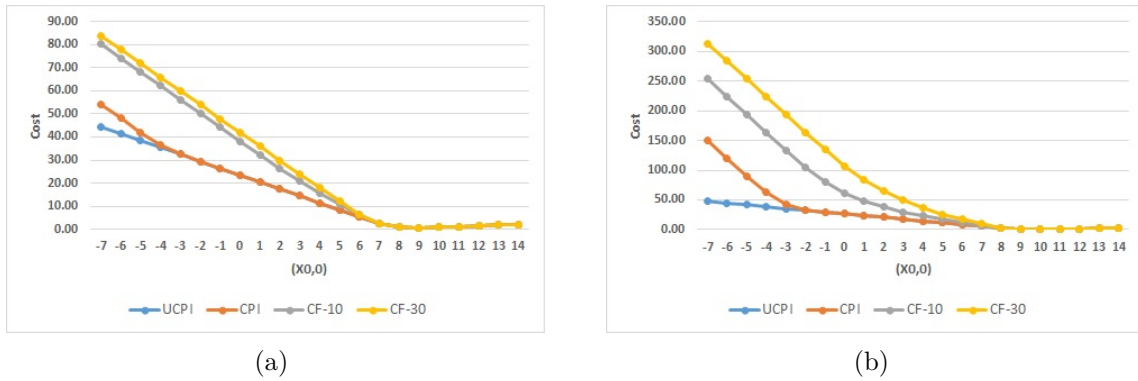


Figure 3.16: Comparing the expected cost of the UCPI, CPI, and CF-based models for various models for a given  $(X_0, 0)$  using  $\mathcal{N}(7, 1)$  when (a)  $b/h = 20$  (b)  $b/h = 100$

Figure 3.17 shows the optimal release of the various models by varying  $b/h$  ratio. For  $b/h=20$ , we get a ‘do nothing’ policy when the CF shape is flat. An additional release will increase the total expected cost; thus, no production resulting in a minimum cost. For the steep CF, the optimal conditions follow Cases 1 and 3 based on the initial states. In all the cases, we witness that Case 3 is typical when  $X_0 > \mathbb{E}[D]$  under the cost assumption, and it does not depend on the  $b/h$  ratio. We observe similar behavior when  $b/h$  is 100, except we get substantial additional releases to increase the workload when

the CF becomes flat.

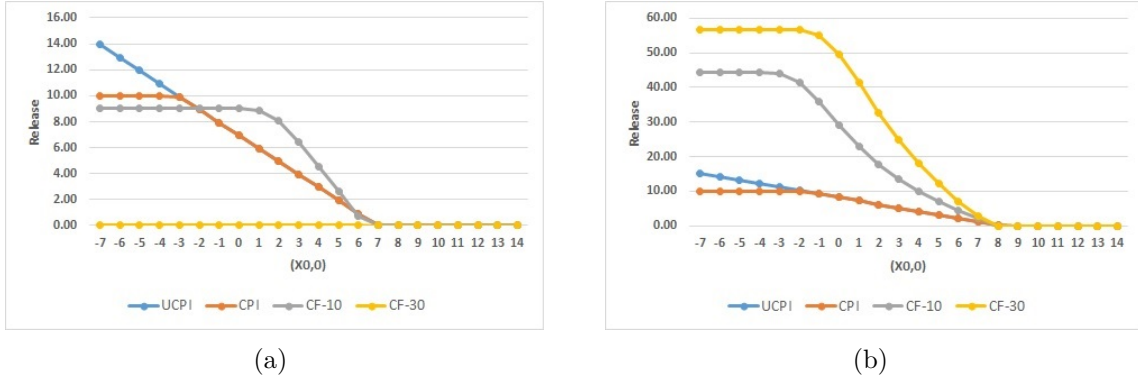


Figure 3.17: Comparing the optimal release of the UCPI, CPI, and CF-based models for various models for a given  $(X_0, 0)$  using  $\mathcal{N}(7, 1)$  when (a)  $b/h = 20$  (b)  $b/h = 100$

Figure 3.18 shows the effect of the shape parameter and  $b/h$  ratio on optimal production. When  $K_2 = 30$ , i.e., using a flat CF, and when  $b/h = 100$ , we require a large production than the steep CF-based model. On the other hand, we observe a 'do nothing' policy when  $b/h = 20$  and  $K_2 = 30$ . This is because of the cost tradeoff between  $C^r$  and  $C^p$  with  $b$ . This behavior requires a large number of releases, and therefore, a large release cost will affect the objective cost function. In summary, the higher the  $b/h$  ratio, the larger the production requirements when the CF is flat. However, when the CF is steep, depending on the cost parameters, the production requirement may vary based on the initial states and other cost parameters.

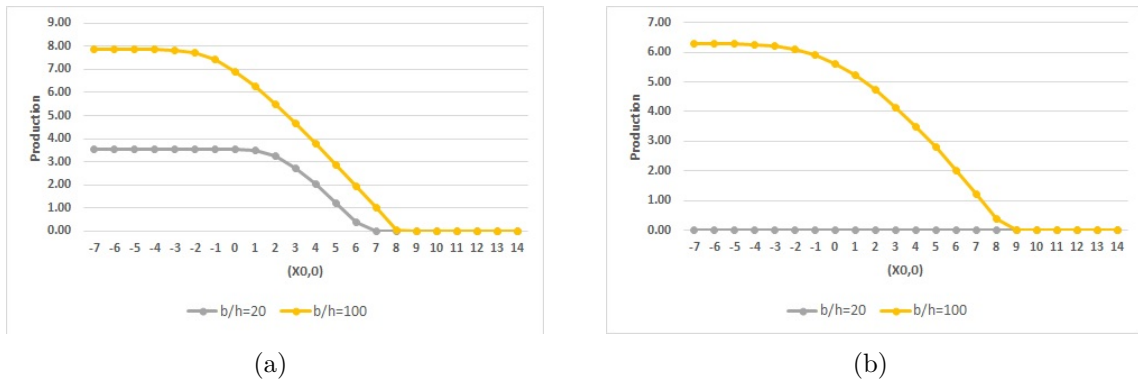


Figure 3.18: Comparison of  $P^*$  using  $b/h \in \{20, 100\}$  (a)  $K_2 = 10$  (b)  $K_2 = 30$

### **Impact of capacity restriction**

We examine the effect of capacity restriction among the models. We know that the UCPI model is a relaxation of the CPI model, which in turn is a relaxation of the CF-based models. Therefore, we restrict our investigation to CF-based models. The key results are the following. In the CF-based model, for instance, using a flat CF with  $K_2=30$ , a large amount of release is necessary to produce FGI. We get the maximum production, yielding a relatively high cost. This increase in the objective is due to the capacity restriction, and the shape of the CF determines how much production is necessary to minimize the objective cost. Otherwise, it is optimal not to produce anything resulting in a minimum expected cost. Based on the experimental results, we conclude that capacity restriction brings congestion in the CF-based production system. Under the given cost assumptions, we expect an additional release to raise the workload equivalent to the expected output to minimize the expected cost of the objective function when the shape of the CF becomes flat.

### **Effect of utilization**

In this experiment, we use two demand distributions  $\mathcal{N}(7, 2)$ , and  $\mathcal{N}(9, 2)$ , with  $b = 30$  in E3 cost parameters and  $W_0 = 5$  to capture utilization's effect at two levels (70% and 90%) with the CF's queuing effect. Figure 3.19 shows the optimal production, release, and expected cost for 70% utilization. At 70% utilization, we get the minimum objective for the UCPI model until  $X_0 = \mathbb{E}[D]$ , beyond which the cost converges to a single number across all the models, including the CF-based model with  $K_2=30$ . We observe all the three optimal conditions in all the models, including the trivial Case 3, when  $X_0 > > \mathbb{E}[D]$  under the given cost assumptions. i.e., no additional release and production are required. The CF-based model requires the highest level of releases to produce output when  $K_2 = 30$  in comparison with the same model when  $K_2 = 10$ .

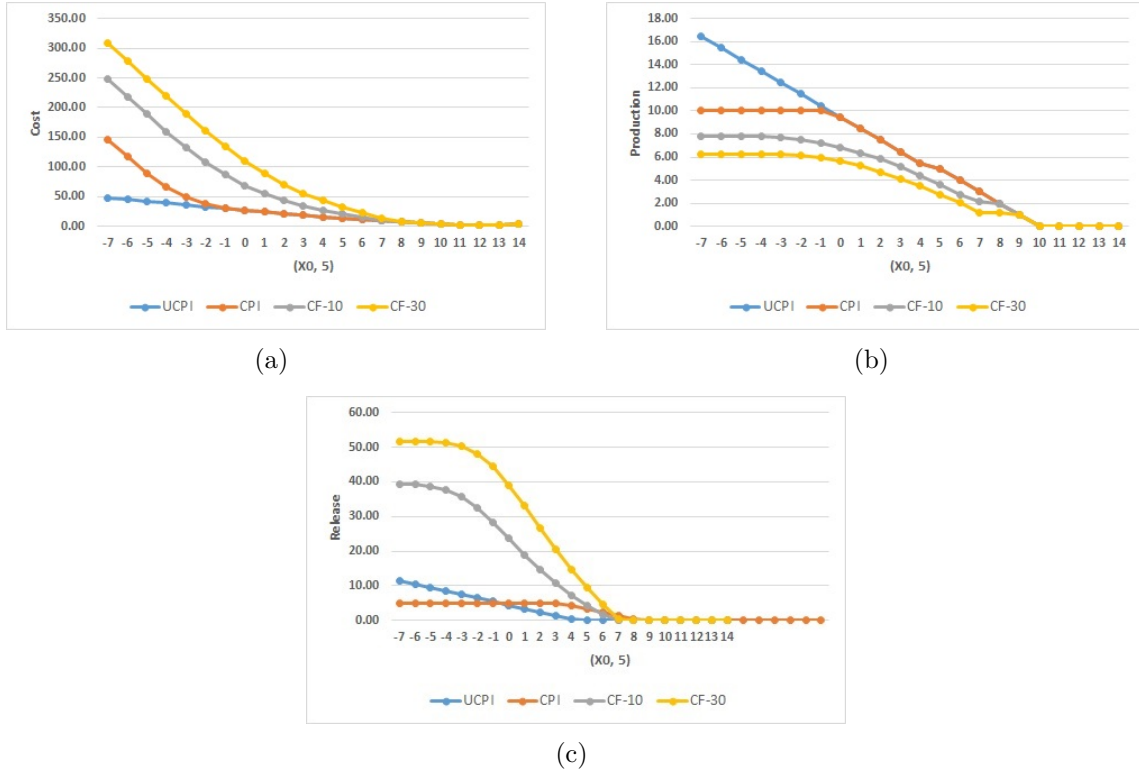


Figure 3.19: Comparing the expected cost,  $P^*$ , and  $R^*$  for various models using 70% utilization with E3 cost parameters and  $b = 30$ . We report the results for various FG states by keeping  $W_0 = 5$  to study the CF's queuing effect (a) Objective cost (b)  $P^*$  (c)  $R^*$

At 90% utilization, we get similar observations and trends as 70%, except the objective function has increased substantially due to the high mean demand concerning the capacity. i.e., the queuing effect in the production system allows more production to occur at high utilization. We witness congestion in the PI models due to high utilization and additional WIP requirement in the PI model.

### Impact of the shape parameter

In this experiment, we use  $\mathcal{N}(9, 2)$ , with  $b=30$  and  $W_0 \in \{0, 40\}$  to capture workload requirements in comparison with the shape parameter  $K_2$  of the CF. Figure 3.20 shows the objective cost,  $P^*$ , and  $R^*$  when  $W_0 = 0$ . In the first experiment, when  $W_0 = 0$ , we observe that the release required to raise the workload is substantial in a flat CF, and the effective production is relatively less in CF with  $K_2 = 30$ . This behavior is primarily due to the additional release required to increase the workload to produce the desired output. Otherwise, it is optimal not to produce the required output and follow a "do nothing" policy.

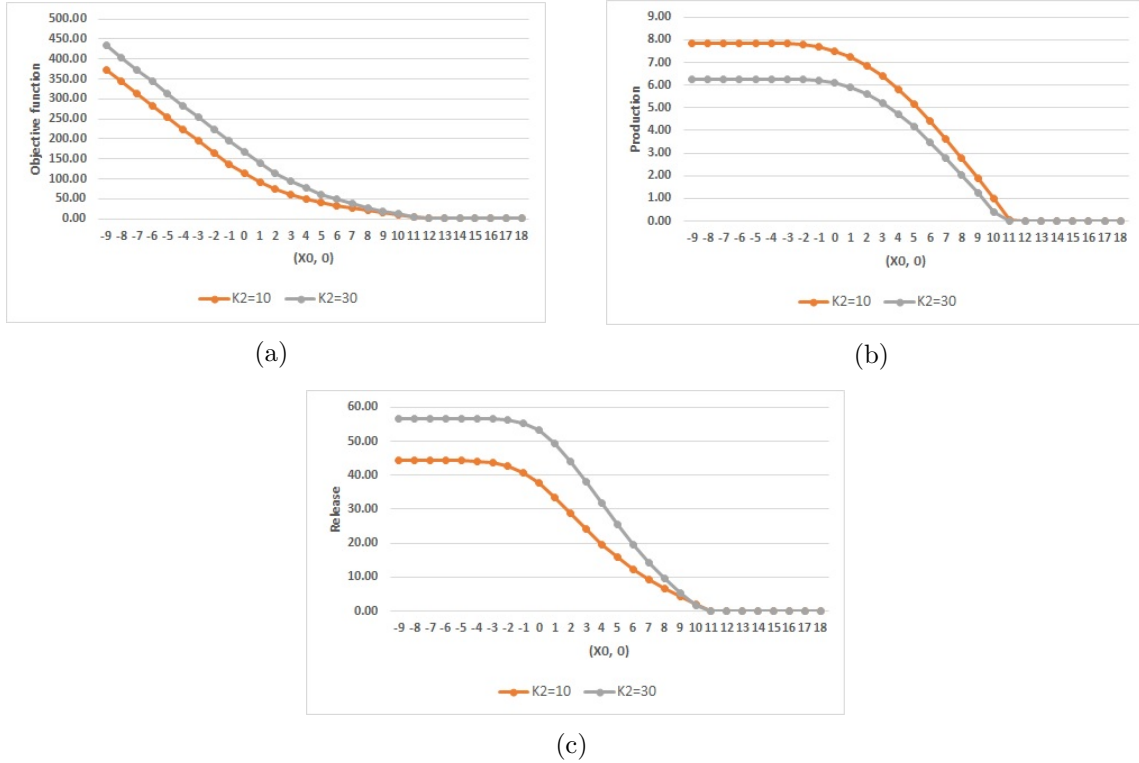


Figure 3.20: Comparing the expected cost,  $P^*$ , and  $R^*$  to study the effect of the CF shape parameter using  $\mathcal{N}(9, 2)$ ,  $b = 30$  when  $W_0 = 0$  (a) Objective cost (b)  $P^*$  (c)  $R^*$

The objective function cost tradeoff is between the backorder cost and the cost of release and production. On the other hand, when  $W_0 = 40$ , the optimal release is simply the difference between the optimal release at  $WIP=0$  and the  $W_0$ . In both scenarios, i.e.,  $W_0 = 0$ , and 40, the optimal production remains constant. The only difference is that the additional release requirement to raise the workload is substantial for a flat CF compared to a steep CF. Similar to the above argument, when there is sufficient initial WIP in the system, additional release requirements reduces; in turn, the system produces maximum production to achieve the minimum objective function that results from the tradeoff between the backorder cost of 30 and the cost of production, release and ending WIP., i.e.,  $b \mathbb{E}[D - X_0 - P]^+ > C^p P^* + C^r R^* + C^w W$

### Performance of UCPI and CPI solution with a CF

We conduct a sensitivity analysis by implementing the UCPI and CPI solutions in the CF-based model. The purpose is to see what happens if a newsvendor solution is implemented when replenishment is governed by a clearing function. Based on our previous discussions, we know that the UCPI and CPI models do not have a congestion effect because WIP

acts like a release. If  $W_0 > 0$ , it is consumed by the production resource first before adding an additional release. The analysis is performed with  $W_0 = 0$  and  $W_0 = 10$ . We consider  $\mathcal{N}(9, 1)$  demand with E3 cost inputs and  $b = 30$ . For  $W_0 = 0$ , both  $R^*$  and  $P^*$  for the UCPI and CPI models are the same due to zero congestion. However, in the CPI model, the maximum production is limited to  $P_{max}$  and hence  $R^* = P^* \leq P_{max}$ . When  $W_0 = 0$ , for a given  $R^*$ ,  $P$  is calculated by  $f(R^*)$ , and for a given  $P^*$ ,  $R$  is calculated by  $f^{-1}(P^*)$ . In contrast, when  $W_0 > 0$ , for a given  $R^*$ ,  $P$  is calculated by  $f(W_0)$  if  $R^* = 0$ ,  $f(R + W_0)$  if  $R^* > 0$ . Likewise, for a given  $P^*$ ,  $R$  is calculated by  $\max[f^{-1}(P^*) - W_0, 0]$ . Using the Missbauer CF, we get corresponding  $P$  for the given  $R^*$  and  $R$  for the given  $P^*$  from both the UCPI and CPI models. Then, we use this new  $(R^*, P)$  and  $(R, P^*)$  to get the objective function values when we substitute in the CF-based models separately for  $K_2=10$  and 30. The ‘Optimal cost’ column in Tables 3.7 and 3.8 indicate the optimal value of the CF-based model and the ‘Exp. cost’ column represents the  $(R^*, P)$  and  $(R, P^*)$  from the UCPI solution substituted in the CF-based objective function.

What could be more meaningful is to compare the expected cost of the CF-based model and the UCPI and CPI-based solutions in the CF-based model’s objective function. We denote the optimal cost of the CF-based solution as  $CFmodel_{cost}$  and the UCPI-based solution in the CF-based model’s objective function as  $UCPIInput_{cost}$ . Then, the optimality gap is given by

$$gap\% = \left( \frac{UCPIInput_{cost} - CFmodel_{cost}}{UCPIInput_{cost}} \right) 100\% \quad (3.20)$$

Similarly, we denote the CPI-based solution in the CF-based model’s objective function as  $CPIInput_{cost}$  and the optimality gap is given by

$$gap\% = \left( \frac{CPIInput_{cost} - CFmodel_{cost}}{CPIInput_{cost}} \right) 100\% \quad (3.21)$$

Tables 3.7 and 3.8 refer to the optimal cost comparison using optimality gap % for  $W_0=0$  using Missbauer CF by varying  $K_2$  between 10 and 30. We define two solutions, namely  $R$  as action and  $P$  as action from the UCPI solution.  $R$  as an action denotes  $R^*$  from UCPI, taken as an input to the CF to calculate the corresponding  $P$ . However,  $P$  as an action is the opposite of  $R$  as an action, where  $P^*$  from the UCPI model is taken as an input to the CF, and the corresponding  $R$  is calculated. We denote this as  $(R^*, P)$ , and  $(R, P^*)$ , respectively. Based on the results, we show the optimality gap using  $R$  and  $P$  as

actions and summarize the following. For  $X_0 < 0$ , where we get  $(R^*, P^* > P_{max})$  from the UCPI models, the solution is infeasible in practice due to capacity restrictions. Table 3.8 shows the infeasible results using  $P$  as an action. However, when we calculate  $P$  from  $R^*$  in  $R$  as action, we observe a feasible  $P < P_{max}$ . Despite the feasibility, we get a large gap in the expected cost shown in Table 3.7. Similarly, for states where a ‘do nothing’ policy is optimal, all the models converge to  $(R = 0, P = 0)$  as the optimal solution, which is a trivial case.

In a steep CF-based model, when  $W_0 = 0$ ,  $R$  as an action-based solution gives a large gap between the optimal cost and the UCPI input-based cost. The reason is that the reduced production for a given  $R^*$  yields a high backorder cost. In contrast, both  $R$  and  $P$  as action-based solutions gives a large gap between the optimal cost and the UCPI input-based cost when  $W_0 = 10$ . At  $W_0 = 10$ , sufficient workload is already present in the system, and the optimal release is zero in the UCPI model. Under a CF with  $K_2 = 10$ , the optimal production is  $f(W_0)$ , giving less production than the actual  $P^*$  from the UCPI model. Therefore,  $R$  as action gives a large gap when  $W_0=10$  and  $K_2 = 10$ . On the other hand, when  $W_0=10$  and  $K_2 = 30$ , we calculate  $R^* = f^{-1}(P) - W_0$  and get a large amount of release, which results in a high objective function cost. The optimality gap % for  $W_0=10$  is shown in the Appendix Tables A.11 and A.12, respectively. In a flat CF-based model, an increase in the additional release due to the large workload requirement increases the expected cost. In  $R$  as action, we get a large gap % similar to  $P$  as action when  $X_0 > 0$ . However, we find infeasible solutions due to capacity restrictions when  $X_0 < 0$  in  $P$  as action.

Table 3.7: Comparison of UCPI solution for demand  $\sim \mathcal{N}(9, 1)$  using  $R$  as action with (a)  $K_2=10$  (b)  $K_2=30$

$(X_0, 0)$	$R^*$	$P$	Exp. Cost	Optimal cost	Optimality Gap
-9	19.23	5.74	401.88	373.69	7%
-8	18.23	5.58	375.15	343.69	8%
-7	17.23	5.42	348.42	313.69	10%
-6	16.23	5.24	322.26	283.69	12%
-5	15.23	5.05	296.38	253.69	14%
-4	14.23	4.84	271.06	223.69	17%
-3	13.23	4.63	245.74	193.69	21%
-2	12.23	4.39	221.27	163.70	26%
-1	11.23	4.15	196.8	133.85	32%
0	10.23	3.88	173.18	105.32	39%
1	9.23	3.60	149.84	80.89	46%
2	8.23	3.30	127.06	62.16	51%
3	7.23	2.98	104.86	48.33	54%
4	6.23	2.64	83.30	37.92	54%
5	5.23	2.28	62.66	29.72	53%
6	4.23	1.89	43.93	22.94	48%
7	3.23	1.49	27.85	17.12	39%
8	2.23	1.05	15.99	11.94	25%
9	1.23	0.60	7.88	7.21	8%
10	0.23	0.12	2.83	2.82	0%

$(X_0, 0)$	$R^*$	$P$	Exp. Cost	Optimal cost	Optimality Gap
-9	19.23	3.45	466.57	433.75	7%
-8	18.23	3.32	439	403.75	8%
-7	17.23	3.19	411.42	373.75	9%
-6	16.23	3.05	384.13	343.75	11%
-5	15.23	2.91	356.83	313.75	12%
-4	14.23	2.77	329.54	283.75	14%
-3	13.23	2.61	302.81	253.75	16%
-2	12.23	2.46	275.79	223.75	19%
-1	11.23	2.30	249.06	193.75	22%
0	10.23	2.13	222.62	163.78	26%
1	9.23	1.95	196.45	134.22	32%
2	8.23	1.77	170.29	106.73	37%
3	7.23	1.58	144.40	83.31	42%
4	6.23	1.39	118.52	64.31	46%
5	5.23	1.19	92.94	49.02	47%
6	4.23	0.98	67.85	36.54	46%
7	3.23	0.76	44.13	26.13	41%
8	2.23	0.54	23.91	17.23	28%
9	1.23	0.30	10.24	9.46	8%
10	0.23	0.06	2.96	2.82	5%

Table 3.8: Comparison of UCPI solution for demand  $\sim \mathcal{N}(9, 1)$  using  $P$  as action with (a)  $K_2=10$  (b)  $K_2=30$

$(X_0, 0)$	$R$	$P^*$	Exp. Cost	Optimal cost	Optimality Gap
-9	Infeasible	19.23	Infeasible	373.69	-
-8	Infeasible	18.23	Infeasible	343.69	-
-7	Infeasible	17.23	Infeasible	313.69	-
-6	Infeasible	16.23	Infeasible	283.69	-
-5	Infeasible	15.23	Infeasible	253.69	-
-4	Infeasible	14.23	Infeasible	223.69	-
-3	Infeasible	13.23	Infeasible	193.69	-
-2	Infeasible	12.23	Infeasible	163.70	-
-1	Infeasible	11.23	Infeasible	133.85	-
0	Infeasible	10.23	Infeasible	105.32	-
1	129.50	9.23	179.99	80.89	55%
2	54.80	8.23	84.87	62.16	27%
3	33.36	7.23	56.32	48.33	14%
4	22.77	6.23	41.33	37.92	8%
5	16.21	5.23	31.38	29.72	5%
6	11.57	4.23	23.83	22.94	4%
7	8.01	3.23	17.63	17.12	3%
8	5.11	2.23	12.26	11.94	3%
9	2.64	1.23	7.42	7.21	3%
10	0.47	0.23	2.96	2.82	5%

$(X_0, 0)$	$R$	$P^*$	Exp. Cost	Optimal cost	Optimality Gap
-9	Infeasible	19.23	Infeasible	433.75	-
-8	Infeasible	18.23	Infeasible	403.75	-
-7	Infeasible	17.23	Infeasible	373.75	-
-6	Infeasible	16.23	Infeasible	343.75	-
-5	Infeasible	15.23	Infeasible	313.75	-
-4	Infeasible	14.23	Infeasible	283.75	-
-3	Infeasible	13.23	Infeasible	253.75	-
-2	Infeasible	12.23	Infeasible	223.75	-
-1	Infeasible	11.23	Infeasible	193.75	-
0	Infeasible	10.23	Infeasible	163.78	-
1	370.03	9.23	480.66	134.22	72%
2	147.95	8.23	201.31	106.73	47%
3	85.63	7.23	121.66	83.31	32%
4	55.86	6.23	82.69	64.31	22%
5	38.16	5.23	58.82	49.02	17%
6	26.25	4.23	42.18	36.54	13%
7	17.56	3.23	29.57	26.13	12%
8	10.85	2.23	19.43	17.23	11%
9	5.45	1.23	10.93	9.46	13%
10	0.95	0.23	3.56	2.82	21%

Tables 3.9 and 3.10 refer to the optimal cost comparison using optimality gap % between the CPI solution and the CF-based model for  $W_0 = 0$  using the Missbauer CF by varying  $K_2$  between 10 and 30. Similar to the UCPI analysis, we use both  $R$  as action (Use  $R^*$  from CPI solution, calculate  $P^*$  from CF) and  $P$  as action (Use  $P^*$  from CPI solution, calculate  $R^*$  from CF) to calculate the CF-based objective function cost. For  $R$

as action, we get  $P^* < P_{max}$ , which is feasible for the CF-based model. So we compare the expected cost of the CPI-based CF model and the optimal cost from the CF-based model shown in Table 3.9. However, in  $P$  as action, we get  $R^* = \infty$  for  $P^* = P_{max}$  and hence infinite cost. Comparing UCPI and CPI solutions, we get a higher expected cost in CPI in comparison with the UCPI-based solution in  $R$  as action when  $R^* > P_{max}$ . The reason for this increase in the expected cost is due to the increase in the backorder cost since  $P^*$  of CPI  $<$   $P^*$  of UCPI. Using  $P$  as action, in the UCPI based solution, we get infeasible  $R^*$  since  $P^* > P_{max}$ , whereas, in a CPI-based solution, we get infinite  $R^*$  since  $P^* = P_{max}$ . We observe a similar trend in terms of the optimality gap and expected cost when  $K_2 = 30$ . Hence these results are not shown here.

Table 3.9: Comparison of CPI solution for demand  $\sim \mathcal{N}(9, 1)$  using  $R$  as action with (a)  $K_2=10$  (b)  $K_2=30$

$(X_0, 0)$	$R^*$	$P$	Exp. Cost	Optimal cost	Optimality Gap
-9	10.00	3.82	444.59	373.69	16%
-8	10.00	3.82	414.59	343.69	17%
-7	10.00	3.82	384.59	313.69	18%
-6	10.00	3.82	354.59	283.69	20%
-5	10.00	3.82	324.59	253.69	22%
-4	10.00	3.82	294.58	223.69	24%
-3	10.00	3.82	264.58	193.69	27%
-2	10.00	3.82	234.59	163.70	30%
-1	10.00	3.82	204.59	133.85	35%
0	10.00	3.82	174.59	105.32	40%
1	9.23	3.60	149.84	80.89	46%
2	8.23	3.30	127.06	62.16	51%
3	7.23	2.98	104.86	48.33	54%
4	6.23	2.64	83.30	37.92	54%
5	5.23	2.28	62.66	29.72	53%
6	4.23	1.89	43.93	22.94	48%
7	3.23	1.49	27.85	17.12	39%
8	2.23	1.05	15.99	11.94	25%
9	1.23	0.60	7.88	7.21	8%
10	0.23	0.12	2.83	2.82	0%

$(X_0, 0)$	$R^*$	$P$	Exp. Cost	Optimal cost	Optimality Gap
-9	10.00	2.09	493.46	433.75	12%
-8	10.00	2.09	463.46	403.75	13%
-7	10.00	2.09	433.46	373.75	14%
-6	10.00	2.09	403.46	343.75	15%
-5	10.00	2.09	373.46	313.75	16%
-4	10.00	2.09	343.46	283.75	17%
-3	10.00	2.09	313.46	253.75	19%
-2	10.00	2.09	283.46	223.75	21%
-1	10.00	2.09	253.46	193.75	24%
0	10.00	2.09	223.46	163.78	27%
1	9.23	1.95	196.45	134.22	32%
2	8.23	1.77	170.29	106.73	37%
3	7.23	1.58	144.40	83.31	42%
4	6.23	1.39	118.52	64.31	46%
5	5.23	1.19	92.94	49.02	47%
6	4.23	0.98	67.85	36.54	46%
7	3.23	0.76	44.13	26.13	41%
8	2.23	0.54	23.91	17.23	28%
9	1.23	0.30	10.24	9.46	8%
10	0.23	0.06	2.96	2.82	5%

Table 3.10: Comparison of CPI solution for demand  $\sim \mathcal{N}(9, 1)$  using  $P$  as action with (a)  $K_2=10$  (b)  $K_2=30$

$(X_0, 0)$	$R$	$P^*$	Exp. Cost	Optimal cost	Optimality Gap
-9	$\infty$	10.00	$\infty$	373.69	-
-8	$\infty$	10.00	$\infty$	343.69	-
-7	$\infty$	10.00	$\infty$	313.69	-
-6	$\infty$	10.00	$\infty$	283.69	-
-5	$\infty$	10.00	$\infty$	253.69	-
-4	$\infty$	10.00	$\infty$	223.69	-
-3	$\infty$	10.00	$\infty$	193.69	-
-2	$\infty$	10.00	$\infty$	163.70	-
-1	$\infty$	10.00	$\infty$	133.85	-
0	$\infty$	10.00	$\infty$	105.32	-
1	129.50	9.23	179.99	80.89	55%
2	54.80	8.23	84.87	62.16	27%
3	33.36	7.23	56.32	48.33	14%
4	22.77	6.23	41.33	37.92	8%
5	16.21	5.23	31.38	29.72	5%
6	11.57	4.23	23.83	22.94	4%
7	8.01	3.23	17.63	17.12	3%
8	5.11	2.23	12.26	11.94	3%
9	2.64	1.23	7.42	7.21	3%
10	0.47	0.23	2.96	2.82	5%

$(X_0, 0)$	$R$	$P^*$	Exp. Cost	Optimal cost	Optimality Gap
-9	$\infty$	10.00	$\infty$	433.75	-
-8	$\infty$	10.00	$\infty$	403.75	-
-7	$\infty$	10.00	$\infty$	373.75	-
-6	$\infty$	10.00	$\infty$	343.75	-
-5	$\infty$	10.00	$\infty$	313.75	-
-4	$\infty$	10.00	$\infty$	283.75	-
-3	$\infty$	10.00	$\infty$	253.75	-
-2	$\infty$	10.00	$\infty$	223.75	-
-1	$\infty$	10.00	$\infty$	193.75	-
0	$\infty$	10.00	$\infty$	163.78	-
1	370.03	9.23	480.66	134.22	72%
2	147.95	8.23	201.31	106.73	47%
3	85.63	7.23	121.66	83.31	32%
4	55.86	6.23	82.69	64.31	22%
5	38.16	5.23	58.82	49.02	17%
6	26.25	4.23	42.18	36.54	13%
7	17.56	3.23	29.57	26.13	12%
8	10.85	2.23	19.43	17.23	11%
9	5.45	1.23	10.93	9.46	13%
10	0.95	0.23	3.56	2.82	21%

Similarly, we find a large optimality gap between the CPI-based solution and the CF-based expected cost. The optimality gap of the CPI solution is as large as the optimality gap corresponding to the UCPI solution. The reasons are as follows: In  $R$  as action,  $R^*$  is restricted by maximum production capacity from the CPI model to yield  $P_{CPI}^* < P_{UCPI}^*$ . Therefore, the expected cost of the CPI model is higher than the corresponding UCPI model due to the high backorder cost. In  $P$  as action, we get  $R^* = \infty$  for  $P^* = P_{max}$ . Therefore, we cannot get any comparison between the CF-based solution and the CPI solution. On the other hand, we find similar solutions between the CPI and UCPI solutions when  $P^* < P_{max}$  since both CPI and UCPI models yield the same  $P^*$ .

In summary, we cannot use newsvendor-based models as a substitute for capacitated production inventory models with congestion effects. Based on the input solution between  $(R^*, P)$  and  $(R, P^*)$ , we find that the optimality gap between UCPI and CPI-based solution and the CF-based solution is high. At high  $P^*$  from UCPI models, a large amount of releases are necessary to feed workload to the production resource. Therefore, for flat CF, using  $P^*$  in a CF-based model requires additional releases, thereby increasing the objective function. On the other hand, when  $R^*$  is relatively low, the CF-based PI model yields less output than expected. This reduced production will not be sufficient to meet demand, thereby increasing the objective function with a high penalty from the backorder cost than the holding cost.

## 3.8 Principal Results

We summarize the key results of the single-stage, single-item, capacitated stochastic production inventory system subject to congestion using a non-linear clearing function-based replenishment.

1. We observe three different optimal solution structures under normal conditions, which are referred to as Case 3: if FGI is sufficiently high, no production or releases are needed; Case 2: if WIP is sufficiently high, but FGI is not, then only production is required without any additional releases; and Case 1: if neither workload nor FGI is sufficiently high, both production and releases will be necessary to meet the external stochastic demand. Results indicate that the optimal solution differs based on initial states, CF parameters, and cost parameters of the single-period problem.
2. When  $W_0$  is low and under the given cost assumptions, the existing WIP is insufficient for the production to meet the stochastic demand when  $X_0 < 0$ . The shape of the CF decides how many additional releases are necessary to get the minimum objective function if the initial FG and initial WIP are insufficient to produce enough output.
3. We find the expected cost of the CF-based PI model increases with the value of the CF shape parameter  $K_2$ . i.e., under the given cost assumptions, when the CF shape is flat, we find a high expected cost compared to the expected cost obtained when the CF is steep.
4. We validate that the COV, utilization, initial state,  $b/h$  ratio, and CF shape affect the optimal production, release, and expected cost of the CF-based PI models. Unless the cost assumptions are modified, the optimal solution follows one of the three optimal conditions.
5. We cannot use newsvendor-based models as a substitute for capacitated production inventory models with congestion effects. Since replenishment is governed by a clearing function, conventional newsvendor models in place of the production inventory models yield poor results.

## 3.9 Conclusion

We consider a single-stage, single-product, single-period capacitated production inventory system with stochastic demand and production governed by a concave clearing function.

We defined the model assumptions, presented a mathematical formulation, and proved the convexity of the problem that yields a global minimum. We solve this non-linear program and characterize the optimal solution using three cases with the initial states  $(X_0, W_0)$ . We denote the optimal solution structure based on optimal releases  $R^*$  and production  $P^*$ . We represent the solution on three different conditions: 1)  $R^* > 0, P^* > 0$ , 2)  $R^* = 0, P^* > 0$ , and 3)  $R^* = 0, P^* = 0$ , respectively.

The optimal solution from the SPP guarantees to follow one of the three optimal cases as long as our cost assumptions hold. We proved the optimal solution structure using the KKT conditions (Bazaraa et al. 2013) and derived the optimality and feasibility conditions. The optimal solution depends on the initial states for all cases, exhibiting a state-dependent optimal solution structure. We compare the UCPI model and CPI models with the Missbauer CF-based models. Results indicate that the solution structure differs based on the system's initial states, CF parameters, and cost parameters.

Single-period problems are building blocks for the multi-period capacitated production inventory problems subject to clearing function-based replenishment to model the dynamics of the production system with the queuing effect due to capacity restriction and the non-linear relationship between the workload and the output. We study the multi-period optimal policy using a stochastic dynamic program and analyze the structure of the optimal policy using  $R$  and  $P$  as actions.

# Chapter 4

## Single-Stage Multi-Period Problem

This chapter presents stochastic dynamic programming algorithms to solve the multi-period problem. There are three main components of stochastic dynamic programming (SDP): the Bellman equation, continuous demand discretization, and value function approximation using numerical methods. We explain the model assumptions and implementation issues and present the SDP algorithm that acts as a benchmark for other approximate methods. We restrict our attention to a single-stage, single-item production inventory system subject to CF-based replenishment that represents the relationship between resource workload and output. We assume an appropriately fitted Missbauer clearing function is available. We aim to find an optimal policy for the multi-period capacitated production inventory model.

### 4.1 Dynamic Programming Formulation

Consider a multi-period production inventory problem consisting of a finite number of periods  $t = 1, \dots, T$ , with  $T + 1$  denoting the terminal period. The system state includes the work-in-process (WIP) and finished goods (FG) inventory at the start of period  $t$ , indicated by  $(X_{t-1}, W_{t-1})$  where  $X_{t-1}$  represents the FG inventory and  $W_{t-1}$  the WIP inventory at the beginning of period  $t$ . We define the following notation:

$h_t$	Unit finished goods inventory (FGI) holding cost in period $t$
$b_t$	Unit backorder cost in period $t$
$C_t^p$	Unit production cost in period $t$
$C_t^w$	Unit work in process (WIP) holding cost in period $t$
$C_t^r$	Unit release (material) cost in period $t$
$X_t$	Net inventory at the end of period $t$
$X_t^+$	On-hand inventory at the end of period $t$ , given by $\max \{0, X_t\}$
$X_t^-$	Backorder level at the end of period $t$ , given by $\min \{0, X_t\}$
$W_t$	Work in process on-hand at the end of period $t$
$\Theta_t(X_{t-1}, W_{t-1})$	Expected cost for given values of $W_{t-1}$ and $X_{t-1}$ in period $t$
$\Theta_{T+1}(X_T, W_T)$	Expected terminal cost for $X_T$ and $W_T$ in period $T + 1$
$\mathbf{D}_t$	Non-negative, random variable representing demand in period $t$
$\phi_t(\cdot)$	Probability density function of demand in period $t$
$\Phi_t(\cdot)$	Cumulative distribution function of demand in period $t$
$R_t$	Release quantity in period $t$
$P_t$	Production quantity in period $t$
$IP_t$	Order position at the beginning of period $t$ , given by $X_{t-1} + P_t$
$P_{max}$	Maximum production capacity
$\Lambda_t$	Workload at the start of period $t$ given by $\Lambda_t = W_{t-1} + R_t$
$f(\Lambda_t)$	Non-decreasing concave clearing function

Without loss of generality, we assume all cost and clearing function parameters are time-stationary across all periods. We assume the costs satisfy the relationships  $C^w < h < b$ ,  $C^r < b$ , and  $C^p < b$  to avoid trivial solutions where ordering nothing is optimal. The FG and WIP inventory balance equations are as follows:

$$X_t = X_{t-1} + P_t - \mathbf{D}_t \quad (4.1)$$

$$W_t = W_{t-1} + R_t - P_t \quad (4.2)$$

We assume that the slope of the CF at the origin  $\leq 1$ , to ensure  $P_t \leq f(\Lambda) \leq \Lambda$ . Therefore, the ending WIP is non-negative and deterministic. All demand values are integral. Moreover, the demand, release, and production levels are non-negative. The expected production is governed by a non-linear clearing function  $P_t = f(R_t + W_{t-1})$ . We consider two versions of the problem that differ in their assumptions about the production system. In the first, which we shall refer to as " $P$  as action," the only decision variable in

each period is the production quantity  $P_t$  that replenishes the FGI. Given  $P_t$ , the planned releases are calculated as  $R_t = \max\{0, f^{-1}(P_t) - W_{t-1}\}$ . The expected total cost of the system with  $P_t$  as decision (action) variable from period  $t$  to  $T$  is given by

$$\Theta_t(X_{t-1}, W_{t-1}) = \min_{P_t \geq 0} \left\{ C_t^p P_t + C_t^w \{f^{-1}(P_t) - P_t\} + C_t^r \max\{0, f^{-1}(P_t) - W_{t-1}\} \right. \\ \left. + g(X_{t-1} + P_t) + \mathbb{E}_{\mathbb{D}} \left[ \Theta_{t+1}(X_{t-1} + P_t - \mathbf{D}_t, f^{-1}(P_t) - P_t) \right] \right\} \quad (4.3)$$

where the expected holding and backorder cost is given by

$$g(X_{t-1} + P_t) = h \int_0^{X_{t-1} + P_t} (X_{t-1} + P_t - \mathbf{D}_t) \phi(\mathbf{D}) d\mathbf{D} \\ + b \int_{X_{t-1} + P_t}^{\infty} (\mathbf{D}_t - X_{t-1} - P_t) \phi(\mathbf{D}) d\mathbf{D} \quad (4.4)$$

which is the single period expected cost function for a newsvendor problem with  $P_t$  as the order quantity.

In the second version, which we shall refer to as " $R$  as action," the only decision variable in each period is the release quantity  $R_t$ . This model assumes that the production system will always produce the maximum amount permitted by the CF given the available workload, i.e.,  $P_t = f(W_{t-1} + R_t)$ . The expected total cost of the system with  $R_t$  as action variable from period  $t$  to  $T$  is given by

$$\Theta_t(X_{t-1}, W_{t-1}) = \min_{R_t \geq 0} \left\{ C_t^p f(R_t + W_{t-1}) + C_t^w [W_{t-1} + R_t - f(R_t + W_{t-1})] + C_t^r (R_t) \right. \\ \left. + g[X_{t-1} + f(R_t + W_{t-1})] + \mathbb{E}_{\mathbb{D}} \left[ \Theta_{t+1}(X_{t-1} + f(R_t + W_{t-1}) - \mathbf{D}_t, W_{t-1} + R_t - f(R_t + W_{t-1})) \right] \right\} \quad (4.5)$$

where the expected holding and backorder cost incurred at the end of period  $t$  is given by

$$g[X_{t-1} + f(R_t + W_{t-1})] = h \int_0^{X_{t-1} + f(R_t + W_{t-1})} (X_{t-1} + f(R_t + W_{t-1}) - \mathbf{D}_t) \phi(\mathbf{D}) d\mathbf{D} \\ + b \int_{X_{t-1} + f(R_t + W_{t-1})}^{\infty} (\mathbf{D}_t - X_{t-1} - f(R_t + W_{t-1})) \phi(\mathbf{D}) d\mathbf{D} \quad (4.6)$$

which is the single period expected cost function for a newsvendor problem with  $f(R_t + W_{t-1})$  as the order quantity. We first develop our dynamic programming formulation with

" $P$  as action" and then with " $R$  as action".

In each period  $t$ , starting from  $t = T + 1$ , we apply the SDP algorithm to enumerate all possible states and actions (releases) and find the optimal policy based on the cost function. For this, we must define a terminal cost function to solve the Bellman equation recursively backward in time. For purposes of analysis, we assume a terminal cost function that is linear in all state variables, of the form,

$$\Theta_{T+1}(X_T, W_T) = h_{T+1}X_T^+ + b_{T+1}X_T^- + C^w W_T \quad (4.7)$$

where  $W_T$  represents the ending in process inventory,  $X_T^+$  the positive ending FG inventory, and  $X_T^-$  the negative ending FG inventory in period  $T$ . In the classical uncapacitated single-item multi-period inventory problem, appropriate selection of the terminal cost function allows a myopic base stock policy to be optimal under time-stationary costs and probability distributions. However, the state-dependent nature of our problem makes it difficult to obtain this type of result.

## 4.2 Policy optimality

### 4.2.1 $P$ as action

The finite horizon, production inventory problem using  $P$  as action is shown in (4.3). In Chapter 3, we showed the single-period problem is convex in both  $R$  and  $P$  as decision variables. Here we assume a convex terminal cost function  $\Theta_{T+1}(\cdot)$ . We re-write  $\Theta_t(X_{t-1}, W_{t-1})$  to separate terms that depend on  $P_t$  from those that depend on  $\in \{X_{t-1}, W_{t-1}\}$  as follows.

$$\begin{aligned} \Theta_t(X_{t-1}, W_{t-1}) = \min_{P_t \geq 0} & \left\{ C_t^p P_t + C_t^w \{f^{-1}(P_t) - P_t\} + C_t^r \max\{0, f^{-1}(P_t) - W_{t-1}\} \right. \\ & \left. + g(X_{t-1} + P_t) + \mathbb{E}_{\mathbb{D}} \left[ \Theta_{t+1}(X_{t-1} + P_t - \mathbf{D}_t, f^{-1}(P_t) - P_t) \right] \right\} \quad (4.8) \end{aligned}$$

It is easy to argue that, if  $\Theta_{t+1}(\cdot)$  is convex, then  $\Theta_t(X_{t-1}, W_{t-1})$  is convex for every  $t$ . We prove this recursively in each period  $t$ , showing that if  $\Theta_{T+1}(\cdot)$  is convex, then  $\Theta_t(X_{t-1}, W_{t-1})$  are convex.

**Lemma 4.2.1.** *If  $\Theta_{T+1}$  is a convex function in  $P_T$ , then:*

1.  $\Theta_T(X_{T-1}, W_{T-1})$  is convex in  $P_T$  for period  $T$ .
2. Any minimizer of  $\Theta_T(X_{T-1}, W_{T-1})$  implies  $\Theta_t(X_{t-1}, W_{t-1})$  is convex in  $P_t$  and the optimal production is  $P_t^*$ .

*Proof.* 1. Suppose  $\Theta_{T+1}$  is a convex function, and we assume stationary costs across all periods. The term  $C^w\{f^{-1}(P_T) - P_T\}$  represents the ending WIP holding cost and the term  $C^r \max\{0, f^{-1}(P_T) - W_T\}$  represents the cost of releases for period  $T$ . The first term  $C^p P_T$  is linear. The clearing function  $f(\Lambda_T)$  is a monotonically non-decreasing concave function. By assumption, in  $P$  as action, we use  $P_T \leq f(\Lambda_T)$ . The CF constraint takes the form  $P_T - f(\Lambda_T) \leq 0$ . Since  $f(\cdot)$  is concave,  $-f(\cdot)$  will be a convex feasible region. We know that the inverse of a concave function is convex (Boyd et al. 2004). Therefore,  $f^{-1}(P_T) - P_T$  is convex in  $P_T$ . Thus,  $C^r \max\{0, f^{-1}(P_T) - W_T\}$ , and  $C^w\{f^{-1}(P_T) - P_T\}$  are both convex in  $P_T$ . Since  $h$  and  $b > 0$ , the second derivative of  $h \mathbb{E}[X_{T-1} + P_T - \mathbf{D}_T]^+ + b \mathbb{E}[\mathbf{D}_T - X_{T-1} - P_T]^+$  is  $(h + b)\phi(X_{T-1} + P_T) \geq 0$ . Thus,  $g(\cdot)$  is convex in  $X_{T-1} + P_T$ , implying it is convex in  $P_T$  since the composition of convex functions is convex (Boyd et al. 2004). By assumption,  $\Theta_{T+1}(\cdot)$  is a convex function and  $D_T$  is a random variable, so  $\mathbb{E}[\Theta_{t+1}(X_{T-1} + P_T - \mathbf{D}_T, f^{-1}(P_T) - P_T)]$  is convex since both expectation and the maximum operator preserve convexity (Bertsekas et al. 2003). Finally, since the sum of convex functions is convex (Bazaraa et al. 2013; Dimitri et al. 1999), the right-hand side objective function is convex in  $P_T$  for period  $T$ .

2. Since minimization preserves convexity (Boyd et al. 2004),  $\Theta_T(X_{T-1}, W_{T-1})$  is convex in  $P_T$ , then  $P_T^*$  is optimal in period  $T$ . From part 1 of Lemma 4.2.1,  $\Theta_{T-1}(\cdot)$  is convex and a  $P_{T-1}^*$  is optimal for period  $T - 1$ . The same procedure can be recursively applied from period  $T$  to period 1. Therefore,  $\Theta_t(X_{t-1}, W_{t-1})$  is convex in  $P_t$  and the corresponding optimal production is  $P_t^*$ . ■

Using standard SDP, we solve the Bellman equation by taking argmin of (4.3) to compute  $P^*$ . This is computationally expensive as we must enumerate all possible actions for a particular state to find the optimal  $P_t^*$  that minimizes (4.3) in every period  $t$ . However, from Lemma 4.2.1, the objective function is convex in  $P_t$ . The convexity of the objective function allows us to use bisection search to solve for  $P_t^*$  numerically, eliminating the need for explicit enumeration of the actions  $P_t$  and saving significant computation

time. We discuss the details of the binary search in Section 4.4.3. Another interesting finding is that, unlike classical inventory problems, where a myopic policy is optimal for time-stationary demand distribution with appropriate terminal cost conditions, this production inventory problem follows a state-dependent policy that does not yield the state-independent order up to the level encountered in the versions of the problem without congestion. Hence, we have not been able to obtain a myopic policy even with a time-stationary demand distribution across all periods.

### 4.2.2 $R$ as action

We show a similar approach for the finite horizon production inventory problem using  $R$  as an action using (4.5). We allow the terminal cost  $\Theta_{T+1}(\cdot)$  to be non-zero and assume that it is convex.

$$\Theta_t(X_{t-1}, W_{t-1}) = \min_{R_t \geq 0} \left\{ C_t^w(W_{t-1}) + G_t(R_t) \right\} \quad (4.9)$$

where

$$\begin{aligned} G_t(R_t) = & C_t^p f(R_t + W_{t-1}) + C_t^w [R_t - f(R_t + W_{t-1})] + C_t^r(R_t) \\ & + g[X_{t-1} + f(R_t + W_{t-1})] + \mathbb{E}_{\mathbb{D}} \left[ \Theta_{t+1}(X_{t-1} + f(R_t + W_{t-1}) - \mathbf{D}_t, W_{t-1} + R_t - f(R_t + W_{t-1})) \right] \end{aligned} \quad (4.10)$$

Following the same lines as the previous section, it is easy to argue that, if  $G_t(R_t)$  is convex, then  $\Theta_t(X_{t-1}, W_{t-1})$  is convex for every  $t$ . We prove this recursively in each period  $t$ , showing that if  $\Theta_{t+1}(\cdot)$  is convex, then  $G_t(R_t)$  and  $\Theta_t(X_{t-1}, W_{t-1})$  are convex.

Similar to the case with  $P$  as action, the  $R$  as action-based SDP procedure fails to produce a myopic policy for time stationary demand distribution for a specific terminal cost function. Since it is state-dependent, a modified order up to the level policy is impossible. The optimal value of the  $P$  as action SDP is a lower bound for the problem with  $R$  as action. The key assumption is the relationship between the actual production  $P$  and the workload  $\Lambda$ . In  $R$  as action, we assume  $P_t = f(\Lambda)$ , whereas, in  $P$  as action, we assume  $P_t \leq f(\Lambda)$  and produce  $P_t^*$  to minimize the total expected costs, with optimal releases calculated from  $P^*$ . Thus  $P$  as an action-based SDP is a relaxation of  $R$  as an action-based SDP.

## 4.3 Demand Discretization

Demand is a non-negative, independent random variable in each period following a probability distribution with pdf  $\phi_t$  and CDF  $\Phi_t$  that must be discretized to calculate the expected future cost for the finite horizon problem. Several authors have proposed discretization algorithms to approximate a continuous distribution with a discrete mass function. Brown (1959) discusses approximating a continuous distribution with a discrete distribution. Miller III and Rice (1983) use a Gaussian quadrature method to determine the approximate discrete probability distribution. Barbiero and Hitaj (2022) use a  $k$ -fixed number of approximating points based on minimizing the distance between distribution functions to build a discrete distribution as an approximation. We thus get  $k$  discrete values and their associated probabilities in the form of a probability mass function (PMF).

In this dissertation, we use an approximation method based on the midpoint numerical approximation of the cumulative distribution function. We illustrate an algorithm to discretize a probability density function demand to a PMF, assuming that the discretized demand values take integer values for simplicity. Refer to Appendix A.1 for the algorithm details and a sample illustration. Thus, we can discretize any continuous distribution into a probability mass function (pmf) and cumulative distribution function (CDF) using the listed procedure. If the given input demand distribution is discrete, we can skip the discretization step in the SDP algorithm and take the demand input directly to solve the Bellman equation.

## 4.4 Stochastic Dynamic Programming Algorithm

### 4.4.1 Stochastic Dynamic Programming using $P$ as action

We use a backward induction algorithm to solve the Bellman equation shown in Section 4.1. Using (4.3), we define a recursive scheme to determine the optimal action for each state in each period  $t$ . The key step in this algorithm is the main recursion loop, where we use  $P$  as an action such that we can produce less than the maximum output permitted by the CF with the available workload. i.e.,  $P_t \leq f(W_{t-1} + R_t)$ . This approach matches the situation considered in the SPP-NLP discussed in Chapter-3, extending the SPP solution method to a multi-period problem using the SDP algorithm.

We need to discretize and truncate the state space for the computational implementation. We truncate the minimum, and the maximum limit for the FGI states using upper

bounds on demand and production in each period, denoted by  $D_{max}$  and  $P_{max}$ , respectively. The minimum state limit is then calculated assuming a demand value of  $D_{max}$  in all periods  $t = 1, 2, \dots, T$  such that the worst-case possible FGI state would be  $-D_{max}T$ . Similarly, the maximum possible value of  $X_t$  is calculated assuming production of  $P_{max}$  in all periods such that the maximum possible replenishment occurs in all periods. If the initial inventory is zero, the worst-case scenario to get the minimum ending inventory level is given by  $X_{min} = -D_{max}T$ . On the other hand, the maximum inventory level is calculated by the product of maximum production and the number of periods. i.e.,  $X_{max} = P_{max}T$ .

To define the state and action spaces for  $W_{t-1}$  and  $P_t$ , we define two parameters  $A$  and  $B$ , such that the range of  $W$  lies between 0 and  $P_{max} + B$ . Likewise, the range for action  $P$  lies between 0 and  $P_{cap} = P_{max} - A$ . The parameter  $A$  is determined based on the size of the grid, while  $B$  is calculated as the workload required to meet a specified fraction of the maximum output the CF can achieve with an infinite workload. In our experiments, we truncate the state space for  $W$  to the value of workload at which 90% of the maximum possible production is achieved.

For example: using the Missbauer CF with parameters  $K_1 = 10$  ( $P_{max}$ ), and  $K_2 = 10$ , 90% of the maximum possible production is achieved when the workload is 99. So, we choose  $B = 89$  to truncate the WIP state space. For the action space, if grid size 0.1 is selected, then  $A = 0.1$  and  $P_{cap}$  becomes 9.9 for  $P_{max} = 10$ . Thus production values  $P_t$  range between 0 and 9.9, in increments of 0.1 will be enumerated in the standard SDP algorithm, and the 2-dimensional system state consists of  $(X, W)$  with  $X = X_{min}, \dots, X_{max}$  and  $W = 0, \dots, W_{max}$  respectively. The optimal policy must specify the optimal action  $P_t$  for each state  $(X_{t-1}, W_{t-1})$ . Computing the optimal policy for a  $T$  period finite horizon problem will require  $\mathcal{O}((X_{max} - X_{min})W_{max}P_{cap}T)$  arithmetic operations. The implementation of the SDP algorithm with  $P$  as action is described in Algorithm 2.

---

**Algorithm 2** SDP for finite horizon inventory problem using Production as action

---

```
1: Initialize parameters:  $\delta$ , accuracy,  $T$ ,  $P_{max}$  and CF parameters  $K_1, K_2$ 
2: Initialize costs:  $h, b, C^p, C^r, C^w$  ▷ assumed stationary costs
3: Initialize parameters:  $A, B$  ▷ to define  $P_{max}$  and  $W_{max}$  boundaries
4: Let the non-stationary demand distribution as  $D_t$  ▷ discrete distribution
5: In case of a continuous distribution, discretize using Algorithm 4
6: Calculate state space boundaries as follows:
7:  $X_{min} = -D_{max}T$  ▷  $D_{max}$  = max. demand in all periods
8:  $X_{max} = P_{max}T$  ▷  $P_{max}$  = maximum production capacity
9:  $W_{max} = P_{max} + B$  ▷ B is determined based on 90% of  $P_{max}$ 
10:  $P_{cap} = P_{max} - A$  ▷ A is determined by the grid size
11: Let  $t=1,2,\dots, T$  ▷ planning period
12: Terminal costs calc.:
13: for  $x = [X_{min}, \dots, X_{max}]$  do
14:   for  $w = [0, \dots, W_{max}]$  do
15:     Compute  $\Theta_{T+1}(x, w)$  using Equation (4.7)
16:   end for
17: end for
18: SDP Main loop:
19: for  $t = T, \dots, 1$  do
20:   for  $x = [X_{min}, \dots, X_{max}]$  do
21:     for  $w = [0, \dots, W_{max}]$  do
22:       for  $p = [0, \dots, P_{cap}]$  do
23:         Compute  $R = \max\{f^{-1}(p) - w, 0\}$ 
24:         Compute  $W = \max\{w + R - p, 0\}$ 
25:         Compute  $\Theta_t(x, w)$  using Equation (4.3) ▷ DP recursion
26:          $P_t(x, w) \leftarrow \underset{P}{\operatorname{argmin}}$  of Equation (4.3)
27:       end for
28:     end for
29:   end for
30: end for
31: return  $\Theta_t(x, w), P_t(x, w) \quad \forall t, x, w$ 
32: Print the optimal policy
```

---

### 4.4.2 Stochastic Dynamic Programming using $R$ as action

A slight modification allows us to adapt the SDP algorithm of the previous section to the situation where releases, not production, are the principal decision variable. Most of the algorithm steps are similar to Algorithm 2 except for the main recursion loop where we use  $R$  as an action such that the production is defined by the CF. i.e.,  $P_t = f(W_{t-1} + R_t)$ . This relationship suggests that the initial WIP is consumed first before adding additional release to the workload to produce output. Thus, in  $R$  as an action-based SDP, the ending WIP is calculated as  $W_t = W_{t-1} + R_t - P_t$  based on the assumption that  $\frac{\partial f}{\partial \Lambda} \leq 1$  for all  $\Lambda \geq 0$ .

An advantage of using  $P$  as an action over  $R$  as an action is the computational runtime savings. In  $P$  as an action, we iterate over a smaller action space compared to  $R$  as an action, and hence the algorithm terminates faster. This is due to the CF-based production requiring a large workload for high levels of production as determined by the shape of the CF. For instance, for a CF with a maximum production capacity of 10, production can take values between 0 and 10, and even if we use 1 decimal place in the action space, we need to enumerate only 101 actions. However, if the action space involves a decimal fraction for  $R$  as an action, it will lead to roughly 1000 actions. Therefore,  $P$  as an action-based SDP algorithm is economical and efficient in benchmarking against approximation algorithms. In addition, the  $P$  as action model is a relaxation of the  $R$  as action model; any feasible solution for  $R$  as action can be obtained by  $P$  as action, but not the converse. Hence the optimal value of the problem with  $P$  as action is a lower bound on that with  $R$  as action.

### 4.4.3 A binary search approach to compute optimal actions

From Lemma 4.2.1, the objective function is convex in  $P_t$ . In standard SDP, we solve the Bellman equation by enumerating all possible actions for each state in period  $t$ , which is computationally burdensome. By exploiting the policy structure and the convexity property, we introduce an efficient approach that combines the SDP algorithm with a binary search strategy. We also leverage the convex structure of the objective function, which implies that once  $P^* = 0$  for a given state  $(X, W)$ , the optimal action for larger values of FGI and WIP remains the same. Hence, we no longer need to enumerate  $P$  for larger values of  $X$  and  $W$ . Under such cases, we skip the binary search procedure and set  $P^* = 0$ .

The binary search (BS) procedure is shown below in Algorithm 3. We use two parameters, tolerance, and step size, to fine-tune the accuracy of the expected cost and  $P^*$ . We use perturbation with a step size to search near the middle value of  $P^* = 0$  and  $P^* = P_{cap}$  such that the search will update the  $P^*$  in each iteration. The binary search subroutine replaces the innermost for-loop in Algorithm 2. Figure 4.1 shows the convex objective function and the corresponding  $P^*$  for each of the three cases as an illustration: 1) Green curve with  $P^* = 0$  as "do nothing" policy, 2) Blue curve with  $P^*$  given by the argmin of (4.3), 3) Orange curve indicates  $P^* = P_{cap}$  when a significant amount of workload is required to produce FG.

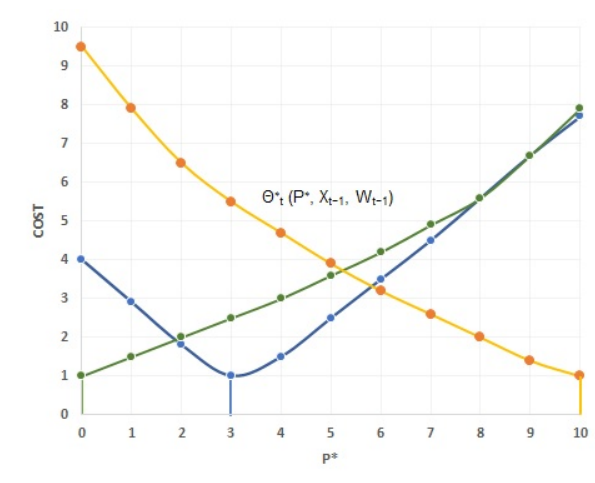


Figure 4.1:  $\Theta_t(X_{t-1}, W_{t-1})$  is convex in  $P_t$ ; (a) Green color with  $P^* = 0$ , (b) Blue curve with  $P^*$  given by argmin of (4.3), (c) Orange curve indicates  $P^* = P_{cap}$

The production values  $\bar{p}$  range between 0 and  $P_{cap}$ , in increments of grid size in the standard SDP algorithm. The binary search procedure reduces the runtime of the innermost loop, and the worst-case computational time complexity reduces from  $\mathcal{O}(\bar{p})$  to  $\mathcal{O}(\log_2 \bar{p})$ , where  $\bar{p}$  represents the number of actions in the inner for-loop in Algorithm 2. We refer to Lin (2019) for convergence and performance of the binary search algorithm. We discuss the results comparing the SDP performance in the computational experimentation in Section 4.7.

---

**Algorithm 3** Binary Search Algorithm for  $P$  as action

---

```
1: procedure BINARY SEARCH( $X, W, costs, parameters, P$ )
2:   Initialize parameters: step size, and tolerance ▷ tol=0.001, step=0.01
3:    $wq, diff, pmid \leftarrow M$  ▷ M as large number
4:    $A$  as parameter ▷ defined by the grid size, e.g., 0.1 for a grid size of 0.1
5:   if  $pmid == 0 \parallel w \geq wq$  then
6:     if  $pmid == 0 \ \& \ wq > w$  then
7:        $wq \leftarrow w$ 
8:     end if
9:      $pmid \leftarrow 0$ 
10:     $P^* \leftarrow pmid$ 
11:     $TCopt \leftarrow \Theta_t^*(P^*, X_{t-1}, W_{t-1})$ 
12:  else
13:     $pleft \leftarrow 0$ 
14:     $pright \leftarrow P_{max} - A$  ▷ Always  $A$  defined by grid size
15:     $flag \leftarrow 0$ 
16:    while  $pmid > tol$  &  $diff > tol$  do ▷ Main loop
17:      mid value:
18:       $u \leftarrow pmid$ 
19:       $pmid \leftarrow pleft + (pright - pleft)/2$ 
20:       $diff \leftarrow abs(u - pmid)$ 
21:      perturbation:
22:       $leftm \leftarrow pmid - step$ 
23:       $rightm \leftarrow pmid + step$ 
24:      cost calculation:
25:      Calculate  $TCl \leftarrow \Theta_t(leftm, X_{t-1}, W_{t-1})$ 
26:      Calculate  $TCr \leftarrow \Theta_t(pmid, X_{t-1}, W_{t-1})$ 
27:      Calculate  $TCm \leftarrow \Theta_t(rightm, X_{t-1}, W_{t-1})$ 
28:      main loop:
29:      if  $TCr \leq TCm \ \& \ TCm < TCl$  then
30:         $pleft \leftarrow pmid$ 
31:      else if  $TCl \leq TCm \ \& \ TCm < TCr$  then
32:         $pright \leftarrow pmid$ 
33:      else
34:         $flag \leftarrow 1$ 
35:        if  $P^* == 0$  then
36:           $P^* \leftarrow round(pmid, 0)$ 
37:        end if
38:         $P^* \leftarrow round(pmid, 2)$ 
39:         $TCopt \leftarrow \Theta_t^*(P^*, X_{t-1}, W_{t-1})$ 
40:      end if
41:    end while
42:    if  $flag == 0$  then
43:      if  $P^* == 0$  then
44:         $P^* \leftarrow round(pmid, 0)$ 
45:      end if
46:       $P^* \leftarrow round(pmid, 2)$ 
47:       $TCopt \leftarrow \Theta_t^*(P^*, X_{t-1}, W_{t-1})$ 
48:    end if
49:  end if
50:  return  $P^*$  and  $\Theta_t^*(P^*, X_{t-1}, W_{t-1})$ 
51: end procedure
```

---

## 4.5 Value Function Approximation

While solving the Bellman equation, an action taken in an initial state may lead to an ending state that is not part of the discretized state space  $(X, W)$ . Thus, the future expected cost operator may not be able to identify the appropriate cost-to-go from the calculations at previous iterations, which are recorded at discrete grid points. The lookup table for a given state  $(X, W)$  in period  $t$  describing the optimal policy consists of  $P^*$  and the expected cost-to-go for that state  $(X, W)$ . While calculating the expected cost of a given state in period  $t$ , we refer to this look-up table to get the expected cost in period  $t + 1$ . The following equations determine the ending FG and WIP levels, and hence the state transitions.

$$X_t = X_{t-1} + P_t - \mathbf{D}_t \quad (4.11)$$

$$W_t = W_{t-1} + R_t - P_t \quad (4.12)$$

Since  $P_t$  can take fractional values, the ending FG inventory and WIP inventory can be fractional. Thus the state transition equations do not preserve the discretization grid from one stage to the next, forcing us to approximate the expected future cost when the ending state of  $(X, W)$  is fractional. We describe two approximation techniques, truncation, and bilinear interpolation for the value function in the next subsections. Finding the right value function approximation determines the quality of the policy returned by the dynamic program. Several value function approximations have been proposed in the literature, including linear combinations of features, neural networks, decision trees, nearest neighbors, stochastic gradient descent, and Fourier/wavelet bases (Bertsekas 1995; Bertsekas and Tsitsiklis 1996; Powell 2007).

### 4.5.1 Truncation approximation

The Truncation approximation (Bertsekas and Tsitsiklis 1996) uses the value of the output cell by approximating it to the nearest cell (lower) to the input grid. i.e., from the current state to the nearest (lowest) possible state defined in the state space definition. The expected cost-to-go  $\Theta_{t+1}(X, W)$  is approximated by  $\Theta_{t+1}(X', W')$ , where  $X' = \max[X \in G, x \leq X]$ , and  $W' = \max[W \in G, w \leq W]$  and set  $G$  denotes the set of  $X$  and  $W$  values in the discretization grid. A simple illustration is shown below to explain the approximation technique. Let the state space  $(X, W)$  be defined to 1 decimal place.

$$X = [\dots, 1.8, 1.9, 2, 2.1, 2.2, 2.3, 2.4, \dots]$$

$$W = [\dots, 8, 8.1, 8.2, 8.3, 8.4, \dots]$$

Let the current state of  $\mathbb{E}(\cdot)$  be (2.34, 8.28). According to the truncation approximation method, the value function is approximated to the state (2.3, 8.2) based on the state space definition listed above. This method is simple to implement and reduces the run time. However, the accuracy will depend on the grid size in the state-space definition.

## 4.5.2 Bilinear Interpolation

Since the expected cost function is convex and continuous, one can interpolate using the expected cost from the look-up tables based on a discrete grid with relative safety (Hadley 1965). Our problem requires two-dimensional states, hence it is necessary to interpolate in tables involving two arguments (Cai 2009; Hadley 1965). For instance, suppose we want to find the approximate value function  $f(x, w)$ , we take the four neighbor points at  $(x_1, w_1)$ ,  $(x_1, w_2)$ ,  $(x_2, w_1)$ , and  $(x_2, w_2)$  such that  $x_1 \leq x \leq x_2$ , and  $w_1 \leq w \leq w_2$ . The corresponding value functions are represented as  $f(x_1, w_1)$ ,  $f(x_1, w_2)$ ,  $f(x_2, w_1)$ , and  $f(x_2, w_2)$  respectively.

Using linear interpolation in the  $X$ -direction, we get

$$Y_1 \approx f(x, w_1) \approx \frac{x_2 - x}{x_2 - x_1} f(x_1, w_1) + \frac{x - x_1}{x_2 - x_1} f(x_2, w_1), \quad (4.13)$$

$$Y_2 \approx f(x, w_2) \approx \frac{x_2 - x}{x_2 - x_1} f(x_1, w_2) + \frac{x - x_1}{x_2 - x_1} f(x_2, w_2), \quad (4.14)$$

Now, we interpolate in the  $W$ -direction to obtain the desired estimate.

$$Y \approx f(x, w) \approx \frac{w_2 - w}{w_2 - w_1} Y_1 + \frac{w - w_1}{w_2 - w_1} Y_2 \quad (4.15)$$

Substituting (4.13) and (4.14) in (4.15) gives the desired result in a single matrix formula,

$$Y \approx f(x, w) \approx \frac{1}{(x_2 - x_1)(w_2 - w_1)} \begin{bmatrix} x_2 - x & x - x_1 \end{bmatrix} \begin{bmatrix} f(x_1, w_1) & f(x_1, w_2) \\ f(x_2, w_1) & f(x_2, w_2) \end{bmatrix} \begin{bmatrix} w_2 - w \\ w - w_1 \end{bmatrix} \quad (4.16)$$

For instance, let the calculated  $(x, w)$  state be (3.25, 1.28). Assuming an integer grid, we need to find an approximation from the expected cost to go computed for period  $t + 1$ . Let  $x = 3.25$  and  $w = 1.28$ . Therefore, we define  $x_1 = 3$  and  $x_2 = 4$  based on the

$x$ -value and  $w_1 = 1$  and  $w_2 = 2$  based on the  $w$ -value. The corresponding value function values are  $(x_1, w_1)$ ,  $(x_1, w_2)$ ,  $(x_2, w_1)$  and  $(x_2, w_2)$  are  $f(x_1, w_1) = 31.4$ ,  $f(x_1, w_2) = 33.4$ ,  $f(x_2, w_1) = 41.2$ , and  $f(x_2, w_2) = 43.2$ . We substitute the above values in (4.16) and calculate  $f(3.25, 1.28) = 34.41$ . Since the Bellman equation is convex in decision variables  $R$  and  $P$ , by approximating the value function using bilinear interpolation seems promising, and gives an upper bound on the true value. The error of approximation would not result in a large deviation from the expected cost.

## 4.6 Model Assumptions and Implementation issues

We code the algorithm in a MATLAB R2019 (MathWorks 2005) environment. The optimal policy is analyzed using MS Excel 2016 as a post-processing step. We also used Microsoft SQL server scripting to analyze the policy when defining the initial state spaces with decimal places. As highlighted, we need faster computing technology since the state space grows rapidly. Therefore, we apply a parallel computing framework using MATLAB’s parallel computing (PC) toolbox (MathWorks 2020; Sharma and Martin 2009).

We develop a MATLAB script as a subroutine to find the discrete demand distribution as shown in Algorithm A.4, given a mean and variance from the Normal distribution. Note that this procedure is a new method of discretizing a continuous distribution into a discrete distribution, based on a specific implementation of midpoint-based approximation using the cumulative distribution function. After discretization, we verified the sum of all probabilities corresponding to the discrete distribution and found close to 99.9999%. For example, we observe more than 99.9999% sum of probabilities for all demand values in each discretization of mean and standard deviation across all experiments. Table A.3 summarizes the mean and standard deviation of the discretized distribution for the 10-period problem with time-varying demand distribution.

## 4.7 Computational Experiments and Results

### 4.7.1 Experimental Design

We first consider a 10-period problem and discretize the continuous demand distribution shown in Table 4.1, and 4.2 using Algorithm A.4. The average demand over the finite horizon is five units per period and eight units per period across all demand patterns. We use 16 experiments consisting of 4 demand patterns with two levels of squared coefficient of variation, and two levels of backorder cost. We assume demand in each period follows

a Normal distribution. We use the  $P$  as action-based SDP algorithms to test the computation time and compare the difference. As dynamic programming is used to find exact expected values, statistical analysis is not required for this portion of the experimental design.

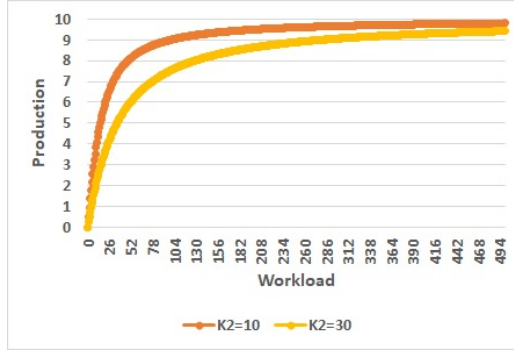
Table 4.1: Demand patterns for an overall mean demand of 5

Variation	1	2	3	4	5	6	7	8	9	10
Mean Rising (MR)	2	3	4	4	5	5	6	6	7	8
Mean Decline (MD)	8	7	6	6	5	5	4	4	3	2
Mean Constant (MC)	5	5	5	5	5	5	5	5	5	5
Mean Peak (MP)	4	4	4	4	5	8	9	4	4	4

Table 4.2: Demand patterns for an overall mean demand of 8

Variation	1	2	3	4	5	6	7	8	9	10
Mean Rising (MR)	4	5	6	7	8	8	9	10	11	12
Mean Decline (MD)	12	11	10	9	8	8	7	6	5	4
Mean Constant (MC)	8	8	8	8	8	8	8	8	8	8
Mean Peak (MP)	7	7	7	7	12	12	7	7	7	7

We begin by taking  $C^r = 1$ ,  $C^w = 0.25$ ,  $C^p=2$ ,  $h = 0.3$  and  $b \in \{6, 30\}$ . We test these models under two different coefficients of variation (SCV) (16%, 4%). For simplicity of exposition, we use zero terminal cost throughout all our experiments in order to focus on the CF-based replenishment and structural properties of the optimal solution. We use a Missbauer CF shown in Figure 4.2 with  $K_1 = P_{max} = 10$  and values of the shape parameter  $K_2$  of 10 and 30. Note that the workload requirement differs between the CFs; the Missbauer CF with  $K_2 = 10$  requires a workload of 99 units to produce 9 units compared with a workload of 279 units when  $K_2 = 30$ . Since we run  $P$  as an action-based SDP algorithm shown in Algorithm 2, we initially limit the maximum WIP to 99 in case of  $K_2 = 10$  and 279 units in case of  $K_2 = 30$ . Preliminary results suggest that we cannot truncate the  $W$  state space when the utilization is high in a flat CF. We investigate the effect of  $W$  state space truncation by calculating the workload required to cover 75% and 90% of production capacity in Section 4.7.7.



(a)

Figure 4.2: Missbauer CF

A simple experimental design varying four demand patterns with two sets of squared coefficient of variation (SCV) and backorder cost resulted in 16 experiments. Table 4.3 shows the experimental design with mean demand as rising (MR), decline (MD), constant (MC), and peak (MP), respectively. Mean demand with respect to the maximum production capacity gives utilization ranges from 50% and 80%, respectively. Unless otherwise mentioned in the text, we use 36 cores in a parallel computing framework to run the DP recursion in parallel.

Table 4.3: Parameter settings

Backorder cost	6	30		
SCV	0.16	0.04		
Demand Pattern	Rising	Decline	Constant	Peak

We run the original SDP and BS-based SDP algorithm using  $P$  as an action and compare the results regarding computation run time, policy changes, and objective function cost for the state  $(0, 0)$  in period 1. We use two levels of VFA: Bilinear interpolation (BILSDP) and Truncation approximation (TruncSDP) and discuss the impact of VFA and its effect on the expected cost and computation time. We summarize the experimental design using VFA and grid size for state and action space in Table 4.4.

Table 4.4: Experimental design using grid size and VFA

Value Function Approximation	BILSDP	TruncSDP	
Grid size for state space	1	0.5	
Grid size for action space	1	0.5	0.1

For the original SDP, we use two levels of state space discretization  $\in \{1, 0.5\}$  and three levels of action space discretization  $\in \{1, 0.5, 0.1\}$  to study the effect of grid size on expected cost and solution time. We compare the expected cost of the original SDP model with the BS-based SDP models of grid size  $\in \{1, 0.5\}$  in state space. Since the binary search algorithm produces the optimal action for a continuous problem, the action space discretization is irrelevant in the BS-SDP algorithm. We compare the expected cost and the computation time as metrics. We summarize the results and examine the effect of discretization, the effect of cost parameters, and the CF's curvature in finding the optimal policy.

The computer used for the experiments ran on two setups: An Intel Xeon processor, 2.1 GHz with 128 GB (72 cores and 36 cores) RAM, and an Intel Xeon processor, 2.1 GHz with 712 GB (32 cores) RAM. The computational run time depends on four factors: state space, action space, number of periods, and the grid size in state and action space discretization. We used 32, 36, and 72-core machines to save on run time and deployed a parallel computing framework across all SDP experiments. Using the original SDP shown in Algorithm 2, we solve the multi-period problem using a predefined state and action space with a specific grid size. For instance, When the number of cores is 36, using  $P$  as an action-based SDP algorithm, using an integer grid size in state and action space, on average, we get 1.42 minutes for a 10-period problem. However, when the grid size becomes finer, for instance, 0.5 grid for state and 0.1 grid for action, on average, we get 40.82 minutes for a 10-period problem. Similarly, on average, we get close to 7.16 minutes for each experiment for 0.5 in state and action space. However,  $R$  as an action-based SDP algorithm involves more computation time. The average run time for an SDP algorithm using integer state and action space was less than 7.45 minutes on a 72-core machine and around 14 minutes on a 32-core machine. However, an Intel Core i7-7500U CPU with a single core and 16 GB RAM took more than 2 hours of run time for each experiment.

#### 4.7.2 Effect of Grid Size in Conventional SDP

We compare the effectiveness of the binary search algorithm with the standard SDP algorithm. Using the standard SDP, as shown in Algorithm 2, we conduct experiments by varying state space grid size  $\in \{0.5, 1\}$ , and action space grid size  $\in \{0.1, 0.5, 1\}$ , respectively. Table 4.5 gives the expected cost and computation times for state  $(0, 0)$  in period 1 for mean demand of 5, using relevant costs, and  $K_2 = 10$  in the Missbauer CF in an  $P$  as an action-based original SDP algorithm.

Table 4.5: Expected cost and computational times (minutes) comparison for various experiments using  $P$ -based standard SDP. We show the impact of action space discretization  $\in \{1, 0.5, 0.1\}$  and state space discretization  $\in \{1, 0.5\}$  with mean demand of 5,  $K_2=10$ . We use bilinear interpolation as VFA on a 36-core CPU. We denote the expected cost for the grid size in state and action as (state grid, action grid)

Expt #	Demand	SCV	$b$	(1, 1)		(1, 0.5)		(1, 0.1)		(0.5, 0.5)		(0.5, 0.1)	
				cost	time	cost	time	cost	time	cost	time	cost	time
E1	MR	0.16	6	210.14	1.1	209.66	2.2	209.32	12.6	209.51	10.2	209.16	53.7
E2	MD	0.16	6	254.59	1.6	249.39	2.2	246.68	12.1	248.88	8.3	246.13	49.6
E3	MC	0.16	6	208.99	1.3	208.08	1.7	207.38	8.9	207.80	5.8	207.14	35.5
E4	MP	0.16	6	216.85	2.8	215.81	2.4	215.35	13.8	215.58	8.8	215.12	57.7
E5	MR	0.04	6	198.51	1.4	198.17	1.7	197.92	8.7	198.02	5.7	197.76	33.8
E6	MD	0.04	6	217.20	1.4	213.20	2.0	210.74	8.7	212.16	5.9	209.98	33.2
E7	MC	0.04	6	192.15	1.2	191.90	1.7	191.66	6.7	191.67	4.5	191.44	24.8
E8	MP	0.04	6	198.63	1.3	198.08	2.0	197.71	9.3	197.90	6.3	197.43	35.0
E9	MR	0.16	30	236.67	1.5	235.81	2.5	235.43	12.1	235.55	8.1	235.20	51.3
E10	MD	0.16	30	370.28	1.5	360.97	2.4	352.41	11.7	360.23	8.1	351.44	49.3
E11	MC	0.16	30	230.73	1.3	228.81	1.9	228.01	8.9	228.36	8.5	227.61	33.3
E12	MP	0.16	30	236.93	1.7	235.19	2.8	234.62	14.9	234.86	11.3	234.30	55.8
E13	MR	0.04	30	213.18	1.4	212.55	2.0	212.33	8.8	212.32	6.0	212.07	33.2
E14	MD	0.04	30	259.11	1.4	254.49	1.9	251.49	8.8	253.54	6.4	250.57	34.7
E15	MC	0.04	30	201.89	1.3	200.79	1.6	200.48	6.7	200.45	5.0	200.17	25.5
E16	MP	0.04	30	206.82	1.0	206.10	2.0	205.81	9.2	205.75	6.6	205.46	36.5

The grid size in action space impacts the expected cost very slightly, except in the MD demand pattern. For all other patterns, we observe negligible improvements between the action grid sizes. We have a similar observation when  $K_2 = 30$  in mean demand of 5 and 8. Hence these results are not given here. The finer the action space grid size, the expected cost becomes accurate. However, the impact on the computational effort is substantial, as expected. When the grid size becomes finer in the state and action space, we get negligible improvements similar to an integer grid in the state space. Comparing 0.1 grid in action between the integer and 0.5 grid in state space yields the same results and does not improve the expected cost except the MD demand pattern. Using fine grid size in state space, we realize negligible improvements over the expected cost despite spending a huge amount of computing time.

Similar to the previous analysis, we realize the computation time is burdensome when the grid size of the state and action becomes finer. Based on the tabulated results, we compare the computation times in minutes for both integer and 0.5 grid size in state space by varying the action space between 1, 0.5, and 0.1. We observe a marginal impact on the expected cost between the 0.5 and 0.1 grids on the action space when the state space grid

is 0.5. Comparing the fine action grid size between the integer and 0.5 state space grid, we observe negligible improvements in the expected cost but a substantial computational burden when the grid size changes from 1 to 0.5 on states. Since computation time is a direct consequence of the SDP algorithm’s expected cost accuracy, for benchmarking purposes, we use a 0.1 grid in action from the original SDP. We share the results of the expected cost using a 0.5 grid on state and 0.1 grid on the action as benchmark figures in the Appendix A.2. We now compare the integer and 0.5 grid in state space using the binary search algorithm against the original SDP solution regarding runtime and solution accuracy.

### 4.7.3 Effect of Binary Search in SDP

We use a similar experimental setup to study the effect of binary search compared to the original SDP. The optimal action for each state is approximated to two decimal places defined by the parameter tolerance in Algorithm 3; hence we do not iterate for all values of action space. We compare the expected cost of the BS method with the conventional SDP of the same grid size in state space definition. We compare the benchmark figures of a 0.5 grid in space and a 0.1 grid in action in the standard SDP with that of a 0.5 grid in state space using the BS-based SDP. Similarly, we use an integer grid in space and a 0.1 grid in action from the original SDP with that of an integer grid in the state space of the BS-based SDP. The results are tabulated based on the percentage gap between the BS-based SDP and the original SDP benchmark. We measure the percentage error of the expected cost across different demand patterns between the BS-based SDP and the original SDP algorithms using the following.

$$Error\% = \left( \frac{Cost_{BS} - Cost_{SDP}}{Cost_{BS}} \right) 100\%$$

and the time difference % of computing times using the following.

$$TimeDifference\% = \left( \frac{Time_{StdSDP} - Time_{BSSDP}}{Time_{StdSDP}} \right) 100\%$$

We refer to the standard SDP as StdSDP and the BS-based SDP as BSSDP. We share the tabulated results of the actual cost and the computation times in Appendix A.3. Tables A.6 and A.7 show the expected cost and computing times of the BS-based SDP algorithm in Appendix A.3.

We report the error % and the computation time difference % between the BS-based SDP and the original SDP in Tables 4.6 and 4.7.

Table 4.6: A comparison of expected cost and computing times between BS-based SDP and the standard SDP methods. We use an integer and 0.5 grid on state and a 0.1 grid on an action-based solution from the standard SDP as a benchmark. We use an integer grid and 0.5 grid on state space for mean demand of 5 and  $K_2 = 10$  in the BS-based SDP. We use a 36-core CPU as a computing environment with bilinear interpolation as VFA

Expt #	Demand	SCV	$b$	(1, 0.1)		(0.5, 0.1)	
				cost	time	cost	time
E1	MR	0.16	6	-0.20%	74%	-0.15%	75%
E2	MD	0.16	6	0.11%	72%	0.10%	73%
E3	MC	0.16	6	-0.12%	66%	-0.08%	74%
E4	MP	0.16	6	-0.16%	67%	-0.12%	74%
E5	MR	0.04	6	0.01%	70%	0.02%	74%
E6	MD	0.04	6	0.07%	72%	0.08%	74%
E7	MC	0.04	6	-0.04%	56%	-0.02%	77%
E8	MP	0.04	6	0.01%	61%	0.02%	73%
E9	MR	0.16	30	-0.06%	70%	-0.04%	76%
E10	MD	0.16	30	0.20%	71%	0.19%	76%
E11	MC	0.16	30	-0.02%	73%	0.00%	76%
E12	MP	0.16	30	0.02%	73%	0.03%	75%
E13	MR	0.04	30	0.01%	73%	0.02%	75%
E14	MD	0.04	30	-0.03%	75%	0.03%	76%
E15	MC	0.04	30	0.01%	73%	0.01%	73%
E16	MP	0.04	30	0.00%	50%	0.01%	74%

Based on the experimental evidence, we summarize our findings in the following. 1) We achieve significant computation time savings using BS-based SDP over the original SDP. On average, we get 69% and 75% reduction in computing time when using an integer and 0.5 grid size on state space for  $K_2 = 10$ . Similarly, we achieve a 78% reduction in computing time when  $K_2 = 30$ . This clearly indicates that the BS-based SDP outperforms the conventional SDP method in computing times. 2) When  $K_2 = 10$  (a steep CF), the binary search algorithm yields essentially the same accuracy as the original SDP in most instances of the demand patterns in both solution quality and in much less computing time. We get substantial savings in computation time using both 0.5 and an integer grid on the state space.

Table 4.7: A comparison of expected cost and computing times between BS-based SDP and the original SDP methods. We use an integer and 0.5 grid on state and a 0.1 grid on an action-based solution from the standard SDP as a benchmark. We use an integer grid and 0.5 grid on state space for mean demand of 5 and  $K_2 = 30$  in the BS-based SDP. We use a 36-core CPU as a computing environment with bilinear interpolation as VFA. We use  $W_{max} = 99$  (75% of  $P_{max}$ ) as  $W$  state space truncation in both the BS-based SDP methods instead of 279 units to avoid extra computing times

Expt #	Demand	SCV	$b$	(1, 0.1)		(0.5, 0.1)	
				cost	time	cost	time
E1	MR	0.16	6	4.59%	76%	5.09%	76%
E2	MD	0.16	6	0.82%	77%	0.77%	77%
E3	MC	0.16	6	3.17%	80%	3.14%	77%
E4	MP	0.16	6	2.79%	79%	2.72%	74%
E5	MR	0.04	6	5.14%	79%	5.05%	75%
E6	MD	0.04	6	0.37%	82%	0.33%	78%
E7	MC	0.04	6	4.30%	79%	4.29%	81%
E8	MP	0.04	6	3.26%	78%	3.23%	78%
E9	MR	0.16	30	0.12%	78%	0.07%	75%
E10	MD	0.16	30	-0.30%	79%	-0.32%	76%
E11	MC	0.16	30	-0.18%	79%	-0.13%	76%
E12	MP	0.16	30	-0.05%	79%	-0.09%	76%
E13	MR	0.04	30	-0.02%	78%	-0.04%	77%
E14	MD	0.04	30	-0.18%	78%	-0.15%	79%
E15	MC	0.04	30	-0.11%	78%	-0.12%	82%
E16	MP	0.04	30	-0.13%	79%	-0.14%	79%

3) Despite some savings on run time, we observe some positive errors when the shape of the CF becomes flat and  $b = 6$  shown in Table 4.7. This pattern is similar in other experiments irrespective of grid sizes. In particular, we see a high error of 5.14% (E5) in the MR case of the integer grid and 5.09% (E1) in the MR case of the 0.5 grid, respectively. The reason for this error is that in a flat CF, the production system requires a high workload to produce a given level of output. When the initial FGI is sufficiently large to meet the external demand, we observe a do-nothing policy as optimal as expected. Otherwise, production takes place from the existing WIP instead of additional releases to the system. Since we truncate the initial WIP to workload equivalent of producing only 75% of the production capacity, the production resource produces less output than expected which in turn increases the backorder cost. On the other hand, when  $b = 30$ , the

$b/h$  ratio is 100; additional releases increase the workload to produce the desired output. Therefore, from E9 to E16, we observe less error %. 4) We observe a similar pattern and observation in mean demand of 8, and the shape of the CF is flat and  $b = 6$ ; hence these results are not given here. We conclude that the BS-based SDP algorithm can be used for benchmarking purposes that yield comparable results with the original SDP. In summary, the BS-based SDP method is helpful in solving the multi-period production inventory problem with significantly reduced run time, and the solution quality is at par with the standard SDP procedure.

Despite some significant advantages of using a binary search procedure with the SDP algorithm in a parallel computing framework, we found the implementation using step size selection, and tolerance play a significant role in the computation time. Our implementation follows a similar analysis described by Burden and Faires (1985) and is guaranteed to converge due to the convex nature of the objective function. Based on the mathematical analysis and experimental investigation of the algorithm, we conclude that the solution time of the binary search implementation differs by selecting the appropriate step size and tolerance limit; refer Burden and Faires (1985); Sikorski (1982); Sikorski and Trojan (1990) for convergence and performance analysis. Therefore, for practical purposes, we suggest using a step size of 0.01 and a tolerance of 0.001 instead of higher order precision ( $< 10^{-2}$  step size and  $< 10^{-3}$  tolerance), which will reduce the computation efficiency.

#### 4.7.4 Effect of Binary Search and Parallel Computing

In this experiment, we study the impact of binary search with and without a parallel computing framework and how the optimal policy computation affects the run time. We use 72 cores in the parallel computing framework and set  $C^r = C^p = 0$ ,  $C^w = 0.5$ ,  $h = 1$ ,  $b \in \{9, 19\}$  with  $K_2 = 10$ . We test these models under two different classical newsvendor-based service levels (SL) (95%, 90%) and holding cost ( $h$ ) to determine the backorder cost given by  $b = \frac{hSL}{1-SL}$ . We refer to 72 cores parallel computing as 72coresPC and binary search procedure implemented on a parallel computing framework as BS-72coresPC. NoPC refers to the SDP implementation on a single-core machine without parallel computing. We compare the effects of parallel computing and BS-based SDP algorithm with the standard SDP. The state space  $X$  denotes the number of FGI states, and  $W$  denotes the number of ending WIP states. Since we use  $P$  as an action with an 0.1 grid on the action, we enumerate 99 possible values of  $P \in \{0, 0.1, \dots, 9.9\}$ .

Table 4.8 shows a comparison of computation times using NoPC, 72coresPC, BS-NoPC, and BS-72coresPC using  $P$  as an action. The binary search procedure shows a significant advantage over the original SDP algorithm. When implemented on a parallel computing framework, the algorithm performs well in terms of computational efficiency. For instance, we observe a significant reduction in computing times in MC (E3, E7, E11, E15) demand patterns. The reason is that the do-nothing policy is identified earlier by the BS algorithm, and hence a significant amount of computing is avoided. In addition, we know the maximum demand occurrence in a mean constant demand pattern is smaller, as there are no demand peaks, unlike MD and MP patterns. Hence the state space does not grow to a large size in MC. For instance, in an MD case, we get 311 states in  $X$  for an integer grid. However, in an MC case, we get only 231 states in  $X$ .

Table 4.8: Computational times (seconds) of the 10-period numerical for different demand, SCV, and SL parameters using a Missbauer CF with  $K_2 = 10$ . We use an integer grid on state space and 0.1 grid action space using the  $P$ -based standard SDP algorithm. We use bilinear interpolation as the VFA

Expt #	Demand	SL	SCV	$X$	$W$	NoPC	72coresPC	BS-NoPC	BS-72coresPC
E1	MR	0.95	0.16	311	101	3318	388	661	166
E2	MD	0.95	0.16	311	101	3284	487	607	195
E3	MC	0.95	0.16	231	101	2354	367	338	247
E4	MP	0.95	0.16	341	101	3703	413	775	202
E5	MR	0.95	0.04	261	101	2580	314	453	167
E6	MD	0.95	0.04	261	101	3027	317	404	159
E7	MC	0.95	0.04	201	101	1952	383	236	94
E8	MP	0.95	0.04	281	101	3002	278	595	164
E9	MR	0.9	0.16	311	101	3756	436	635	200
E10	MD	0.9	0.16	311	101	3462	472	666	191
E11	MC	0.9	0.16	231	101	2499	297	478	152
E12	MP	0.9	0.16	341	101	3956	494	870	235
E13	MR	0.9	0.04	261	101	2348	344	447	164
E14	MD	0.9	0.04	261	101	2298	296	433	162
E15	MC	0.9	0.04	201	101	1812	275	291	119
E16	MP	0.9	0.04	281	101	2603	345	624	131

The average computational times of NoPC, 72coresPC, BS-NoPC, and BS-72coresPC are 47.8, 6.15, 8.86, and 2.86 minutes.

On average, we achieve an 82% reduction in run time between NoPC and BS-NoPC,

and using a parallel computing framework; we achieve a 53% reduction between 72coresPC and BS-72coresPC, respectively. The reduction is less in PC due to the extra overhead time spent on allocating parallel cores compared to an efficient binary search procedure that relies on the optimal policy structure and the convexity properties of the objective function. All experiments show a significant improvement in runtime between the 72coresPC and the BS-NoPC runs. This means that the single-core implementation of the BS outperforms the NoPC setup. However, for practical purposes, we recommend using BS-72coresPC implementation to yield results in faster computing times.

#### 4.7.5 Effect of Actions: $R$ versus $P$

In this experiment, we examine the effect of  $R$  and  $P$  as actions while calculating the computation time using an integer grid in actions in the  $R$ -based SDP and 0.1 grid in actions in the  $P$ -based SDP. We set  $C^r = C^p = 0$ ,  $C^w = 0.5$ ,  $h = 1$ ,  $b \in \{9, 19\}$  with  $K_2 = 10$ . We test these models under two different classical newsvendor-based service levels (SL) (95%, 90%). The details of the  $R$  as an action-based SDP and its computing times with and without parallel computing are shown in Appendix A.4.

Table 4.9 summarizes the expected cost and show the gap % between  $R$  and  $P$  as actions for the experimental data under study. We observe slight differences of negligible deviation in the expected cost. The reason for the close costs is due to the value of the cost parameters, which encourage keeping the CF tight at the optimum. Based on the cost comparison, when  $P$  as action is applied, we produce only up to the required quantity given by  $P \leq f(\Lambda)$ . However, when we apply  $R$  as action, we assume all releases and initial WIP are converted to finished goods given by  $P = f(\Lambda)$ . When the demand pattern follows MD and MP, we find a large workload requirement at the beginning of the planning horizon and during the peak season in the middle of the horizon. Since  $P$  as action policy considers all possible values of  $R$  and can produce  $P \leq f(\Lambda)$ , whereas  $R$  action uses all releases, convert into FG by following  $P = f(\Lambda)$ , the expected cost of the system using  $R$  as action is an upper bound to the expected cost of the system using  $P$  as action. This is due to the fact that  $P$  as an action-based SDP is a relaxation of  $R$  as an action-based SDP.

Table 4.9: Expected cost of the 10-period numerical for different demand, SCV, and SL parameters using  $R$  and  $P$  as actions for a Missbauer CF with  $K_2 = 10$ . We use an integer grid on state and action space in the standard SDP algorithm for mean demand of 5 and bilinear interpolation as VFA. Gap % is calculated as  $100\%(Cost_R - Cost_P)/Cost_R$

Expt #	Demand	SL	SCV	$R$ as action Cost	$P$ as action Cost	Gap%
1	MR	95%	16%	118.79	118.68	0.09%
2	MD	95%	16%	158.43	157.02	0.89%
3	MC	95%	16%	112.39	112.35	0.04%
4	MP	95%	16%	129.12	128.17	0.74%
5	MR	95%	4%	90.28	90.00	0.31%
6	MD	95%	4%	111.85	111.15	0.63%
7	MC	95%	4%	77.52	77.43	0.11%
8	MP	95%	4%	96.32	95.74	0.60%
9	MR	90%	16%	102.48	102.28	0.19%
10	MD	90%	16%	118.77	118.19	0.49%
11	MC	90%	16%	96.18	96.18	0.00%
12	MP	90%	16%	107.72	107.35	0.35%
13	MR	90%	4%	81.23	81.02	0.26%
14	MD	90%	4%	91.61	91.35	0.29%
15	MC	90%	4%	72.04	71.99	0.06%
16	MP	90%	4%	85.97	85.71	0.31%

Recall the discussion of using  $R$  versus  $P$  actions in Section 4.4.2; we discuss  $P$  as an action-based SDP algorithm to give computational run-time savings. In a  $P$  as an action-based SDP, the total run time is less than 3 minutes which is a drastic improvement over the  $R$  as an action-based SDP algorithm. This means that in  $P$  as action, we iterate over a smaller action space than  $R$  as action; hence, the algorithm terminates faster. In addition, the state space  $(X_{t-1}, W_{t-1})$  is reduced significantly due to the size of the maximum possible production compared to the maximum possible releases required up to the maximum production capacity bound. Thus, the runtime of  $R$  as the action (as shown in Table A.8) takes longer than  $P$  as an action shown in Table 4.8. Furthermore,  $P$  as an action-based SDP gives a lower bound compared with the  $R$  as an action due to the nature of the production and release assumption. Therefore, for benchmarking purposes, we use  $P$  as an action-based SDP.

#### 4.7.6 Effect of Value Function Approximations

We now discuss the impact of different value function approximations discussed in Section 4.5 and compare the expected cost for the experimental data. The objective of this experiment is two folds: 1) What is the best VFA to find the optimal policy for the finite

horizon multi-period production inventory problem using SDP? For this, we compare the expected cost between the methods for the state  $(0, 0)$  in period 1 and report the gap % shown in (4.17). 2) How much deviation do we get in computation time if we use bilinear interpolation and truncation approximation? To answer these questions, in this section, we present numerical experiments using the following models. BILSDP represents the bilinear interpolation-based SDP, and Truncation represents the truncation approximation denoted by Truncation. In both models, we use the original SDP algorithm and use the above-said methods while implementing the VFA to calculate the expected cost-to-go function. We use two experiments to study the effect of the VFA: 1) An integer grid on state and action space to calculate the optimal policy and computation time in an integer grid size problem with reduced state space problem and 2) 0.5 grid size in the state space and 0.1 grid size in the action space to study the impact of VFA in a fine grid size based large state space problem. We set relevant costs  $C^r = 1$ ,  $C^w = 0.25$ ,  $C^p=2$ ,  $h = 0.3$  and  $b \in \{6, 30\}$ . In addition, we compare the results of two different mean demands that result in 50% and 80% utilization with integer and fractional grid size-based problems. Using  $P$  as an action-based SDP, we summarize the cost for an initial state  $(0, 0)$  in period 1 in each experiment using BILSDP and Truncation approximation. The results are tabulated based on the percentage gap between the BILSDP and the Truncation-based original SDP method. We measure the percentage error of the expected cost across different demand patterns between the two VFA methods in the original SDP algorithm.

$$Gap\% = \left( \frac{Cost_{Truncation} - Cost_{BILSDP}}{Cost_{Truncation}} \right) 100\% \quad (4.17)$$

Table 4.10 shows the expected cost and computational times comparison between BILSDP and Truncation approximation. The details of the results are shown in the Table caption.

Table 4.10: Comparing the expected cost and computational times (minutes) between the BILSDP and Truncation approximation using  $P$ -based standard SDP algorithm. We report results for grid size (1, 1) using mean demand of 5 and  $K_2 = 10$ , and for grid size (0.5, 0.1), we report the results for mean demand of 8 using  $K_2 = 10$ . We use  $W_{max} = 99$  covering 75% of  $P_{max}$  as  $W$  state space truncation. We calculate the gap% for the expected cost between the two VFAs

Expt #	Demand	SCV	$b$	(1, 1)				(0.5, 0.1)				Comparison	
				BILSDP		Truncation		BILSDP		Truncation		(1, 1)	(0.5, 0.1)
				cost	time	cost	time	cost	time	cost	time	gap%	gap%
E1	MR	0.16	6	210.14	1.07	211.05	1.08	406.21	97.6	408.9	61.50	0.43%	0.66%
E2	MD	0.16	6	254.59	1.63	254.67	1.07	753.09	97.9	776.3	60.37	0.03%	2.99%
E3	MC	0.16	6	208.99	1.28	210.02	0.87	487.45	67.2	496.8	40.73	0.49%	1.88%
E4	MP	0.16	6	216.85	2.80	217.25	1.23	505.01	97.5	514.5	60.88	0.19%	1.84%
E5	MR	0.04	6	198.51	1.35	199.27	0.95	377.04	62.0	378.9	36.48	0.38%	0.48%
E6	MD	0.04	6	217.20	1.37	217.49	0.90	675.08	61.2	698.9	37.18	0.14%	3.40%
E7	MC	0.04	6	192.15	1.17	192.88	0.72	408.87	46.7	416.3	27.17	0.38%	1.79%
E8	MP	0.04	6	198.63	1.32	199.06	1.45	432.70	63.3	440.6	36.23	0.22%	1.80%
E9	MR	0.16	30	236.67	1.45	237.03	1.93	547.02	96.2	557.1	57.48	0.15%	1.81%
E10	MD	0.16	30	370.28	1.48	370.28	1.08	1917.11	99.2	1977.3	58.05	0.00%	3.04%
E11	MC	0.16	30	230.73	1.25	231.75	0.98	788.28	68.2	811.4	38.98	0.44%	2.85%
E12	MP	0.16	30	236.93	1.65	238.02	1.20	831.99	99.3	855.5	59.68	0.46%	2.74%
E13	MR	0.04	30	213.18	1.35	213.69	0.95	448.08	62.9	454.1	36.82	0.24%	1.33%
E14	MD	0.04	30	259.11	1.37	259.20	0.95	1523.76	62.2	1586.1	36.82	0.04%	3.93%
E15	MC	0.04	30	201.89	1.25	202.54	0.82	497.61	45.8	517.1	27.48	0.32%	3.77%
E16	MP	0.04	30	206.82	1.01	207.32	1.03	548.05	62.2	569.7	37.60	0.24%	3.79%

By comparing the results, we find that the percentage error between the methods in an integer grid is below 0.5%. The average error % is 0.26%. Based on the results, we conclude that the expected cost given by the BILSDP method yields a reasonable approximation to the expected cost computation. For mean demand of 5, the expected cost estimated by the BILSDP and the Truncation approximation methods yields similar results without significant differences in the expected cost. This implies that the VFA given by weighted average costs in bilinear interpolation is well approximated by the truncation approximation in the ending FG and WIP state. The truncation approximation happens only at the expected cost-to-go function, similar to the bilinear interpolation approximation. Therefore, it is evident that both the expected cost between Truncation and BILSDP seem promising in the integer grid size of state space.

In the second experiment, we compare the results of BILSDP and Truncation for a larger state space using fractional grid size. To study the impact of grid size and its impact, in this experiment, we use a 0.5 grid on state and 0.1 grid on action to analyze the effect of grid size and VFA in calculating the expected cost for a state. The purpose

is to compare how different VFA methods affect the optimal policy using a fine grid size on state and action space. We observe that the expected cost between the methods differs significantly. The Truncation method yields poor results compared to the BILSDP. Columns 9-12 from Table 4.10 summarize the results for (0.5, 0.1) grid-based state space problem using BILSDP and Truncation. On average, we get close to a 2.38% error gap between the methods. In particular, we witness a huge gap when the demand pattern follows MD (E14 with 3.93%) and MP pattern (E16 with 3.79%), both at  $b = 30$ . This clearly shows that the Truncation approximation underestimates the ending states, which reflects less  $P^*$  yielding high backorder cost.

BILSDP uses the weighted average costs corresponding to 4 neighboring states. This approximation yields a reasonable solution compared to the Truncation-based SDP procedure because of the convexity of the cost function by its linear cost structure. However, Truncation approximation affects the expected cost-to-go by approximating the ending states using only one nearest neighbor (next lowest state). When the cost function increases in one direction, either above or below  $P^*$ , the Truncation method yields a poor approximation of the expected cost-to-go calculation due to the nearest state approximation. This results in a poor estimation of the expected cost-to-go function. The error present in this type of approximation successively accumulates the error percentage and hence yields an approximate expected cost in period 1.

We now compare the computational times for the two VFAs based on the grid size defined in the two experiments listed above. On average, using a 0.5 grid in state and 0.1 grid in action, the BILSDP takes 74.34 minutes, and the truncation approximation takes 44.59 minutes. In contrast, using an integer grid which takes just 1.08 minutes using Truncation and 1.42 minutes using BILSDP. We observe wide ranges of run time between the methods. For example, E12 takes 99.3 minutes using BILSDP, whereas, Truncation takes only 59.68 minutes. Comparing the two methods, BILSDP takes large run times. The reasons for the extra time taken by BILSDP are the following. The expected cost-to-go function uses additional calculation (calling any of three subroutines based on interpolation using  $X$ ,  $W$ , and  $(X, W)$  directions). Then it applies bilinear interpolation (a weighted average of 4 neighboring states). However, truncation approximation uses the previous possible nearest state, which prevents any extra calculation from finding the expected cost-to-go. Despite minor differences, overall, the run times are reasonable in an integer grid. However, we find a huge difference in terms of cost gap % between

the BILSDP and Truncation using a 0.5 and 0.1 grid on state and action space. Despite the fact that Truncation approximation gives a solution in less run time, the BILSDP method seems promising in terms of expected cost gap%. Therefore, we recommend using BILSDP as the principal method for all benchmarking purposes.

#### 4.7.7 Effect of $W$ State Space Reduction

In this experiment, we study the impact of the  $W$  state space truncation and how it affects the expected cost when the shape of the CF becomes flat. The objective of this experiment is two folds: 1) What is the effect of  $W$  state space truncation that depends on the workload required to cover a certain percentage of maximum production capacity? We compare the expected cost between two levels of  $W$  under two levels of grid size in the state (integer and 0.5 grid) and 0.1 grid size in action space. 2) What is the impact of  $W$  state space truncation in computing times for two different levels of  $W_{max}$ ? We present numerical experiments using the following data. We take relevant costs  $C^r = 1$ ,  $C^w = 0.25$ ,  $C^p = 2$ ,  $h = 0.3$ , with Missbauer CF whose parameters are  $K_1 = 10$ , and  $K_2 = 30$ . We use bilinear interpolation and  $P$  as an action-based original SDP algorithm to solve these models. We study the impact of the  $W$  truncation using two mean demands shown in Tables 4.1, and 4.2 with time-varying patterns described in Section 4.7.1. These mean demands represent 50% and 80% utilization with respect to the maximum production capacity of 10. We study the impact of  $W$  state space using 75% and 90% of the maximum production capacity. We obtain  $W_{max}$  as 99 units for 75% and 279 units for 90% of the production capacity of 10 using Missbauer CF with  $K_2=30$ .

Table 4.11 summarizes the expected cost and computational times for mean demand = 5. We compare the expected cost and computing times of integer and 0.5 grid in state space with 0.1 grid in action space. Results indicate that there is no difference between  $W=99$  and 279 units when utilization is 50%. We get an exact match of the expected cost between the two  $W$  state spaces. However, the computational burden is substantial, as expected. We conclude that the  $W$  state space truncation works well with just 75% of the production capacity. This is because, in a low utilization case, the optimal production does not take the high values of output that require high workloads. Therefore, we expect that the expected cost does not change when the utilization is 50% in a flat CF.

Table 4.11: Expected cost and computational times comparison for various experiments using  $P$ -based standard SDP. We use a 36-core CPU and bilinear interpolation as VFA. We show the impact of state space discretization  $\in \{1, 0.5\}$  for 0.1 grid in action space using mean demand = 5 and  $K_2 = 30$  in a Missbauer CF. We compare the results of  $W_{max}$  between 99 and 279 units covering 75% and 90% of maximum production capacity. We denote the expected cost for the grid size in state and action as (state grid, action grid). We calculate the gap % for the expected cost as  $100\%(Cost_{99}-Cost_{279})/Cost_{99}$

Expt #	Demand	SCV	$b$	(1, 0.1)				(0.5, 0.1)				Comparison	
				W=279		W=99		W=279		W=99		(1, 0.1)	(0.5, 0.1)
				cost	minutes	cost	minutes	cost	hours	cost	minutes	gap %	gap%
E1	MR	0.16	6	274.88	32.43	274.88	11.32	274.63	2.34	274.63	47.58	0.00%	0.00%
E2	MD	0.16	6	389.48	32.17	389.48	11.75	388.64	2.28	388.64	47.85	0.00%	0.00%
E3	MC	0.16	6	289.29	23.98	289.30	8.83	288.67	1.66	288.67	33.27	0.00%	0.00%
E4	MP	0.16	6	302.23	35.88	302.23	14.13	301.79	2.64	301.79	51.32	0.00%	0.00%
E5	MR	0.04	6	261.11	24.62	261.11	8.17	260.88	1.55	260.88	32.00	0.00%	0.00%
E6	MD	0.04	6	339.68	21.97	339.68	10.05	338.33	1.54	338.33	32.32	0.00%	0.00%
E7	MC	0.04	6	259.75	17.67	259.75	6.53	259.06	1.22	259.06	24.58	0.00%	0.00%
E8	MP	0.04	6	274.74	23.75	274.74	8.77	274.08	1.68	274.08	33.60	0.00%	0.00%
E9	MR	0.16	30	322.34	31.03	322.34	11.63	321.82	2.32	321.82	47.87	0.00%	0.00%
E10	MD	0.16	30	596.45	31.97	596.07	11.68	594.08	2.33	593.73	46.65	-0.06%	-0.06%
E11	MC	0.16	30	336.37	23.97	336.37	8.65	334.84	1.65	334.84	32.12	0.00%	0.00%
E12	MP	0.16	30	347.97	36.18	347.98	13.77	346.98	2.65	346.98	53.07	0.00%	0.00%
E13	MR	0.04	30	289.77	22.70	289.77	8.00	289.18	1.53	289.18	31.92	0.00%	0.00%
E14	MD	0.04	30	456.95	21.98	456.63	8.13	454.61	1.56	454.40	31.78	-0.07%	-0.05%
E15	MC	0.04	30	277.00	17.07	277.00	6.08	275.98	1.17	275.98	25.42	0.00%	0.00%
E16	MP	0.04	30	293.25	23.28	293.25	8.98	292.12	1.67	292.12	35.48	0.00%	0.00%

Table 4.12 shows the summary of the expected cost and computing times when mean demand is 8 with 80% utilization. Results indicate that there are substantial improvements in the expected cost only when we use 279 units (90% of capacity) than just 99 units in the  $W$  state space. This result validates our hypothesis on  $W$  state space that we need additional workload when the  $P^*$  is after the curvature of the CF (close to the maximum production capacity). Based on the experimental evidence, we illustrate that we cannot truncate the  $W$  state space below 90% of production capacity when utilization is high in a flat CF. Despite some improvements on the expected cost when  $W=279$  units in 80% utilization, we get an exact match of the expected cost for MR demand pattern when  $b=6$ . Refer to the expected cost of E1, and E5 cases, for instance. The reason is that the cost tradeoff between the backorder cost and the sum of release and production cost is substantial when  $b=30$ . Therefore, additional releases bring more  $P^*$ , which in turn reduces the expected cost when  $b=30$ . On the other hand, when  $b=6$ , and in an MR demand, we encounter a non-decreasing mean demand from period 1 to  $T$ . While iterating the backward induction algorithm, we find the maximum demand occurrence

at the end of the horizon fades away from the FG material balance equation. So the expected cost does not get affected by the  $P^*$ . Hence the tradeoff for additional releases is not significant. So in the case of MR demand, the cost between the workload = 279 versus cost in workload = 99 units does not affect, which in turn reduces the difference in the expected cost when we compare the expected cost for (0, 0) in period 1.

Table 4.12: Expected cost and computational times comparison for various experiments using  $P$ -based standard SDP. We use a 36-core CPU and bilinear interpolation as VFA. We show the impact of state space discretization  $\in \{1, 0.5\}$  for 0.1 grid in action space using mean demand = 8 and  $K_2 = 30$  in a Missbauer CF. We compare the results of  $W_{max}$  between 99 and 279 units covering 75% and 90% of maximum production capacity. We denote the expected cost for the grid size in state and action as (state grid, action grid). We calculate the gap % for the expected cost as  $100\%(Cost_{99}-Cost_{279})/Cost_{99}$

Expt #	Demand	SCV	$b$	(1, 0.1)				(0.5, 0.1)				Comparison	
				W=279		W=99		W=279		W=99		(1, 0.1)	(0.5, 0.1)
				Cost	minutes	Cost	minutes	Cost	hours	Cost	hours	gap %	gap%
E1	MR	0.16	6	583.01	32.92	583.03	23.17	582.42	4.92	582.43	1.67	0.00%	0.00%
E2	MD	0.16	6	1080.56	32.48	1109.44	23.73	1080.17	4.86	1108.94	1.65	2.60%	2.59%
E3	MC	0.16	6	749.69	24.18	758.30	16.00	748.95	3.40	757.33	1.15	1.14%	1.11%
E4	MP	0.16	6	768.62	31.92	777.18	22.90	767.99	4.98	776.32	1.74	1.10%	1.07%
E5	MR	0.04	6	547.50	21.70	547.50	14.17	546.94	3.10	546.94	1.02	0.00%	0.00%
E6	MD	0.04	6	1027.47	22.07	1073.97	14.17	1027.04	3.18	1073.68	1.04	4.33%	4.34%
E7	MC	0.04	6	670.25	18.18	687.49	10.62	669.27	2.29	686.03	0.73	2.51%	2.44%
E8	MP	0.04	6	700.59	24.00	715.28	14.07	699.91	2.99	714.18	1.04	2.05%	2.00%
E9	MR	0.16	30	872.02	36.25	939.95	21.62	870.65	4.93	937.35	1.62	7.23%	7.12%
E10	MD	0.16	30	2570.15	37.63	2955.88	22.82	2568.35	4.88	2952.51	1.66	13.05%	13.01%
E11	MC	0.16	30	1290.66	27.95	1510.76	15.67	1285.12	3.40	1505.59	1.04	14.57%	14.64%
E12	MP	0.16	30	1350.64	38.95	1599.21	22.93	1345.73	4.97	1594.85	1.61	15.54%	15.62%
E13	MR	0.04	30	726.80	26.83	766.15	14.80	725.48	3.07	762.75	0.99	5.14%	4.89%
E14	MD	0.04	30	2185.26	22.43	2609.17	14.38	2183.92	3.09	2604.94	1.03	16.25%	16.16%
E15	MC	0.04	30	917.67	13.47	1102.68	10.83	908.07	2.27	1094.73	0.74	16.78%	17.05%
E16	MP	0.04	30	1003.03	24.53	1230.88	14.50	995.29	3.19	1224.81	1.06	18.51%	18.74%

## 4.8 Principal Results

We summarize the key results of the single-stage, single-item, multi-period capacitated stochastic production inventory system subject to congestion using a non-linear clearing function-based replenishment.

1. We solve the multi-period problem using  $P$  and  $R$  as action-based SDP algorithms to find the optimal policy.  $P$  as an action-based SDP algorithm always gives a lower bound than  $R$  as an action. This is due to the nature of the release consumption and production assumptions that  $P$  as an action-based SDP algorithm is a relaxation of  $R$  as an action-based SDP algorithm.

2. For solving large instances, we propose a binary search approach instead of iterating all possible actions as a predefined procedure. The binary search procedure works based on the following. It reduces the computational time by exploiting the convex structure of the objective cost function. It simply uses  $P^* = 0$  and calculates the expected cost when the optimal policy follows a "do nothing" policy. Using a step size of 0.01, and tolerance of 0.001, we get a reasonable run time and good accuracy in terms of the expected cost.
3. Since the CF can yield fractional production, the ending WIP and FG state becomes continuous against a discrete state space definition. Hence we need a value function approximation to calculate the expected cost-to-go function in the Bellman equation. Based on the experimental evidence, we produce an upper bound on the expected cost using bilinear interpolation. The optimal policy produced using BILSDP can be used as a benchmark for other approximate solutions.
4. Of all the demand patterns, mean decline and mean peak patterns require more computation time when the grid size becomes finer, and give expected cost with a 5% gap when the shape of the CF becomes flat. This is due to the number of additional computations due to state space increase based on the maximum demand occurrence in the early periods of the  $T$  period problem. When the shape of the CF becomes flat, an additional workload is necessary to satisfy peak demands in the early periods of the finite horizon. Under such conditions, the solution quality using the binary search method gives a comparable benchmark.
5. Appropriate  $W$  state space truncation is critical in finding the expected cost. Experimental evidence suggests that we cannot truncate the  $W$  state space when utilization is high in a flat CF. Results indicate that the workload required to meet 90% of the maximum production capacity yields promising results in terms of the expected cost despite showing additional computational burden due to the increase in the state space.

## 4.9 Conclusion

In this chapter, we propose two versions of the SDP algorithm to solve the capacitated production inventory system for a single-item, single-stage, multi-period problem. Previous works did not consider a clearing function-based replenishment in a finite horizon

stochastic inventory problem. We propose  $R$  and  $P$  as action-based SDP algorithms to solve the multi-period production inventory problem. We find the  $P$  as an action-based SDP gives a lower bound to the production inventory problem than the  $R$  as the action. While solving the DP algorithm, an action taken in an initial state may lead to an ending state that is not part of the discretized state space  $(X, W)$ . Thus, the future expected cost operator may not find the cost-to-go from the lookup table. We examine two value function approximations: truncation approximation and bilinear interpolation methods. There is always a trade-off between speed and accuracy in the approximation methods. Finally, we incorporate a binary search algorithm in a parallel computing framework. The BS-based SDP algorithm accelerates the DP recursion by exploiting the problem structure and convexity properties of the Bellman equation. Implementing the BS procedure on the PC framework significantly reduces runtime, and using BILSDP with  $P$  as action solves the problem quickly compared with the original SDP algorithm.

# Chapter 5

## Conclusion and Future Directions

### 5.1 Summary

In this dissertation, we studied single-stage, single-item, single, and multi-period capacitated stochastic production inventory problems subject to congestion. We used a concave non-decreasing clearing function as a meta-model to describe the relationship between the expected output and workload in a period. Several approximate methods are available to solve capacitated production inventory problems using a CF. Exact solutions to these complex problems are computationally burdensome to obtain. We need an exact solution to benchmark the approximation methods using chance-constrained programming, stochastic programming, robust optimization, and simulation optimization-based methods. The only solution method to get an exact solution for production inventory models is stochastic dynamic programming which suffers from the curse of dimensionality. Therefore we need an accelerated SDP procedure that gives a benchmark solution to solve a multi-period problem. Single-period problems are building blocks of multi-period problems, and therefore, we first developed a single-stage single-period mathematical program to find the optimal releases and production.

In Chapter 2, we presented a literature review of classical inventory models and their extensions in a broad sense. We then discussed the optimal policies and structures of the optimal policies in single-echelon and multi-echelon systems. We reviewed exact solution methods using stochastic dynamic programming and approximate dynamic programming algorithms and motivated the need for value function approximations. We briefly shared the relevant literature using chance-constrained models, robust optimization, stochastic

programming, and simulation-based optimization models. In Chapter 3, for the single period problem, using the Karush Kuhn Tucker conditions, we gave the optimal conditions and proposed optimal solution characterization using three cases of release and production. We then developed a single-stage multi-period finite horizon solution using a stochastic dynamic program as an exact solution in Chapter 4. We discussed the policy structure in terms of cost and clearing function parameters. We developed a binary search-based SDP algorithm that significantly reduces the run time while implemented in a parallel computing framework. We used bilinear interpolation as a value function approximation technique to calculate the expected cost-to-go function. We discretized state space such that the optimal policy will act as a benchmark that can be helpful to compare the solution quality and optimality gap with other approximation methods. We studied the single period and multiple periods problem using an extensive test bed and discussed the summary of the principal results.

## 5.2 Future Directions

A number of future directions can be identified based on our work in this dissertation. We divide the future extensions for single and multi-period problems described in Chapters 3 and 4, respectively. From Chapter 3, a clear direction is to extend to multi-item, single-stage problems. We can then understand the evolution of production requirements for multiple items when the capacity is limited by a maximum production capacity. A multi-item problem is complicated when dealing with a single production resource sharing its production capacity. When a CF-based replenishment happens, the allocated clearing function model proposed by Asmundsson et al. (2009, 2006) can be helpful to study the optimal solution characterization as the problem is intuitive from a mathematical perspective, given the solution structure for the single-item, single-period solution from Chapter 3.

Another clear direction is to extend this work into a multi-echelon inventory system for a single item and multi-item systems in a multi-period setting. Developing a multi-item, multi-stage DP model is complex due to the curse of dimensionality. Solving such models requires decomposition methods; see Federgruen et al. (2018) for lower bound approximation using Lagrange relaxation, which is quite challenging. With the single item single period and multi-period solution in place, we can decompose the multi-item, multi-stage problems into single-item, single-stage problems and explore possible directions of

finding optimal policies. We can extend future work in the SDP algorithm implementation for multiple items by decomposing the main problem into sub-problems and solving them separately; see Federgruen et al. (2018); Kunnumkal and Topaloglu (2008, 2011) for references on decomposition procedures. Based on experimental experience, computing is faster when we utilize the convexity of the objective function and, in particular, the "Do Nothing" policy. We know the SDP algorithm suffers from the curse of dimensionality, and it is not a feasible approach to solving complex problems on an industrial scale. However, based on our experience using the effective BS algorithm in parallel computing framework, the BS-based SDP algorithm is computationally efficient. It can be helpful for solving large-scale instances in a reasonable run time at better accuracy defined by the fine grid size. Subsequent works can be explored to find approximate heuristic solutions to solve the production inventory problem. Thus, the accelerated SDP algorithm solution is useful for benchmarking against the approximation solutions.

This dissertation lies at the intersection of production planning and stochastic optimization, linking queueing behavior of production resources with stochastic demand. Finding a scalable solution for large-scale problems is difficult due to the time complexity of exact solution methods. One natural extension is to explore some approximate dynamic programming models (Powell 2022) and use-value function approximations instead of using the future expected cost-to-go lookup table to find the total expected cost of the system. The multi-echelon extensions must necessarily begin by considering a simple serial topology in an infinite horizon situation, evaluating the complexity of the MIP models imposed by the increasing number of stages and the need to model interactions between them. More complex network topologies will require additional approximations whose accuracy will need to be evaluated by analytical and experimental means. Comparing the solutions obtained from the guaranteed-service approach is also of interest. We will extend stochastic DP-based approximation algorithms such as Monte Carlo policy iteration, and reinforcement learning methods such as  $Q$ -learning, temporal difference, and  $\epsilon$ -policy methods to find near-optimal solutions. We use approximate dynamic programming methods to solve the DP formulation using various value function approximations by exploiting the structure of the optimal policies.

A wide range of future research is open for extending the current SDP algorithm. This includes variations of binary search implementation (Nishihara and Nishino 1987), incorporating a stochastic ending in-process inventory instead of the current deterministic

version, and simultaneous optimization of fitting a CF and finding the optimal policy using data-driven methods. Furthermore, production rounding plays a crucial role in calculating the expected cost when using  $R$  as an action. Integer rounding on  $R$  as action and  $P$  as action yield approximations based on VFA. Hence, developing improved rounding methods is an essential direction for future work.

With modern solvers and technology development, mathematical programming techniques using linear and integer programming show promise for approximate solutions to solve production inventory problems with queuing influence and stochastic demand. We can develop improved heuristics for multi-item, multi-period problems using mathematical programming models as an approximation. One possible direction is to find approximate solutions using mathematical programming models such as chance-constrained programming to exploit the power of solver technology to obtain near-optimal solutions to complex production inventory problems governed by queuing behavior (Aouam and Uzsoy 2012a, 2015; Bookbinder and Tan 1988; Lin and Uzsoy 2016; Ravindran et al. 2011; Ziarnetzky et al. 2018). Stochastic optimization techniques such as chance-constrained programming, stochastic programming, and robust optimization models have been applied and compared in recent decades. While these models have shown promising performance in computational experiments, the difficulty of obtaining exact solutions has prevented rigorous assessment of the quality of their solutions relative to optimality. Since we have a benchmark solution for the single-period and multi-period problems, we can compare the approximate solutions with the SPP and SDP solutions to assess the quality of solutions relative to optimality.

Another direction is to explore the rolling horizon planning (RP) using a DP-based heuristic with the latest demand distribution updates instead of known demand distributions in advance. A similar approach is suggested in the literature for lot-sizing problems; see Lee and Denardo (1986); Van Den Heuvel and Wagelmans (2005). This approach may yield promising results since it involves the latest demand distribution (new data) and exploits the DP algorithm's shorter periods instead of solving  $T$  periods. Decomposing the horizon into epochs and solving for shorter instances of the problem minimizes the overall runtime of solving a large-scale problem that suffers from the curse of dimensionality. We will then compare the solution quality of the RP-based DP solution in terms of runtime with other approximation solutions, such as CC-RP models, and evaluate the performance in terms of solution quality and practical implementation feasibility

determined by adequate run time.

An alternate extension of the RP-based DP solution is with the high-quality, robust optimization (RO) approximate models. The RO models proposed by Bertsimas and Sim (2004) and discussed a single-stage capacitated production-inventory system application in Bertsimas and Thiele (2006b). Using robust optimization models, dealing with uncertainty sets ("rectangularity" property), and solving inventory problems have been potential research areas. Extending the robust optimization models using DP-based approaches is another direction of extension; see Georghiou et al. (2019); Iyengar (2005) for details on robust DP approaches.

## BIBLIOGRAPHY

- Abdel-Malek, L., Shan, P., and Montanari, R. (2020). A constructive methodology to solving the capacitated newsvendor problem: an approximate approach. In *SN Operations Research Forum*, volume 1, pages 1–16. Springer.
- Albey, E. and Uzsoy, R. (2015). Lead time modeling in production planning. In *2015 Winter Simulation Conference (WSC)*, pages 1996–2007. IEEE.
- Albey, E., Uzsoy, R., and Kempf, K. G. (2016). A chance constraint based multi-item production planning model using simulation optimization. In *2016 Winter Simulation Conference (WSC)*, pages 2719–2730. IEEE.
- Albey, E., Yanıkoğlu, İ., and Uzsoy, R. (2019). A robust optimization approach for production planning under exogenous planned lead times. In *2019 Winter Simulation Conference (WSC)*, pages 2312–2323. IEEE.
- Alfaro, J. A. and Corbett, C. J. (2003). The value of sku rationalization in practice (the pooling effect under suboptimal inventory policies and nonnormal demand). *Production and Operations Management*, 12(1):12–29.
- Andriolo, A., Battini, D., Grubbström, R. W., Persona, A., and Sgarbossa, F. (2014). A century of evolution from harris basic lot size model: Survey and research agenda. *International Journal of Production Economics*, 155:16–38. Celebrating a century of the economic order quantity model.
- Aouam, T. and Uzsoy, R. (2012a). Chance-constraint-based heuristics for production planning in the face of stochastic demand and workload-dependent lead times. In *Decision Policies for Production Networks*, pages 173–208. Springer.
- Aouam, T. and Uzsoy, R. (2012b). An exploratory analysis of production planning in the face of stochastic demand and workload-dependent lead times. decision policies for production networks. kg kempf and d. armbruster.
- Aouam, T. and Uzsoy, R. (2015). Zero-order production planning models with stochastic demand and workload-dependent lead times. *International Journal of Production Research*, 53(6):1661–1679.
- Arora, S. and Barak, B. (2009). *Computational complexity: a modern approach*. Cambridge University Press.
- Arrow, K. J., Harris, T., and Marschak, J. (1951). Optimal inventory policy. *Econometrica: Journal of the Econometric Society*, pages 250–272.

- Asmundsson, J., Rardin, R. L., Turkseven, C. H., and Uzsoy, R. (2009). Production planning with resources subject to congestion. *Naval Research Logistics (NRL)*, 56(2):142–157.
- Asmundsson, J., Rardin, R. L., and Uzsoy, R. (2006). Tractable nonlinear production planning models for semiconductor wafer fabrication facilities. *IEEE Transactions on Semiconductor Manufacturing*, 19(1):95–111.
- Aviv, Y. and Federgruen, A. (1997). Stochastic inventory models with limited production capacity and periodically varying parameters. *Probability in the Engineering and Informational Sciences*, 11(1):107–135.
- Axsäter, S. (2015). *Inventory control*, volume 225. Springer.
- Bang, J.-Y. and Kim, Y.-D. (2009). Hierarchical production planning for semiconductor wafer fabrication based on linear programming and discrete-event simulation. *IEEE Transactions on Automation Science and Engineering*, 7(2):326–336.
- Barbiero, A. and Hitaj, A. (2022). Discrete approximations of continuous probability distributions obtained by minimizing cramér-von mises-type distances. *Statistical Papers*, pages 1–29.
- Bazaraa, M. S., Sherali, H. D., and Shetty, C. M. (2013). *Nonlinear programming: theory and algorithms*. John Wiley & Sons.
- Bellman, R. (1957). *Dynamic programming*, Princeton University Press, New Jersey. Dover paperback edition.
- Ben-Tal, A. and Nemirovski, A. (2002). Robust optimization—methodology and applications. *Mathematical programming*, 92(3):453–480.
- Bertsekas, D. (2012). *Dynamic programming and optimal control: Volume I*, volume 1. Athena scientific.
- Bertsekas, D., Nedic, A., and Ozdaglar, A. (2003). *Convex analysis and optimization*, volume 1. Athena Scientific.
- Bertsekas, D. P. (1995). *Dynamic programming and optimal control*, volume 1. Athena scientific Belmont, MA.
- Bertsekas, D. P. et al. (2011). Dynamic programming and optimal control 3rd edition, volume ii. *Belmont, MA: Athena Scientific*.
- Bertsekas, D. P. and Tsitsiklis, J. N. (1996). *Neuro-dynamic programming*. Athena Scientific.
- Bertsimas, D., Brown, D. B., and Caramanis, C. (2011). Theory and applications of robust optimization. *SIAM review*, 53(3):464–501.

- Bertsimas, D., Gupta, V., and Kallus, N. (2018). Data-driven robust optimization. *Mathematical Programming*, 167:235–292.
- Bertsimas, D. and Sim, M. (2004). The price of robustness. *Operations research*, 52(1):35–53.
- Bertsimas, D. and Thiele, A. (2006a). Robust and data-driven optimization: modern decision making under uncertainty. In *Models, methods, and applications for innovative decision making*, pages 95–122. INFORMS.
- Bertsimas, D. and Thiele, A. (2006b). A robust optimization approach to inventory theory. *Operations research*, 54(1):150–168.
- Birge, J. R. (1997). State-of-the-art-survey—stochastic programming: Computation and applications. *INFORMS journal on computing*, 9(2):111–133.
- Birge, J. R. and Louveaux, F. (2011). *Introduction to stochastic programming*. Springer Science & Business Media.
- Blau, R. A. (1974). Stochastic programming and decision analysis: an apparent dilemma. *Management Science*, 21(3):271–276.
- Bookbinder, J. H. and Tan, J.-Y. (1988). Strategies for the probabilistic lot-sizing problem with service-level constraints. *Management Science*, 34(9):1096–1108.
- Boute, R. N., Gijbrecchts, J., van Jaarsveld, W., and Vanvuchelen, N. (2021). Deep reinforcement learning for inventory control: a roadmap. *European Journal of Operational Research*.
- Boyd, S., Boyd, S. P., and Vandenberghe, L. (2004). *Convex optimization*. Cambridge university press.
- Brown, D. T. (1959). A note on approximations to discrete probability distributions. *Information and control*, 2(4):386–392.
- Brunskill, E. (2019). Introduction to reinforcement learning. Online lecture.
- Burden, R. and Faires, J. (1985). Numerical analysis, 2.1 the bisection algorithm.
- Buzacott, J. A. and Shanthikumar, J. G. (1992). Design of manufacturing systems using queueing models. *Queueing systems*, 12(1):135–213.
- Buzacott, J. A. and Shanthikumar, J. G. (1993). *Stochastic models of manufacturing systems*, volume 4. Prentice Hall Englewood Cliffs, NJ.
- Cai, Y. (2009). *Dynamic Programming and its application in Economics and Finance*. PhD thesis, Stanford University.

- Cai, Y. and Judd, K. L. (2013). Shape-preserving dynamic programming. *Mathematical Methods of Operations Research*, 77(3):407–421.
- Cai, Y. and Judd, K. L. (2015). Dynamic programming with hermite approximation. *Mathematical Methods of Operations Research*, 81(3):245–267.
- Cárdenas-Barrón, L. E., Chung, K.-J., and Treviño-Garza, G. (2014). Celebrating a century of the economic order quantity model in honor of ford whitman harris.
- Carey, M. (1987). Optimal time-varying flows on congested networks. *Operations research*, 35(1):58–69.
- Carey, M. (1992). Nonconvexity of the dynamic traffic assignment problem. *Transportation Research Part B: Methodological*, 26(2):127–133.
- Carey, M. and Bowers, M. (2012). A review of properties of flow–density functions. *Transport Reviews*, 32(1):49–73.
- Charnes, A. and Cooper, W. W. (1959). Chance-constrained programming. *Management science*, 6(1):73–79.
- Chen, F. and Zheng, Y.-S. (1994). Lower bounds for multi-echelon stochastic inventory systems. *Management Science*, 40(11):1426–1443.
- Cheng, F. and Sethi, S. P. (1999). Optimality of state-dependent (s, s) policies in inventory models with markov-modulated demand and lost sales. *Production and operations management*, 8(2):183–192.
- Choi, T.-M. (2012). *Handbook of Newsvendor problems: Models, extensions and applications*, volume 176. Springer.
- Ciarallo, F. W., Akella, R., and Morton, T. E. (1994). A periodic review, production planning model with uncertain capacity and uncertain demand—optimality of extended myopic policies. *Management science*, 40(3):320–332.
- Clark, A. J. and Scarf, H. (1960). Optimal policies for a multi-echelon inventory problem. *Management science*, 6(4):475–490.
- Clausen, J. B. B. and Li, H. (2022). Big data driven order-up-to level model: Application of machine learning. *Computers & Operations Research*, 139:105641.
- Dantzig, G. B. (1955). Linear programming under uncertainty. *Management science*, 1(3-4):197–206.
- Das, T. K., Gosavi, A., Mahadevan, S., and Marchallick, N. (1999). Solving semi-markov decision problems using average reward reinforcement learning. *Management Science*, 45(4):560–574.

- De Kok, A. and Seidel, H. (1990). Analysis of stock allocation in a two echelon distribution system. In *CQM-Note 98*. Centre for Quantitative Methods, Philips Electronics Eindhoven.
- De Kok, A. d. and Graves, S. C. (2003). *Supply chain management: Design, coordination and operation*. Elsevier.
- De Kok, T. (2018). Inventory management: Modeling real-life supply chains and empirical validity. *Foundations and Trends® in Technology, Information and Operations Management*, 11(4):343–437.
- De Kok, T., Grob, C., Laumanns, M., Minner, S., Rambau, J., and Schade, K. (2018). A typology and literature review on stochastic multi-echelon inventory models. *European Journal of Operational Research*, 269(3):955–983.
- De Kok, T. G. and Fransoo, J. C. (2003). Planning supply chain operations: definition and comparison of planning concepts. *Handbooks in operations research and management science*, 11:597–675.
- De Kok, T. G. and Visschers, J. W. (1999). Analysis of assembly systems with service level constraints. *International Journal of Production Economics*, 59(1-3):313–326.
- Diks, E. and De Kok, A. (1999). Computational results for the control of a divergent n-echelon inventory system. *International Journal of Production Economics*, 59(1-3):327–336.
- Diks, E., De Kok, A., and Lagodimos, A. (1996). Multi-echelon systems: A service measure perspective. *European Journal of Operational Research*, 95(2):241–263.
- Diks, E. B. and De Kok, A. (1998). Optimal control of a divergent multi-echelon inventory system. *European journal of operational research*, 111(1):75–97.
- Dimitri, P. et al. (1999). *Nonlinear programming*. Athena Scientific.
- Edgeworth, F. Y. (1888). The mathematical theory of banking. *Journal of the Royal Statistical Society*, 51(1):113–127.
- Eppen, G. and Schrage, L. (1981). Centralized ordering policies in multi-echelon systems with lead times and random demand. lb schwarz, ed. *Multi-Level Production/Inventory Control Systems: Theory and Practice*.
- Eppen, G. D. (1979). Note—effects of centralization on expected costs in a multi-location newsboy problem. *Management science*, 25(5):498–501.
- Erlenkotter, D. (2014). Ford whitman harris economical lot size model. *International Journal of Production Economics*, 155:12–15. Celebrating a century of the economic order quantity model.

- Ermol'ev, Y. M. and Mirzoakhmedov, F. (1976). Direct methods of stochastic programming in inventory control problems. *Cybernetics*, 12(6):887–895.
- Federgruen, A., Guetta, C. D., and Iyengar, G. (2018). Two-echelon distribution systems with random demands and storage constraints. *Naval Research Logistics (NRL)*, 65(8):594–618.
- Federgruen, A. and Zipkin, P. (1984). Approximations of dynamic, multilocation production and inventory problems. *Management Science*, 30(1):69–84.
- Federgruen, A. and Zipkin, P. (1986a). An inventory model with limited production capacity and uncertain demands i. the average-cost criterion. *Mathematics of Operations Research*, 11(2):193–207.
- Federgruen, A. and Zipkin, P. (1986b). An inventory model with limited production capacity and uncertain demands ii. the discounted-cost criterion. *Mathematics of Operations Research*, 11(2):208–215.
- Fu, M. C. (1994). Optimization via simulation: A review. *Annals of operations research*, 53(1):199–247.
- Fu, M. C. (2002). Optimization for simulation: Theory vs. practice. *INFORMS Journal on Computing*, 14(3):192–215.
- Fu, M. C. et al. (2015). *Handbook of simulation optimization*, volume 216. Springer.
- Gallego, G., Özer, Ö., and Zipkin, P. (2007). Bounds, heuristics, and approximations for distribution systems. *Operations Research*, 55(3):503–517.
- Garey, M. R. and Johnson, D. S. (1979). *Computers and intractability*. W.H.Freeman company.
- Georghiou, A., Tsoukalas, A., and Wiesemann, W. (2019). Robust dual dynamic programming. *Operations Research*, 67(3):813–830.
- Giannoccaro, I. and Pontrandolfo, P. (2002). Inventory management in supply chains: a reinforcement learning approach. *International Journal of Production Economics*, 78(2):153–161.
- Glasserman, P. (1997). Bounds and asymptotics for planning critical safety stocks. *Operations Research*, 45(2):244–257.
- Glasserman, P. and Tayur, S. (1994). The stability of a capacitated, multi-echelon production-inventory system under a base-stock policy. *Operations Research*, 42(5):913–925.
- Glasserman, P. and Tayur, S. (1995). Sensitivity analysis for base-stock levels in multi-echelon production-inventory systems. *Management Science*, 41(2):263–281.

- Glasserman, P. and Tayur, S. (1996). A simple approximation for a multistage capacitated production-inventory system. *Naval Research Logistics (NRL)*, 43(1):41–58.
- Gopalswamy, K. (2019). *Production Planning with Clearing Functions: Data-driven Approaches and Conic Programming*. PhD thesis, NC State University.
- Gopalswamy, K., Fathi, Y., and Uzsoy, R. (2019). Valid inequalities for concave piecewise linear regression. *Operations Research Letters*, 47(1):52–58.
- Gorissen, B. L., Yanikoğlu, İ., and den Hertog, D. (2015). A practical guide to robust optimization. *Omega*, 53:124–137.
- Graves, S. C. (1985). A multi-echelon inventory model for a repairable item with one-for-one replenishment. *Management science*, 31(10):1247–1256.
- Graves, S. C. (1986). A tactical planning model for a job shop. *Operations Research*, 34(4):522–533.
- Graves, S. C. (1988). Safety stocks in manufacturing systems. *Journal of Manufacturing and Operations Management*, 1:67–101.
- Graves, S. C. and Lesnaia, E. (2004). Optimizing safety stock placement in general network supply chains. In *Proceedings of Singapore-MIT Alliance Annual Symposium Conference*, pages 7–13.
- Graves, S. C. and Schoenmeyr, T. (2016). Strategic safety-stock placement in supply chains with capacity constraints. *Manufacturing & Service Operations Management*, 18(3):445–460.
- Graves, S. C. and Willems, S. P. (2000). Optimizing strategic safety stock placement in supply chains. *Manufacturing & Service Operations Management*, 2(1):68–83.
- Graves, S. C. and Willems, S. P. (2003a). Erratum: optimizing strategic safety stock placement in supply chains.(correspondence). *Manufacturing & Service Operations Management*, 5(2):176–178.
- Graves, S. C. and Willems, S. P. (2003b). Supply chain design: safety stock placement and supply chain configuration. *Handbooks in operations research and management science*, 11:95–132.
- Hackman, S. T. and Leachman, R. C. (1989). A general framework for modeling production. *Management Science*, 35(4):478–495.
- Hadley, G. (1965). *Nonlinear and dynamic programming*. Addison Wesley publishing company inc.
- Hadley, G. and Whitin, T. M. (1963). *Analysis of inventory systems*. Prentice-Hall.

- Harris, F. (1915). Operations and cost, aw shaw co. *Factory Management Series, Chicago*.
- Harris, F. W. (1913). How many parts to make at once.
- Heath, D. C. and Jackson, P. L. (1994). Modeling the evolution of demand forecasts ith application to safety stock analysis in production/distribution systems. *IIE transactions*, 26(3):17–30.
- Higle, J. L. and Kempf, K. G. (2010). Production planning under supply and demand uncertainty: A stochastic programming approach. In *Stochastic Programming*, pages 297–315. Springer.
- Holt, M. and Modigliani, J. (1960). Muth, and simon, planning production. inventories, and work force. *Englewood Cliffs, NJ*.
- Hopp, W. J. and Spearman, M. L. (2011). *Factory physics*. Waveland Press.
- Houda, M. (2007). Convexity and dependence in chance-constrained programming. Technical report, Research Report 2190, Institute of Information Theory and Automation.
- Howard, R. A. (1960). *Dynamic programming and markov processes*. John Wiley.
- Huh, W. T. and Janakiraman, G. (2010). Base-stock policies in capacitated assembly systems: Convexity properties. *Naval Research Logistics (NRL)*, 57(2):109–118.
- Humair, S. and Willems, S. P. (2011). Optimizing strategic safety stock placement in general acyclic networks. *Operations Research*, 59(3):781–787.
- Hung, Y.-F. and Leachman, R. C. (1996). A production planning methodology for semiconductor manufacturing based on iterative simulation and linear programming calculations. *IEEE Transactions on Semiconductor manufacturing*, 9(2):257–269.
- Iglehart, D. L. (1963). Optimality of (s, s) policies in the infinite horizon dynamic inventory problem. *Management science*, 9(2):259–267.
- Inderfurth, K. (1991). Safety stock optimization in multi-stage inventory systems. *International Journal of Production Economics*, 24(1-2):103–113.
- Irdem, D. F., Kacar, N. B., and Uzsoy, R. (2008). An experimental study of an iterative simulation-optimization algorithm for production planning. In *2008 Winter Simulation Conference*, pages 2176–2184. IEEE.
- Irdem, D. F., Kacar, N. B., and Uzsoy, R. (2010). An exploratory analysis of two iterative linear programming—simulation approaches for production planning. *IEEE Transactions on Semiconductor Manufacturing*, 23(3):442–455.
- Iyengar, G. N. (2005). Robust dynamic programming. *Mathematics of Operations Research*, 30(2):257–280.

- Johnson, E. L. (1968). On  $(s, s)$  policies. *Management Science*, 15(1):80–101.
- Johnson, L. A., Montgomery, D. C., and Montgomery, D. C. (1974). *Operations research in production planning, scheduling, and inventory control*. John Wiley & Sons Incorporated.
- Judd, K. L. and Judd, K. L. (1998). *Numerical methods in economics*. MIT press.
- Kacar, N. B. (2012). *Fitting Clearing Functions to Empirical Data: Simulation Optimization and Heuristic Algorithms*. PhD thesis, North Carolina State University.
- Kacar, N. B., Irtem, D. F., and Uzsoy, R. (2011). An experimental comparison of production planning using clearing functions and iterative linear programming-simulation algorithms. *IEEE Transactions on Semiconductor Manufacturing*, 25(1):104–117.
- Kacar, N. B., Mönch, L., and Uzsoy, R. (2013). Planning wafer starts using nonlinear clearing functions: A large-scale experiment. *IEEE Transactions on Semiconductor Manufacturing*, 26(4):602–612.
- Kacar, N. B., Mönch, L., and Uzsoy, R. (2016). Modeling cycle times in production planning models for wafer fabrication. *IEEE Transactions on Semiconductor Manufacturing*, 29(2):153–167.
- Kapuściński, R. and Tayur, S. (1998). A capacitated production-inventory model with periodic demand. *Operations Research*, 46(6):899–911.
- Karlin, S. (1960). Dynamic inventory policy with varying stochastic demands. *Management Science*, 6(3):231–258.
- Karmarkar, U. S. (1989). Capacity loading and release planning with work-in-progress (wip) and leadtimes. *Journal of Manufacturing and Operations Management*, 2(105-123).
- Keerthi, S. S. and Ravindran, B. (1994). A tutorial survey of reinforcement learning. *Sadhana*, 19(6):851–889.
- Kemmer, L., von Kleist, H., de Rochebouët, D., Tziortziotis, N., and Read, J. (2018). Reinforcement learning for supply chain optimization. In *European Workshop on Reinforcement Learning*, volume 14.
- Khouja, M. (1999). The single-period (news-vendor) problem: literature review and suggestions for future research. *omega*, 27(5):537–553.
- Kim, B. and Kim, S. (2001). Extended model for a hybrid production planning approach. *International Journal of Production Economics*, 73(2):165–173.

- Kingman, J. (1961). The single server queue in heavy traffic. In *Mathematical Proceedings of the Cambridge Philosophical Society*, volume 57, pages 902–904. Cambridge University Press.
- Korom, M. (2017). Inventory optimization with guaranteed service time. Technical report, Eötvös Lorand University of Science, Department of Operational Research.
- Kunnumkal, S. and Topaloglu, H. (2008). A duality-based relaxation and decomposition approach for inventory distribution systems. *Naval Research Logistics (NRL)*, 55(7):612–631.
- Kunnumkal, S. and Topaloglu, H. (2011). Linear programming based decomposition methods for inventory distribution systems. *European Journal of Operational Research*, 211(2):282–297.
- 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.
- Lau, H.-S. and Lau, A. H.-L. (1997). Some results on implementing a multi-item multi-constraint single-period inventory model. *International Journal of Production Economics*, 48(2):121–128.
- Lee, C.-Y. and Denardo, E. V. (1986). Rolling planning horizons: Error bounds for the dynamic lot size model. *Mathematics of Operations Research*, 11(3):423–432.
- Lejeune, M. A. and Prékopa, A. (2018). Relaxations for probabilistically constrained stochastic programming problems: review and extensions. *Annals of Operations Research*, pages 1–22.
- Lesnaia, E. (2003). Optimizing strategic safety stock placement in two-layer supply chains. Technical report, Massachusetts Institute of Technology.
- Lesnaia, E. (2004). *Optimizing safety stock placement in general network supply chains*. PhD thesis, Massachusetts Institute of Technology.
- Lesnaia, E., Vasilescu, I., and Graves, S. C. (2005). The complexity of safety stock placement in general-network supply chains. Technical report, Massachusetts Institute of Technology.
- Li, M., Yang, F., Uzsoy, R., and Xu, J. (2016). A metamodel-based monte carlo simulation approach for responsive production planning of manufacturing systems. *Journal of Manufacturing Systems*, 38:114–133.
- Li, Z., Ding, R., and Floudas, C. A. (2011). A comparative theoretical and computational study on robust counterpart optimization: I. robust linear optimization and robust mixed integer linear optimization. *Industrial & engineering chemistry research*, 50(18):10567–10603.

- Li, Z. and Floudas, C. A. (2014). A comparative theoretical and computational study on robust counterpart optimization: Iii. improving the quality of robust solutions. *Industrial & engineering chemistry research*, 53(33):13112–13124.
- Lin, A. (2019). Binary search algorithm. *WikiJournal of Science*, 2(1):1–13.
- Lin, P.-C. and Uzsoy, R. (2016). Chance-constrained formulations in rolling horizon production planning: an experimental study. *International Journal of Production Research*, 54(13):3927–3942.
- Little, J. D. (1961). A proof for the queuing formula:  $L = \lambda w$ . *Operations research*, 9(3):383–387.
- Liu, J., Li, C., Yang, F., Wan, H., and Uzsoy, R. (2011). Production planning for semiconductor manufacturing via simulation optimization. In *Proceedings of the 2011 winter simulation conference (WSC)*, pages 3612–3622. IEEE.
- Luedtke, J., Ahmed, S., and Nemhauser, G. L. (2010). An integer programming approach for linear programs with probabilistic constraints. *Mathematical programming*, 122(2):247–272.
- Magnanti, T. L., Shen, Z.-J. M., Shu, J., Simchi-Levi, D., and Teo, C.-P. (2006). Inventory placement in acyclic supply chain networks. *Operations Research Letters*, 34(2):228–238.
- MathWorks, I. (2005). *MATLAB: The Language of Technical Computing. Getting started with MATLAB, version 7*, volume 1. MathWorks, Incorporated.
- MathWorks, I. (2020). *Parallel Computing Toolbox: User’s Guide, R2020*. MathWorks, Incorporated.
- Medhi, J. (2002). *Stochastic models in queueing theory*. Elsevier.
- Merchant, D. K. and Nemhauser, G. L. (1978). A model and an algorithm for the dynamic traffic assignment problems. *Transportation science*, 12(3):183–199.
- Miller III, A. C. and Rice, T. R. (1983). Discrete approximations of probability distributions. *Management science*, 29(3):352–362.
- Missbauer, H. (2002). Aggregate order release planning for time-varying demand. *International Journal of Production Research*, 40(3):699–718.
- Missbauer, H. (2011). Order release planning with clearing functions: a queueing-theoretical analysis of the clearing function concept. *International Journal of Production Economics*, 131(1):399–406.

- Missbauer, H. (2020). Order release planning by iterative simulation and linear programming: Theoretical foundation and analysis of its shortcomings. *European Journal of Operational Research*, 280(2):495–507.
- Missbauer, H. and Uzsoy, R. (2011). Optimization models of production planning problems. In *Planning production and inventories in the extended enterprise*, pages 437–507. Springer.
- Missbauer, H. and Uzsoy, R. (2020a). Applications of clearing functions. In *Production Planning with Capacitated Resources and Congestion*, pages 239–262. Springer.
- Missbauer, H. and Uzsoy, R. (2020b). *Production Planning with Capacitated Resources and Congestion*. Springer.
- Missbauer, H. and Uzsoy, R. (2021). Order release in production planning and control systems: challenges and opportunities. *International Journal of Production Research*, pages 1–21.
- Modigliani, F. and Hohn, F. E. (1955). Production planning over time and the nature of the expectation and planning horizon. *Econometrica, Journal of the Econometric Society*, pages 46–66.
- Morse, P. M., Kimball, G. E., and Gass, S. I. (1951). *Methods of operations research*. Courier Corporation.
- Mula, J., Poler, R., García-Sabater, J. P., and Lario, F. C. (2006). Models for production planning under uncertainty: A review. *International journal of production economics*, 103(1):271–285.
- Nishihara, S. and Nishino, H. (1987). Binary search revisited: Another advantage of fibonacci search. *IEEE transactions on computers*, 100(9):1132–1135.
- Norouzi, A. (2013). *The effect of Forecast Evolution on Production Planning with Resources Subject to Congestion*. PhD thesis, NC State University.
- Oroojlooyjadid, A., Snyder, L. V., and Takáč, M. (2020). Applying deep learning to the newsvendor problem. *IIE Transactions*, 52(4):444–463.
- Panzer, M. and Bender, B. (2021). Deep reinforcement learning in production systems: a systematic literature review. *International Journal of Production Research*, pages 1–26.
- Pochet, Y. and Wolsey, L. A. (2006). *Production planning by mixed integer programming*. Springer Science & Business Media.
- Powell, W. B. (2007). *Approximate Dynamic Programming: Solving the curses of dimensionality*, volume 703. John Wiley & Sons.

- Powell, W. B. (2014). Clearing the jungle of stochastic optimization. In *Bridging data and decisions*, pages 109–137. Informs.
- Powell, W. B. (2021). From reinforcement learning to optimal control: A unified framework for sequential decisions. In *Handbook of Reinforcement Learning and Control*, pages 29–74. Springer.
- Powell, W. B. (2022). *Reinforcement Learning and Stochastic Optimization: A unified framework for sequential decisions*. John Wiley & Sons.
- Powell, W. B. et al. (2022). *Sequential Decision Analytics and Modeling: Modeling with Python*. Now Publishers.
- Prekopa, A. (1973). Contributions to the theory of stochastic programming. *Mathematical Programming*, 4(1):202–221.
- Prékopa, A. (2013). *Stochastic programming*, volume 324. Springer Science & Business Media.
- Prékopa, A., Vizvari, B., and Badics, T. (1998). *Programming under probabilistic constraint with discrete random variable*. Springer.
- Prékopa, A., Yoda, K., and Subasi, M. M. (2011). Uniform quasi-concavity in probabilistic constrained stochastic programming. *Operations Research Letters*, 39(3):188–192.
- Puterman, M. L. (2014). *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons.
- Qin, Y., Wang, R., Vakharia, A. J., Chen, Y., and Seref, M. M. (2011). The newsvendor problem: Review and directions for future research. *European Journal of Operational Research*, 213(2):361–374.
- Qiu, R., Sun, M., and Lim, Y. F. (2017). Optimizing (s, s) policies for multi-period inventory models with demand distribution uncertainty: Robust dynamic programming approaches. *European Journal of Operational Research*, 261(3):880–892.
- Ravindran, A., Kempf, K. G., and Uzsoy, R. (2011). Production planning with load dependent lead times and safety stocks for a single product. *International Journal of Planning and Scheduling*, 1(1-2):58–89.
- Ravindran, B. (2017). Lecture notes in reinforcement learning. Online lecture.
- Rong, Y., Atan, Z., and Snyder, L. V. (2017). Heuristics for base-stock levels in multi-echelon distribution networks. *Production and Operations Management*, 26(9):1760–1777.
- Rosling, K. (1989). Optimal inventory policies for assembly systems under random demands. *Operations Research*, 37(4):565–579.

- Ross, S. M. (2014). *Introduction to probability models*. Academic press.
- Scarf, H. (April, 1959). The optimality of (s,s) policies in the dynamic inventory problem. Technical report, Applied Mathematics and statistics laboratory, Stanford University.
- Scarf, H. E. (2002). Inventory theory. *Operations Research*, 50(1):186–191.
- Schäl, M. (1976). On the optimality of (s,s)-policies in dynamic inventory models with finite horizon. *SIAM Journal on Applied Mathematics*, 30(3):528–537.
- See, C.-T. and Sim, M. (2010). Robust approximation to multiperiod inventory management. *Operations research*, 58(3):583–594.
- Shang, K. H. and Song, J.-S. (2003). Newsvendor bounds and heuristic for optimal policies in serial supply chains. *Management Science*, 49(5):618–638.
- Shapiro, A. (2006). On complexity of multistage stochastic programs. *Operations Research Letters*, 34(1):1–8.
- Shapiro, A., Dentcheva, D., and Ruszczyński, A. (2021). *Lectures on stochastic programming: modeling and theory*. SIAM.
- Sharma, G. and Martin, J. (2009). Matlab®: a language for parallel computing. *International Journal of Parallel Programming*, 37(1):3–36.
- Sikorski, K. (1982). Bisection is optimal. *Numerische Mathematik*, 40(1):111–117.
- Sikorski, K. and Trojan, G. (1990). Asymptotic near optimality of the bisection method. *Numerische Mathematik*, 57(1):421–433.
- Silver, D. (2015). Introduction to reinforcement learning. Online lecture.
- Simpson Jr, K. F. (1958). In-process inventories. *Operations Research*, 6(6):863–873.
- Snyder, L. V. and Shen, Z.-J. M. (2019). *Fundamentals of supply chain theory*. Wiley Online Library.
- Spitter, J., De Kok, A., and Dellaert, N. (2003). *Cost implications of planned lead times in supply chain operations planning*. Beta, Research School for Operations Management and Logistics.
- Spitter, J., Hurkens, C. A., De Kok, A., Lenstra, J. K., and Negenman, E. G. (2005). Linear programming models with planned lead times for supply chain operations planning. *European Journal of operational research*, 163(3):706–720.
- Srinivasan, A., Carey, M., Morton, T. E., et al. (1988). Resource pricing and aggregate scheduling in manufacturing systems. Technical report, Carnegie Mellon University, Tepper School of Business.

- Sui, Z., Gosavi, A., and Lin, L. (2010). A reinforcement learning approach for inventory replenishment in vendor-managed inventory systems with consignment inventory. *Engineering Management Journal*, 22(4):44–53.
- Sutton, R. S. and Barto, A. G. (2018). *Reinforcement learning: An introduction*. MIT press.
- Tarim, S. A. and Kingsman, B. G. (2004). The stochastic dynamic production/inventory lot-sizing problem with service-level constraints. *International Journal of Production Economics*, 88(1):105–119.
- Tayur, S. R. (1993). Computing the optimal policy for capacitated inventory models. *Stochastic Models*, 9(4):585–598.
- Van Den Heuvel, W. and Wagelmans, A. P. (2005). A comparison of methods for lot-sizing in a rolling horizon environment. *Operations Research Letters*, 33(5):486–496.
- Van Houtum, G., Inderfurth, K., and Zijm, W. H. (1996). Materials coordination in stochastic multi-echelon systems. *European Journal of Operational Research*, 95(1):1–23.
- Van Houtum, G.-J. and Zijm, W. H. M. (1991). Computational procedures for stochastic multi-echelon production systems. *International Journal of Production Economics*, 23(1-3):223–237.
- Veinott, Jr, A. F. (1966). On the optimality of (s,s) inventory policies: New conditions and a new proof. *SIAM Journal on Applied Mathematics*, 14(5):1067–1083.
- Veinott Jr, A. F. (1965). Optimal policy for a multi-product, dynamic, nonstationary inventory problem. *Management Science*, 12(3):206–222.
- Veinott Jr, A. F. and Wagner, H. M. (1965). Computing optimal (s, s) inventory policies. *Management Science*, 11(5):525–552.
- Verderame, P. M., Elia, J. A., Li, J., and Floudas, C. A. (2010). Planning and scheduling under uncertainty: a review across multiple sectors. *Industrial & engineering chemistry research*, 49(9):3993–4017.
- Vollmann, T. E., Berry, W. L., and Whybark, D. C. (1997). *Manufacturing planning and control systems*. Irwin/McGraw-Hill.
- Voß, S. and Woodruff, D. L. (2006). *Introduction to computational optimization models for production planning in a supply chain*, volume 240. Springer Science & Business Media.
- Waldmann, K.-H. (1984). Inventory control in randomly varying environments. *SIAM Journal on Applied Mathematics*, 44(3):657–666.

- Werbos, P. J. (1987). Building and understanding adaptive systems: A statistical/numerical approach to factory automation and brain research. *IEEE Transactions on Systems, Man, and Cybernetics*, 17(1):7–20.
- Wets, R. J. (1998). Stochastic programs with chance constraints: Generalized convexity and approximation issues. In *Generalized Convexity, Generalized Monotonicity: Recent Results*, pages 61–74. Springer.
- Zhang, Z., Yeming, G., and Zailin, G. (2022). Clearing function-based simulation optimization for release planning under digital twin wafer fabs. *IFAC-PapersOnLine*, 55(10):2539–2544.
- Zheng, Y.-S. (1991). A simple proof for optimality of (s, s) policies in infinite-horizon inventory systems. *Journal of Applied Probability*, 28(4):802–810.
- Ziarnetzky, T., Moench, L., and Uzsoy, R. (2020). Simulation-based performance assessment of production planning models with safety stock and forecast evolution in semiconductor wafer fabrication. *IEEE TRANSACTIONS ON SEMICONDUCTOR MANUFACTURING*, 33(1):1–12.
- Ziarnetzky, T. and Mönch, L. (2016). Simulation-based optimization for integrated production planning and capacity expansion decisions. In *2016 Winter Simulation Conference (WSC)*, pages 2992–3003. IEEE.
- Ziarnetzky, T., Mönch, L., and Uzsoy, R. (2018). Rolling horizon, multi-product production planning with chance constraints and forecast evolution for wafer fabs. *International Journal of Production Research*, 56(18):6112–6134.
- Zipkin, P. H. (2000). *Foundations of inventory management*. The IRWIN/MCGRAW-HILL series.

## APPENDIX

# Appendix A

## Acronyms

A summary of all acronyms is documented in Table A.1.

Table A.1: A summary of acronyms used in alphabetical order

Acronym	Abbreviation
Approximate Dynamic Programming	ADP
Binary Search	BS
Chance Constraint	CC
Clearing function	CF
Cycle time	CT
Dynamic Programming	DP
Finished Goods	FG
Non-Linear Programming	NLP
Parallel Computing	PC
Production Planning	PP
Raw Material	RM
Reinforcement Learning	RL
Release	R
Robust Optimization	RO
Single Period Problem	SPP
Stochastic Dynamic Programming	SDP
Work in Process	WIP

## A.1 Demand Discretization

For probability distributions with infinite support, we begin by establishing upper and lower bounds on the support. We then specify a discretization grid size of  $\delta$ , and proceed as follows:

**Step 1:** Distributions that support a real line, such as the Normal distribution, involve positive probabilities of negative demand, which must be suppressed. The occurrence of negative demands is invalid in case of any production inventory system. Therefore, we assume that the probability of negative demand values is very low, and assign the sum of all probabilities associated with negative demand to  $D = 0$ .

Tables A.2 and A.3 show a sample illustration of discretization procedure with mean demand and variance as inputs and how the expected demand of PMF is close to the true mean of the continuous distribution. We use two values of squared coefficient of variation (SCV)  $\in \{0.16, 0.04\}$  and four demand profiles  $\in \{\text{mean rising(MR)}, \text{mean decline(MD)}, \text{mean constant(MC)}, \text{mean peak (MP)}\}$ , respectively. After discretization, for all experimental conditions, the expected mean demand agreed to two decimal places with the average deviation (Root Mean Square Error) of 0.005631 and maximum deviation of 0.000122 between the input continuous distribution and the output discrete distribution. However, it is interesting to note the variance of discretized distribution varies with SCV values. We noticed the average deviation (RMSE) being 0.10457 and a maximum deviation of 0.071175 between the input continuous distribution and the output discrete distribution. For higher SCV values of the continuous distribution, we get lower variance in the PMF. In contrast, for low SCV values in the continuous distribution, after discretization, we get a higher variance across all demand patterns. The rounding effect is shown in line 13 and the step size  $\delta$  in line 9 of Algorithm 4 plays a significant role. i.e., the rounding of the middle value gives the integer demand values and the step size divides each grid by a strip length which determines the accuracy of discretization. We observe the truncation at both the upper and lower tail end of the distribution as per Step 1. The PMF's corresponding mean and standard deviation is close to the continuous distribution's true mean and standard deviation.

**Step 2:** For better accuracy, we keep the grid size  $\delta < 1$  to get finer discretization limits for each discrete demand value in the mass function. However, when  $\delta < 1$ , i.e., we take a small grid size, we get duplicate demand values when we round the average values of  $x$  and  $y$  per line 13 in Algorithm 4. For example, when  $\delta = 0.25$ , and initial

$x = 0.34$ , then  $y = 1.09$ , using line 13, we get  $D_1 = 0$  and when we iterate again, we get  $x = 0.59$  and  $y = 0.84$  results in  $D_2 = 1$ . However, in the next iteration, we get  $x = 0.84$ ,  $y = 1.09$ , and  $D_3 = 1$  which is a duplicate demand as per  $D_2$ . Therefore, for convenience, we use step size  $\delta = 0.5$ , and thus, we get a discrete integer demand distribution as a probability mass function after rounding the mid-value.

**Step: 3** Demand probabilities are checked on the upper tail of the distribution. If demand probabilities are lower than the hyperparameter  $accuracy=10^{-5}$ , we truncate the demand values. The probabilities are summed to update the probability corresponding to the upper limit demand value. This step gives a demand distribution truncated at the upper limit of the demand values corresponding to the demand mass function.

---

**Algorithm 4** Discretization subroutine used in Step 6 of Algorithm 2

---

```
1: procedure DISCRETIZATION( $n, l, \text{accuracy}, \text{input parameters}$ )
2:   Initialize hyperparameters:  $l = 5$ ,  $\delta = 0.5$  and  $\text{accuracy} = 10^{-5}$ 
3:   for  $i := 1$  to  $T$  do ▷  $T$  periods
4:     Calculate  $UL(i)$  ▷ using input parameters
5:     Calculate  $LL(i)$  ▷ using input parameters
6:   end for
7:   for  $j := 1$  to  $T$  do
8:      $x \leftarrow LL(j)$ 
9:      $steps \leftarrow \delta$  ▷  $\delta = 0.5$ 
10:     $y \leftarrow x + steps$ 
11:     $i \leftarrow 1$ 
12:    while  $y \leq UL(j)$  do ▷ Main loop
13:       $D(i, 1) \leftarrow \text{round}\left(\frac{x+y}{2}\right)$ 
14:       $D(i, 2) \leftarrow \text{cdf}(y) - \text{cdf}(x)$  ▷ CDF:  $P(X \leq x)$ 
15:       $x \leftarrow y$ 
16:       $y \leftarrow x + steps$ 
17:       $i \leftarrow i + 1$ 
18:    end while
19:    Add demand probabilities  $\leq 0$  to  $D(1)$  in the lower tail ▷ Step 1
20:    for  $t := 1$  to  $\text{numel}(D(:, 1))$  do ▷ Step 2
21:      Check  $D(i, 1)$  and its probability in  $D(i, 2)$ 
22:      Rearrange the integer demand values in  $D(i, 1)$ 
23:      Add associated probabilities to the corresponding  $D(i, 1)$ 
24:      Update the demand mass function,  $D$ 
25:    end for
26:    if  $D(i, 2) \leq \text{accuracy}$  then ▷ Step 3
27:      Truncate demand probabilities at the tail end of the distribution
28:      Update  $D(\text{end}, 2)$ ; Upper limit by adding demand probabilities  $< \text{accuracy}$ 
29:    end if
30:    Update  $D(j)$ 
31:  end for
32:  return  $D(t)$  ▷  $D$  is the discretized demand
33: end procedure
```

---

A summary of demand discretization is shown below as a sample illustration.

Table A.2: Input continuous demand distribution for 10-period problem

Expt #	SCV	Demand	Periods	1	2	3	4	5	6	7	8	9	10
1	0.16	MR	$\mu$	6	7	8	9	10	10	11	12	13	14
			$\sigma^2$	5.76	7.84	10.24	12.96	16	16	19.36	23.04	27.04	31.36
2	0.16	MD	$\mu$	14	13	12	11	10	10	9	8	7	6
			$\sigma^2$	31.36	27.04	23.04	19.36	16	16	12.96	10.24	7.84	5.76
3	0.16	MC	$\mu$	10	10	10	10	10	10	10	10	10	10
			$\sigma^2$	16	16	16	16	16	16	16	16	16	16
4	0.16	MP	$\mu$	9	9	9	9	10	13	13	10	9	9
			$\sigma^2$	12.96	12.96	12.96	12.96	16	27.04	27.04	16	12.96	12.96
5	0.04	MR	$\mu$	6	7	8	9	10	10	11	12	13	14
			$\sigma^2$	1.44	1.96	2.56	3.24	4	4	4.84	5.76	6.76	7.84
6	0.04	MD	$\mu$	14	13	12	11	10	10	9	8	7	6
			$\sigma^2$	7.84	6.76	5.76	4.84	4	4	3.24	2.56	1.96	1.44
7	0.04	MC	$\mu$	10	10	10	10	10	10	10	10	10	10
			$\sigma^2$	4	4	4	4	4	4	4	4	4	4
8	0.04	MP	$\mu$	9	9	9	9	10	13	13	10	9	9
			$\sigma^2$	3.24	3.24	3.24	3.24	4	6.76	6.76	4	3.24	3.24

Table A.3: Discretized demand distribution for 10 period problem

Expt #	SCV	Demand	Periods	1	2	3	4	5	6	7	8	9	10
1	0.16	MR	Mean	6.005	7.005	8.006	9.007	10.008	10.008	11.009	12.009	13.010	14.011
			Var	5.782	7.838	10.211	12.901	15.907	15.907	19.228	22.867	26.822	31.093
2	0.16	MD	Mean	14.011	13.010	12.009	11.009	10.008	10.008	9.007	8.006	7.005	6.005
			Var	31.093	26.822	22.867	19.228	15.907	15.907	12.901	10.211	7.838	5.782
3	0.16	MC	Mean	10.008	10.008	10.008	10.008	10.008	10.008	10.008	10.008	10.008	10.008
			Var	15.907	15.907	15.907	15.907	15.907	15.907	15.907	15.907	15.907	15.907
4	0.16	MP	Mean	9.007	9.007	9.007	9.007	10.008	13.010	13.010	10.008	9.007	9.007
			Var	12.901	12.901	12.901	12.901	15.907	26.822	26.822	15.907	12.901	12.901
5	0.04	MR	Mean	6.000	7.000	8.000	9.000	10.000	10.000	11.000	12.000	13.000	14.000
			Var	1.523	2.043	2.643	3.323	4.083	4.083	4.923	5.843	6.843	7.923
6	0.04	MD	Mean	14.000	13.000	12.000	11.000	10.000	10.000	9.000	8.000	7.000	6.000
			Var	7.923	6.843	5.843	4.923	4.083	4.083	3.323	2.643	2.043	1.523
7	0.04	MC	Mean	10.000	10.000	10.000	10.000	10.000	10.000	10.000	10.000	10.000	10.000
			Var	4.083	4.083	4.083	4.083	4.083	4.083	4.083	4.083	4.083	4.083
8	0.04	MP	Mean	9.000	9.000	9.000	9.000	10.000	13.000	13.000	10.000	9.000	9.000
			Var	3.323	3.323	3.323	3.323	4.083	6.843	6.843	4.083	3.323	3.323

## A.2 Original SDP results

The following tables show the benchmark cost for different demand patterns by varying the SCV,  $b$ , and  $K_2$  values.

Table A.4: Expected cost for 0.5 grid in state and 0.1 grid in action for mean demand of 5 using  $K_2 \in \{10, 30\}$

Expt #	Demand	SCV	$b$	$K_2=10$	$K_2=30$
E1	MR	0.16	6	209.159	274.626
E2	MD	0.16	6	246.127	388.64
E3	MC	0.16	6	207.14	288.667
E4	MP	0.16	6	215.117	301.793
E5	MR	0.04	6	197.761	260.881
E6	MD	0.04	6	209.979	338.3288
E7	MC	0.04	6	191.435	259.0604
E8	MP	0.04	6	197.4297	274.078
E9	MR	0.16	30	235.197	321.823
E10	MD	0.16	30	351.438	593.731
E11	MC	0.16	30	227.607	334.844
E12	MP	0.16	30	234.2978	346.984
E13	MR	0.04	30	212.072	289.179
E14	MD	0.04	30	250.567	454.4
E15	MC	0.04	30	200.17	275.9808
E16	MP	0.04	30	205.459	292.1199

Table A.5: Expected cost for 0.5 grid in state and 0.1 grid in action for mean demand of 8 using  $K_2 \in \{10, 30\}$

Expt #	Demand	SCV	$b$	$K_2=10$	$K_2=30$
E1	MR	0.16	6	406.21	582.43
E2	MD	0.16	6	753.09	1108.94
E3	MC	0.16	6	487.45	757.33
E4	MP	0.16	6	505.01	776.32
E5	MR	0.04	6	377.04	546.94
E6	MD	0.04	6	675.08	1073.68
E7	MC	0.04	6	408.87	686.03
E8	MP	0.04	6	432.70	714.18
E9	MR	0.16	30	547.02	937.35
E10	MD	0.16	30	1917.11	2952.51
E11	MC	0.16	30	788.28	1505.59
E12	MP	0.16	30	831.99	1594.85
E13	MR	0.04	30	448.08	762.75
E14	MD	0.04	30	1523.76	2604.94
E15	MC	0.04	30	497.61	1094.73
E16	MP	0.04	30	548.05	1224.81

### A.3 BS-based SDP results

The following tables show the BS-based SDP cost and computation times for different demand patterns by varying the SCV,  $b$ , and  $K_2$  values. We tabulate the computation time in minutes and the expected cost for  $(0, 0)$  in period 1. We use an integer and 0.5 grid size on state space while computing the optimal policy.

Table A.6: We report the expected cost and computation times (minutes) for BS-based SDP using an integer grid and 0.5 grid on state space for mean demand of 5 and  $K_2 = 10$

Expt #	Demand	SCV	$b$	Integer grid on state		0.5 grid on state	
				cost	time	cost	time
E1	MR	0.16	6	209.73	3.3	209.48	13.6
E2	MD	0.16	6	246.41	3.4	245.87	13.3
E3	MC	0.16	6	207.63	3.0	207.31	9.2
E4	MP	0.16	6	215.69	4.5	215.37	14.7
E5	MR	0.04	6	197.91	2.6	197.73	8.7
E6	MD	0.04	6	210.60	2.4	209.82	8.6
E7	MC	0.04	6	191.74	3.0	191.47	5.8
E8	MP	0.04	6	197.69	3.7	197.39	9.5
E9	MR	0.16	30	235.58	3.6	235.28	12.3
E10	MD	0.16	30	351.72	3.4	350.76	12.0
E11	MC	0.16	30	228.07	2.4	227.61	7.9
E12	MP	0.16	30	234.58	4.0	234.23	13.8
E13	MR	0.04	30	212.31	2.4	212.04	8.3
E14	MD	0.04	30	251.56	2.3	250.49	8.2
E15	MC	0.04	30	200.47	1.8	200.15	6.8
E16	MP	0.04	30	205.80	4.6	205.44	9.4

Table A.7: We report the expected cost and computation times (minutes) for BS-based SDP using an integer grid and 0.5 grid on state space for mean demand of 5 and  $K_2 = 30$

Expt #	Demand	SCV	$b$	Integer grid on state		0.5 grid on state	
				cost	time	cost	time
E1	MR	0.16	6	288.11	2.7	289.36	11.5
E2	MD	0.16	6	392.69	2.7	391.64	11.0
E3	MC	0.16	6	298.78	1.8	298.04	7.7
E4	MP	0.16	6	310.89	3.0	310.22	13.1
E5	MR	0.04	6	275.26	1.7	274.75	8.0
E6	MD	0.04	6	340.95	1.8	339.44	7.2
E7	MC	0.04	6	271.42	1.4	270.66	4.6
E8	MP	0.04	6	284.00	2.0	283.24	7.4
E9	MR	0.16	30	322.73	2.6	322.06	11.8
E10	MD	0.16	30	594.31	2.5	591.83	11.2
E11	MC	0.16	30	335.75	1.8	334.42	7.6
E12	MP	0.16	30	347.82	2.9	346.66	12.9
E13	MR	0.04	30	289.72	1.8	289.06	7.4
E14	MD	0.04	30	455.80	1.8	453.72	6.8
E15	MC	0.04	30	276.69	1.4	275.66	4.7
E16	MP	0.04	30	292.88	1.9	291.71	7.6

## A.4 $R$ as action-based SDP results

Table A.8 gives an overview of the computation time of various settings discussed in Section 4.7.1 for different parameters. The results are presented for a full factorial design of mean demand, SCV, and SL for a Missbauer CF using  $R$  as action. We compare the effects of parallel computing and BS-based SDP algorithm with the standard SDP. The state space  $X$  denotes the number of FGI states, and  $W$  denotes the number of ending WIP states. Since we use  $R$  as an action with an integer grid, we enumerate 91 possible values of  $R \in \{0, 1, \dots, 90\}$ .

Table A.8 shows the computational run time for  $R$  as an action using Missbauer CF. We increase the  $W$  state space to 101 and  $R$  state space to 91 to cover large workload requirements in a Missbauer CF. The BS procedure shows significant improvement over the conventional SDP algorithm. On average, we achieve a 96% reduction in run time between NoPC and BS-NoPC, a 54% reduction between 72coresPC and BS-72coresPC, and using a parallel computing framework, we achieve a 66% reduction between 72coresPC and BS-NoPC, respectively. It is difficult to make a general remark on runtime for demand patterns and other parameters. We use integer rounding (floor) on the  $R^*$  calculation to get a policy similar to the original SDP algorithm. Based on the experimental investigation, this reduces the error accumulation from period  $T$  to 1, and we get an expected cost close to the actual cost of the original SDP algorithm for a given initial state. It is interesting to witness the effect of an efficient BS algorithm in BS-NoPC and BS-72coresPC, where the problem structure is exploited. Due to high overheads on computing using PC, BS-NoPC outperforms BS-72coresPC in terms of run time. We get significant run time savings for the "Do Nothing" policy.

Table A.8: Computational times (seconds) of the 10-period numerical for different demand, SCV, and SL parameters using  $R$  as action and a Missbauer CF of  $K_1 = 10$

Expt #	Demand	SL	SCV	$X$	$W$	NoPC	72coresPC	BS-NoPC	BS-72coresPC
E1	MR	0.95	0.16	1111	101	13419	1443	456	656
E2	MD	0.95	0.16	1111	101	13455	1474	563	758
E3	MC	0.95	0.16	1031	101	13850	1359	441	548
E4	MP	0.95	0.16	1141	101	15357	1515	614	795
E5	MR	0.95	0.04	1061	101	10732	1108	313	435
E6	MD	0.95	0.04	1061	101	10786	1168	355	520
E7	MC	0.95	0.04	1001	101	11161	1073	369	419
E8	MP	0.95	0.04	1081	101	11467	1237	382	537
E9	MR	0.9	0.16	1111	101	15321	1415	612	829
E10	MD	0.9	0.16	1111	101	15376	1430	508	722
E11	MC	0.9	0.16	1031	101	13279	1302	494	575
E12	MP	0.9	0.16	1141	101	14774	1577	613	843
E13	MR	0.9	0.04	1061	101	10848	1192	424	559
E14	MD	0.9	0.04	1061	101	10970	1197	353	490
E15	MC	0.9	0.04	1001	101	10252	1038	315	410
E16	MP	0.9	0.04	1081	101	11758	1211	371	550

In this experiment, it is difficult to make a general remark on runtime concerning different demand patterns and other parameters. However, it depends primarily on the state-action pairs and the number of arithmetic operations required for the SDP algorithm. At the same time, the policy looks similar between the BS method and the original SDP when the step size is 0.01. Therefore, for practical purposes, we prefer to use the BS-NoPC-based SDP implementation and use the results as a benchmark to verify the optimality gap for other approximation algorithms.

## A.5 Sensitivity analysis - $R$ and $P$ as action

Continuing the results shown in Chapter 3 for the performance analysis of the UCPI and CPI solution, we show the results for  $W_0 = 10$  using Missbauer CF by varying  $K_2$  between 10 and 30 using  $R$  and  $P$  as actions in the CF-based PI model. We observe a similar behavior shown in Chapter 3 for  $W_0 = 10$  in  $R$  as action. Tables A.9 and A.10 refer to the results for UCPI solution using  $R$  and  $P$  as actions.

Table A.9: Comparison of UCPI solution for demand  $\sim \mathcal{N}(9, 1)$  using  $R$  as action with (a)  $K_2=10$  (b)  $K_2=30$

$(X_0, 10)$	$R^*$	$P$	Exp. Cost	Optimal cost	$R$ as action
-9	9.23	5.74	391.88	363.69	7%
-8	8.23	5.58	365.15	333.69	9%
-7	7.23	5.42	338.42	303.69	10%
-6	6.23	5.24	312.26	273.69	12%
-5	5.23	5.05	286.38	243.69	15%
-4	4.23	4.84	261.06	213.69	18%
-3	3.23	4.63	235.74	183.69	22%
-2	2.23	4.39	211.27	153.70	27%
-1	1.23	4.15	186.8	123.85	34%
0	0.23	3.88	163.18	95.32	42%
1	0.00	3.82	134.59	70.89	47%
2	0.00	3.82	104.59	52.16	50%
3	0.00	3.82	74.74	38.33	49%
4	0.00	3.82	46.36	27.92	40%
5	0.00	3.82	24.14	19.72	18%
6	0.00	3.82	12.95	12.95	0%
7	0.00	3.82	10.14	9.96	2%
8	0.00	3.82	10.05	8.21	18%
9	0.00	3.82	10.33	6.46	37%
10	0.00	3.82	10.63	4.71	56%

$(X_0, 10)$	$R^*$	$P$	Exp. Cost	Optimal cost	$R$ as action
-9	9.23	3.45	456.57	423.75	7%
-8	8.23	3.32	429	393.75	8%
-7	7.23	3.19	401.42	363.75	9%
-6	6.23	3.05	374.13	333.75	11%
-5	5.23	2.91	346.83	303.75	12%
-4	4.23	2.77	319.54	273.75	14%
-3	3.23	2.61	292.81	243.75	17%
-2	2.23	2.46	265.79	213.75	20%
-1	1.23	2.30	239.06	183.75	23%
0	0.23	2.13	212.62	153.78	28%
1	0.00	2.09	183.46	124.22	32%
2	0.00	2.09	153.46	96.73	37%
3	0.00	2.09	123.46	73.31	41%
4	0.00	2.09	93.47	54.31	42%
5	0.00	2.09	63.78	39.02	39%
6	0.00	2.09	36.45	26.54	27%
7	0.00	2.09	16.96	16.13	5%
8	0.00	2.09	8.61	8.61	0%
9	0.00	2.09	6.99	6.46	7%
10	0.00	2.09	7.09	4.71	34%

We find a small reduction in the expected cost using a UCPI-based solution when  $W_0 = 10$ . That is, any additional workload will impact the output and hence reduces the objective function cost. Comparing the expected cost between the UCPI models when  $W_0 = 0$  and  $W_0 = 10$ , we observe less optimality gap in  $R$  as action. On the other hand, we observe a significant increase in the optimality gap using  $P$  as action when  $W_0 = 10$ . The additional workload is insufficient to produce the necessary output, given the small amount of release at the beginning of the period. Therefore, we witness an increase in the optimality gap between the shape parameters and when  $W_0 = 10$ .

Tables A.11 and A.12 refer to the results comparison of the CPI solution using  $R$  and  $P$  as actions when  $W_0 = 10$ . In  $R$  as action, we find a large optimality gap when  $X_0 < 0$  and  $K_2 = 10$ . Similarly, in  $P$  as action, we find a similar trend to the UCPI-based solution analysis, and we find similar solutions between the CPI and UCPI solutions when

$P^* < P_{max}$  since both CPI and UCPI models yield the same  $P^*$ .

Table A.10: Comparison of UCPI solution for demand  $\sim \mathcal{N}(9, 1)$  using  $P$  as action with (a)  $K_2=10$  (b)  $K_2=30$

$(X_0, 10)$	$R$	$P^*$	Exp. Cost	Optimal cost	$P$ as action
-9	Infeasible	19.23	Infeasible	363.69	-
-8	Infeasible	18.23	Infeasible	333.69	-
-7	Infeasible	17.23	Infeasible	303.69	-
-6	Infeasible	16.23	Infeasible	273.69	-
-5	Infeasible	15.23	Infeasible	243.69	-
-4	Infeasible	14.23	Infeasible	213.69	-
-3	Infeasible	13.23	Infeasible	183.69	-
-2	Infeasible	12.23	Infeasible	153.70	-
-1	Infeasible	11.23	Infeasible	123.85	-
0	Infeasible	10.23	Infeasible	95.32	-
1	186.92	9.49	254.11	70.89	72%
2	54.87	8.49	87.30	52.16	40%
3	27.39	7.49	51.20	38.33	25%
4	15.01	6.49	33.98	27.92	18%
5	7.68	5.49	23.06	19.72	15%
6	2.65	4.49	15.03	12.95	14%
7	0.00	3.49	9.96	9.96	0%
8	0.00	2.49	8.21	8.21	0%
9	0.00	1.49	6.46	6.46	0%
10	0.00	0.49	4.71	4.71	0%

$(X_0, 10)$	$R$	$P^*$	Exp. Cost	Optimal cost	$P$ as action
-9	Infeasible	19.23	Infeasible	423.75	-
-8	Infeasible	18.23	Infeasible	393.75	-
-7	Infeasible	17.23	Infeasible	363.75	-
-6	Infeasible	16.23	Infeasible	333.75	-
-5	Infeasible	15.23	Infeasible	303.75	-
-4	Infeasible	14.23	Infeasible	273.75	-
-3	Infeasible	13.23	Infeasible	243.75	-
-2	Infeasible	12.23	Infeasible	213.75	-
-1	Infeasible	11.23	Infeasible	183.75	-
0	Infeasible	10.23	Infeasible	153.78	-
1	561.76	9.49	722.66	124.22	83%
2	167.63	8.49	228.25	96.73	58%
3	87.18	7.49	125.94	73.31	42%
4	52.05	6.49	80.28	54.31	32%
5	32.06	5.49	53.54	39.02	27%
6	18.97	4.49	35.43	26.54	25%
7	9.60	3.49	21.96	16.13	27%
8	2.46	2.49	11.29	8.61	24%
9	0.00	1.49	6.46	6.46	0%
10	0.00	0.49	4.71	4.71	0%

Table A.11: Comparison of CPI solution for demand  $\sim \mathcal{N}(9, 1)$  using  $R$  as action with (a)  $K_2=10$  (b)  $K_2=30$

$(X_0, 10)$	$R^*$	$P$	Exp. Cost	Optimal cost	$R$ as action
-9	0.00	3.82	434.59	363.69	16%
-8	0.00	3.82	404.59	333.69	18%
-7	0.00	3.82	374.59	303.69	19%
-6	0.00	3.82	344.59	273.69	21%
-5	0.00	3.82	314.59	243.69	23%
-4	0.00	3.82	284.58	213.69	25%
-3	0.00	3.82	254.58	183.69	28%
-2	0.00	3.82	224.59	153.70	32%
-1	0.00	3.82	194.59	123.85	36%
0	0.00	3.82	164.59	95.32	42%
1	0.00	3.82	134.59	70.89	47%
2	0.00	3.82	104.59	52.16	50%
3	0.00	3.82	74.74	38.33	49%
4	0.00	3.82	46.36	27.92	40%
5	0.00	3.82	24.14	19.72	18%
6	0.00	3.82	12.95	12.95	0%
7	0.00	3.82	10.14	9.96	2%
8	0.00	3.82	10.05	8.21	18%
9	0.00	3.82	10.33	6.46	37%
10	0.00	3.82	10.63	4.71	56%

$(X_0, 10)$	$R^*$	$P$	Exp. Cost	Optimal cost	$R$ as action
-9	0.00	2.09	483.46	423.75	12%
-8	0.00	2.09	453.46	393.75	13%
-7	0.00	2.09	423.46	363.75	14%
-6	0.00	2.09	393.46	333.75	15%
-5	0.00	2.09	363.46	303.75	16%
-4	0.00	2.09	333.46	273.75	18%
-3	0.00	2.09	303.46	243.75	20%
-2	0.00	2.09	273.46	213.75	22%
-1	0.00	2.09	243.46	183.75	25%
0	0.00	2.09	213.46	153.78	28%
1	0.00	2.09	183.46	124.22	32%
2	0.00	2.09	153.46	96.73	37%
3	0.00	2.09	123.46	73.31	41%
4	0.00	2.09	93.47	54.31	42%
5	0.00	2.09	63.78	39.02	39%
6	0.00	2.09	36.45	26.54	27%
7	0.00	2.09	16.96	16.13	5%
8	0.00	2.09	8.61	8.61	0%
9	0.00	2.09	6.99	6.46	7%
10	0.00	2.09	7.09	4.71	34%

Table A.12: Comparison of CPI solution for demand  $\sim \mathcal{N}(9, 1)$  using  $P$  as action with (a)  $K_2=10$  (b)  $K_2=30$

$(X_0, 10)$	$R$	$P^*$	Exp. Cost	Optimal cost	$P$ as action
-9	$\infty$	10.00	$\infty$	363.69	-
-8	$\infty$	10.00	$\infty$	333.69	-
-7	$\infty$	10.00	$\infty$	303.69	-
-6	$\infty$	10.00	$\infty$	273.69	-
-5	$\infty$	10.00	$\infty$	243.69	-
-4	$\infty$	10.00	$\infty$	213.69	-
-3	$\infty$	10.00	$\infty$	183.69	-
-2	$\infty$	10.00	$\infty$	153.70	-
-1	$\infty$	10.00	$\infty$	123.85	-
0	$\infty$	10.00	$\infty$	95.32	-
1	186.92	9.49	254.11	70.89	72%
2	54.87	8.49	87.30	52.16	40%
3	27.39	7.49	51.20	38.33	25%
4	15.01	6.49	33.98	27.92	18%
5	7.68	5.49	23.06	19.72	15%
6	2.65	4.49	15.03	12.95	14%
7	0.00	3.49	9.96	9.96	0%
8	0.00	2.49	8.21	8.21	0%
9	0.00	1.49	6.46	6.46	0%
10	0.00	0.49	4.71	4.71	0%

$(X_0, 10)$	$R$	$P^*$	Exp. Cost	Optimal cost	$P$ as action
-9	$\infty$	10.00	$\infty$	423.75	-
-8	$\infty$	10.00	$\infty$	393.75	-
-7	$\infty$	10.00	$\infty$	363.75	-
-6	$\infty$	10.00	$\infty$	333.75	-
-5	$\infty$	10.00	$\infty$	303.75	-
-4	$\infty$	10.00	$\infty$	273.75	-
-3	$\infty$	10.00	$\infty$	243.75	-
-2	$\infty$	10.00	$\infty$	213.75	-
-1	$\infty$	10.00	$\infty$	183.75	-
0	$\infty$	10.00	$\infty$	153.78	-
1	561.76	9.49	722.66	124.22	83%
2	167.63	8.49	228.25	96.73	58%
3	87.18	7.49	125.94	73.31	42%
4	52.05	6.49	80.28	54.31	32%
5	32.06	5.49	53.54	39.02	27%
6	18.97	4.49	35.43	26.54	25%
7	9.60	3.49	21.96	16.13	27%
8	2.46	2.49	11.29	8.61	24%
9	0.00	1.49	6.46	6.46	0%
10	0.00	0.49	4.71	4.71	0%

## A.6 Using Graves CF

We use Graves CF as a special case of an uncapacitated clearing function. In the following subsection, we discuss the solution characterization and the expected cost to illustrate how Graves CF-based solution affects the production inventory system without capacity restrictions compared to the capacitated Missbauer CF. Since Graves CF-based model is an uncapacitated version, we compare the results of the UCPI model and analyze the solution with various characteristics. Graves CF uses constant cycle time to yield production output. We compare the performance of the Graves CF with respect to the uncapacitated PI model to draw insights.

The details of the Graves CF are as follows. Graves (1986) uses a constant cycle time-based clearing function to model production behavior (Missbauer and Uzsoy 2020b). The Graves CF processes a fraction of the workload as production and assumes a fixed lead time of  $\frac{1}{\alpha}$  that can be maintained regardless of the workload level, with no limit for the maximum production capacity. Production takes place as long as a sufficient workload is available for the production resource. Let  $X_0$  and  $W_0$  represent the initial FG and in-process inventory. The Graves CF is defined by,

$$P = \alpha(W_0 + R)$$

where the domain and the range of  $P$  are  $[0, \infty)$  and  $[0, \infty)$ . and its inverse by

$$f^{-1}(\Lambda) = \frac{P}{\alpha} \tag{A.1}$$

The domain of  $f^{-1}(\Lambda) = [0, \infty)$  and the range of  $f^{-1}(\Lambda) = [0, \infty)$ . Figure A.1 shows the Graves CF for two fixed lead time  $\alpha = 0.5$ , and  $0.25$ . Since there is no capacity bound, a fraction  $\alpha$  of workload is produced to replenish the FGI. The slope of the function determines the expected production and the two lines shown in the Figure result in one-fourth and one-half of the workload as output production. For instance, to produce 10 units of production, we need 20 units of workload when  $\alpha = 0.5$  and we need 40 units of workload when  $\alpha = 0.25$ , respectively. Using these CF parameters, we calculate the expected cost and optimal parameters in the example illustrated below.

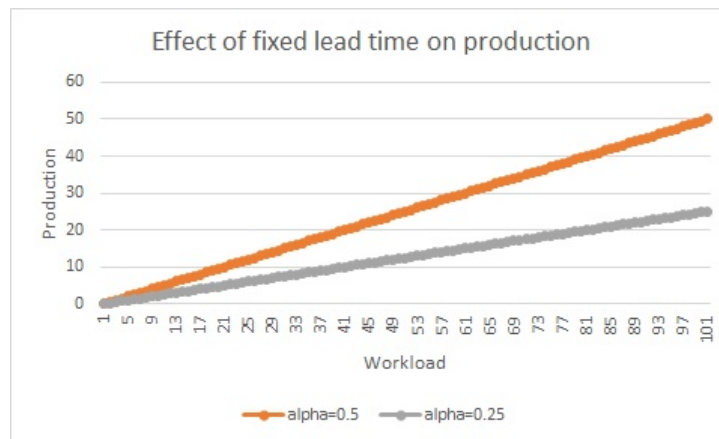


Figure A.1: Effect of  $\alpha$  using Graves CF

### A.6.1 Graves CF - Solution characterization

We discuss the optimal solution characterization using the uncapacitated Graves CF by varying the slope  $\alpha$  of the CF. We use the E3 cost parameters to illustrate the impact of CF using the Graves CF with  $\alpha = 0.5$  and  $0.25$ . Since the Graves CF defines no maximum production capacity, the actual production is given by  $P = f(W_0 + R)$ . The optimal production and the planned releases are summarized below for the initial states  $(X_0, W_0)$ . The computational run time was similar to the Missbauer CF experiments. Under the same experimental conditions, the optimal solution structure follows one of

the three cases.

Figures A.2 and A.3 illustrate the optimal solution based on the optimality conditions derived in Section 3.5. In the first experiment, we use  $b = 6$  in E3 cost inputs to share a sample of experimental output for states  $X_0 \in \{-7, \dots, 14\}$  and  $W_0 \in \{0, 1, 2, 5, 10, 20, 40, 60, 80\}$  with mean demand of 7 and standard deviation of 1, respectively. We use two levels of Graves CF parameter  $\alpha \in \{0.5, 0.25\}$  throughout all the experiments. The three optimality cases are shown as follows: blue for Case 1, yellow for Case 2, and green for Case 3, respectively. For instance, in Figure A.2(a), (1, 10) follows Case 1, and (4, 10) follows Case 2. Similarly, in Figure A.2(b), we observe Case 3 when  $X_0 \geq 8$  irrespective of  $W_0$ , and  $(X_0, 0)$ . Finally we observe Case 2 for all the cases when  $W_0 > 0$  and  $X_0 < 8$ .

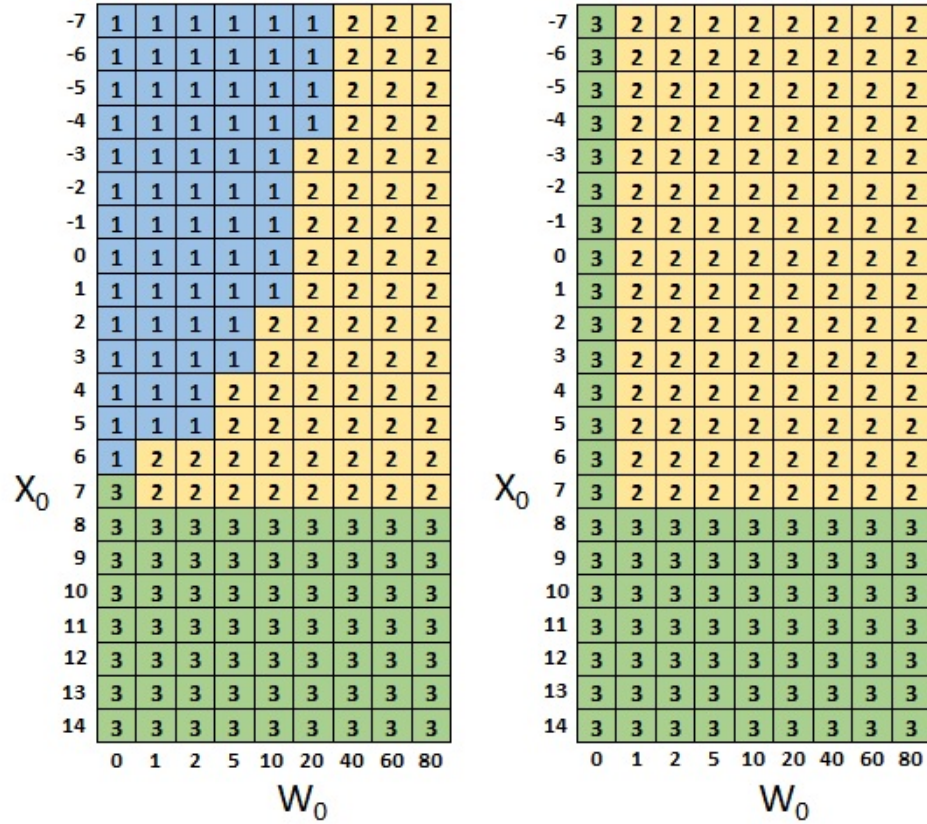


Figure A.2: For demand  $\sim \mathcal{N}(7, 1)$  (a) For  $\alpha=0.5$  (b) For  $\alpha=0.25$

From Figure A.2(a), we find that the optimal solution depends on the initial states and the value of  $\alpha$ . When  $\alpha = 0.5$ , we observe the policy follows Case 1 when sufficient

FGI and WIP are unavailable, and hence additional releases are necessary. Similarly, when sufficient WIP is available, it is optimal not to increase the workload with new releases; production takes place from the in-process inventory resulting in the optimal expected cost. Lastly, we observe the "do nothing" policy when there is sufficiently large FGI, i.e., when  $X_0 \geq 8$ , so it is optimal not to produce, and no additional releases are necessary.

When  $\alpha = 0.25$ , the system dynamics change concerning input costs and the CF parameter. For example, the optimal condition falls into Case 3 when there is no WIP in the system and  $X_0 \geq 8$  across all WIP states. However, we observe Case 2 as the optimal condition for all FG states between -7 and 7 across all initial WIP except  $W_0 = 0$ . Here, it is optimal not to add any additional releases, and production takes place with the existing WIP since it takes a long lead time to process the workload. i.e., the system expects a sufficiently large workload to convert raw material into FGI. The costs trade-off between the cost of the backorder and the workload cost affects the optimal releases and production quantity. Under such situations, we get minimum objective function cost when the policy follows "do nothing". However, for all the other cases, we observe Case 2  $R^* = 0, P^* > 0$  as the optimal condition. This result implies that production is necessary when there is a non-zero WIP in the system, which results in a relatively lower cost than Case 3, where the optimal condition follows a do-nothing policy.

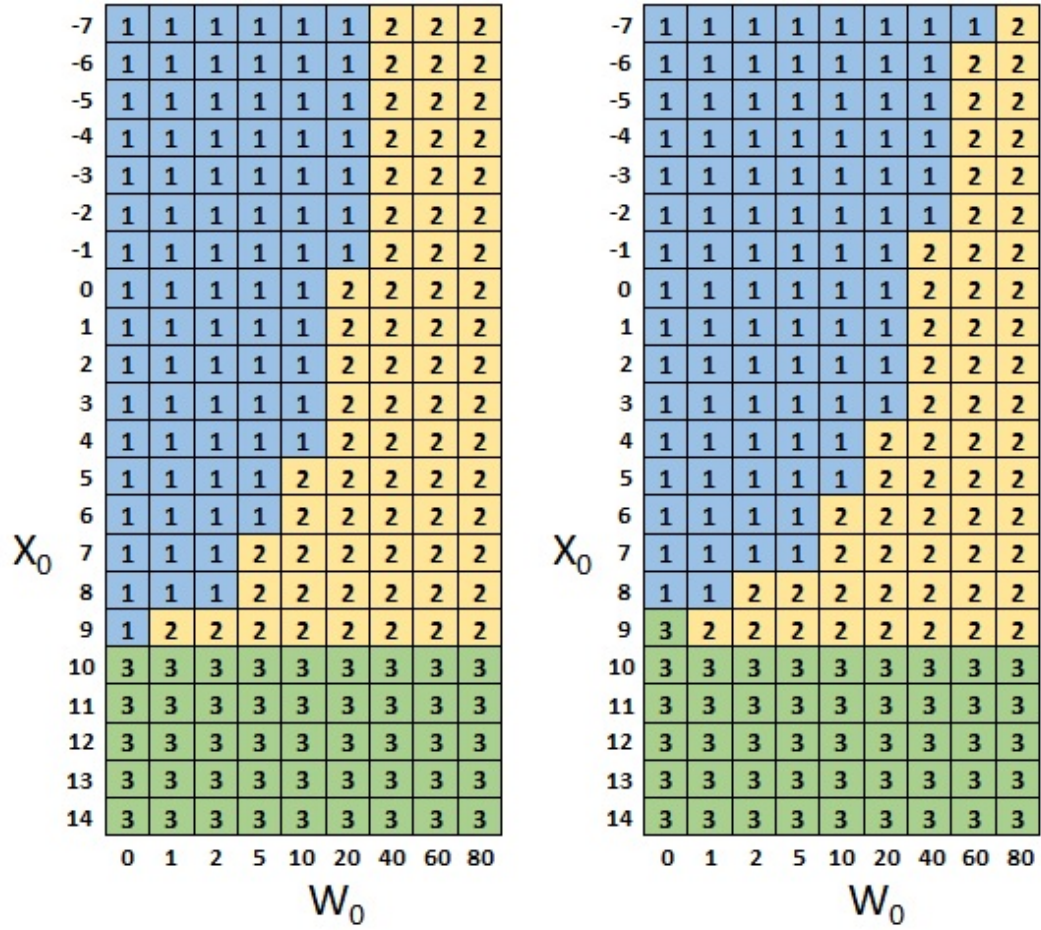


Figure A.3: For demand  $\sim \mathcal{N}(7, 2)$  (a) For  $\alpha=0.5$  (b) For  $\alpha=0.25$

It is interesting to validate the role of backorder cost in finding the optimal release and production under less fraction of the workload. For instance, when  $b = 30$ , we find the optimal solution changes substantially when  $\alpha = 0.25$ , i.e., for producing one unit of output, we need 4 times of workload, given by the planned lead time of 4 periods. Figures A.3(a) and (b) show the optimal conditions for  $\mathcal{N}(7, 2)$ . We compare the optimality conditions in Figures A.2 and A.3 (b) to illustrate the effect of backorder cost. Under the high backorder cost ratio with the holding cost, we see that the optimality condition changes from Case 2 to 1 in some instances where it is optimal to increase the workload to yield the required output. This scenario is evident in states shaded in blue in Figure A.3(b) in comparison with the yellow shaded region in Figure A.2(b).

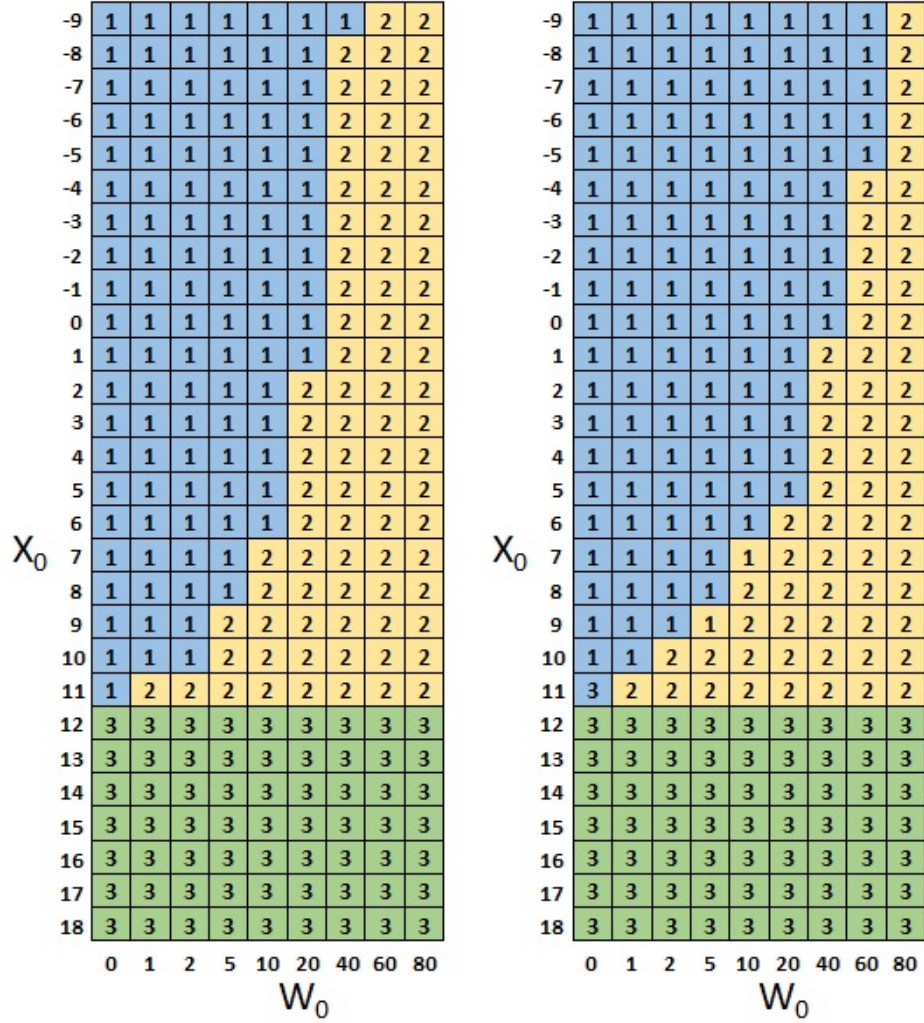


Figure A.4: For demand  $\sim \mathcal{N}(9, 2)$  (a) For  $\alpha=0.5$  (b) For  $\alpha=0.25$

We observe a similar trend when the mean demand increases from 7 to 9 when  $b = 30$ . Figures A.4(a) and (b) represent the optimal solution for states  $X_0 \in \{-9, \dots, 18\}$  and  $W_0 \in \{0, 1, 2, 5, 10, 15, 20, 40, 60\}$ . Since there is no capacity constraint, the initial state and the cost inputs determine the optimal condition. For example, from Figure A.4(a), we see the optimality condition switches between Case 1 and 2 given by blue and yellow shaded region. For states  $(12, W_0)$  onwards, we see Case 3 is optimal, which determines the expected cost to be minimum when  $R^* = P^* = 0$ . Figure A.4(b) shows a similar pattern when mean demand is 7. The main difference between these models is given by the shape parameter  $\alpha$ . i.e., the higher the  $\alpha$ , the lesser the workload requirement; hence, fewer releases are necessary. However, when  $\alpha$  becomes small, the model requires  $\frac{1}{\alpha}$  times

workload requirements to produce output.

Table A.13: Results for demand  $\sim \mathcal{N}(7, 1)$

		$\alpha = 0.5$			$\alpha = 0.25$		
$X_0$	$W_0$	$R^*$	$P^*$	Cost	$R^*$	$P^*$	Cost
-7	5	21.82	13.41	56.61	0.00	1.25	79.94
-6	5	19.82	12.41	52.36	0.00	1.25	73.94
-5	5	17.82	11.41	48.11	0.00	1.25	67.94
-4	5	15.82	10.41	43.86	0.00	1.25	61.94
-3	5	13.82	9.41	39.61	0.00	1.25	55.94
-2	5	11.82	8.41	35.36	0.00	1.25	49.94
-1	5	9.82	7.41	31.11	0.00	1.25	43.94
0	5	7.82	6.41	26.86	0.00	1.25	37.94
1	5	5.82	5.41	22.61	0.00	1.25	31.94
2	5	3.82	4.41	18.36	0.00	1.25	25.94
3	5	1.82	3.41	14.11	0.00	1.25	19.94
4	5	0.00	2.50	9.87	0.00	1.25	14.04
5	5	0.00	2.45	7.02	0.00	1.25	8.76
6	5	0.00	1.45	5.27	0.00	1.25	5.32
7	5	0.00	0.45	3.52	0.00	0.45	3.52
8	5	0.00	0.00	2.07	0.00	0.00	2.07
9	5	0.00	0.00	1.90	0.00	0.00	1.90
10	5	0.00	0.00	2.15	0.00	0.00	2.15

Table A.13 summarizes the optimal solution and the expected cost for mean demand 7 and standard deviation 1. We report the cost for  $(X_0, 5)$ , discuss the impact of the planned lead time, and compare it with the optimal conditions. For  $\alpha = 0.5$ , we get a steeper CF, and for  $\alpha = 0.25$ , we get a flatter CF. That is, when  $\alpha = 0.25$ , the expected production requires a high workload compared to a larger  $\alpha$ . Only a fraction of the workload is converted to FG, and the system requires additional releases to maintain the cost differential between underage cost and cost of production. We get a marginally higher cost when  $\alpha = 0.25$  compared to  $\alpha = 0.5$ . We observe all three cases as optimal policies when  $\alpha = 0.5$ . However, we observe only Case 2 over Case 3 when  $\alpha = 0.25$ . We notice that the expected costs due to initial states differ by the underage cost ( $b = 6$ ) when Case 2 is optimal. This difference starts decreasing when the policy is transformed to Case 3. On the other hand, for states  $(8, 5)$  and above, we find the optimal policy is

to do-nothing in both the values of  $\alpha$ .

By comparing the optimal conditions between Missbauer and Graves CF, we observe interesting patterns regarding the expected cost and the optimal conditions. For illustration, we consider E3 cost inputs with  $b = 30$  comparing the Missbauer CF with shape parameter  $K_2$  and Graves shape parameter  $\alpha$  with the UCPI model. The results are shown in Figures A.5 and A.6, respectively.

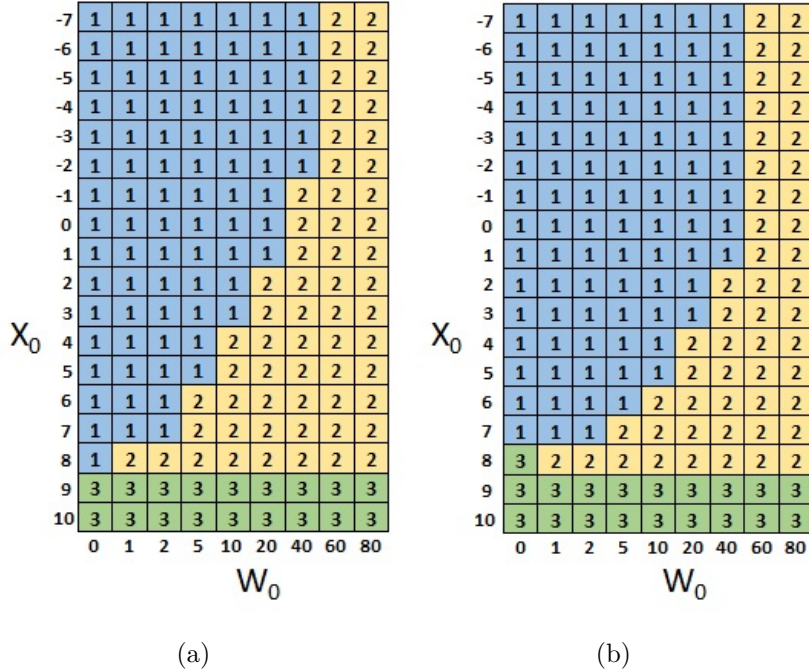


Figure A.5:  $\mathcal{N}(7, 1)$ : Missbauer CF (a) For  $K_2=10$  (b) For  $K_2=30$

To compare the optimal conditions, we take the uncapacitated production inventory model without congestion effect and discuss the change in the cases given by blue, yellow, and green shaded regions. In the Missbauer CF, the production capacity restricts the expected production; hence the additional cost is incurred for releases to satisfy external demand. However, with Graves CF, the excess production quantity yields a trade-off between backorder and workload costs. Hence, the system requires additional releases if the WIP does not support the required workload given by  $\alpha$ . Thus, the shape of the CF plays an essential role in the optimal solution characterization. Secondly, the optimal policy parameters corresponding to the initial states depend on the cost structure and the shape of the CF. The marginal capacity yields a reduction in the objective function by the dual price of the CF capacity restriction at the optimum. Therefore, production takes place using a Missbauer CF before the slope  $(\frac{\partial f}{\partial \Lambda})$  reaches the curvature near the

maximum production capacity (when  $\frac{\partial f}{\partial \Lambda} \ll 1$ ) of the CF beyond which the production system requires a large amount of workload to produce marginal production quantity. Figures A.5 confirm the impact of marginal production per workload increase. Similarly, when the slope of the Graves CF decreases, a substantial amount of workload is necessary to yield output.

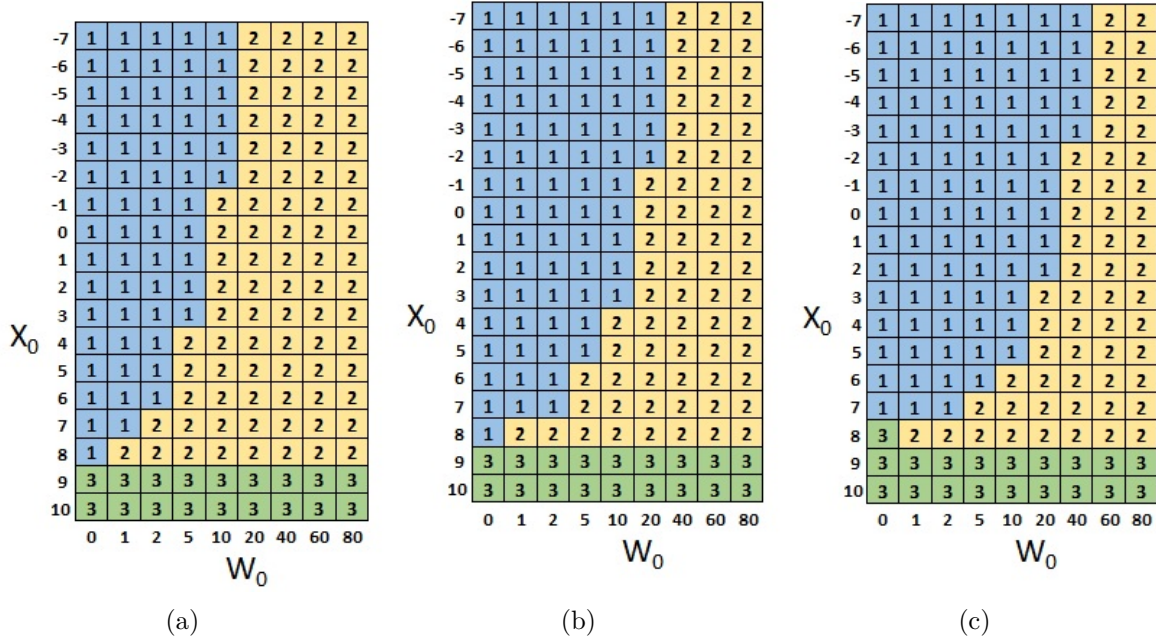
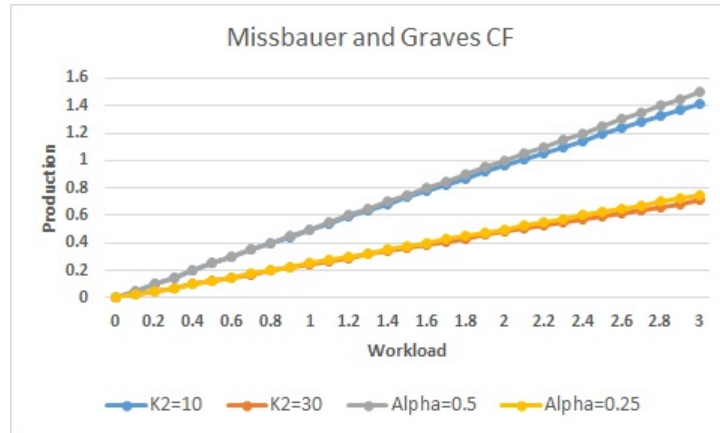


Figure A.6:  $\mathcal{N}(7, 1)$  (a) UCPI (b) Graves CF: For  $\alpha=0.5$  (c) Graves CF: For  $\alpha=0.25$

Figure A.6 illustrates the comparison between the UCPI with the Graves CF by varying  $\alpha$ . We visualize an interesting perspective across all the models for  $(8, 0)$ . The UCPI model requires a fraction of the production quantity (0.23); hence,  $R^* = 0.23$  is necessary to process the workload. Regarding slope, we conclude that the UCPI model uses  $P^* = R^*$  when  $W_0 = 0$  and follows Case 1 with the expected cost of 2.66. In the case of Graves models, when  $\alpha = 0.5$ , we see that the model follows Case 1 with  $R^* = 0.07$ , to produce  $P^* = 0.04$ , which is close to zero. However, when  $\alpha = 0.25$ , we see that the model follows a "do nothing" policy that results in the expected cost of 2.82 same as the expected cost of the Graves model when  $\alpha = 0.5$ . On the other hand, when we compare the capacitated Missbauer CF using  $K_2 = 10$ , the model follows Case 1 with  $R^* = 0.07$  to produce  $P^* = 0.03$  which is close to zero with the objective function of 2.82. However, when  $K_2 = 30$ , the model follows Case 3 like the Graves model objective function. Using  $W_0 = 0$ , when the shape parameter of Graves CF approaches 1, Graves CF-based model simplifies to the uncapacitated PI model. On the other hand, we compare the Missbauer

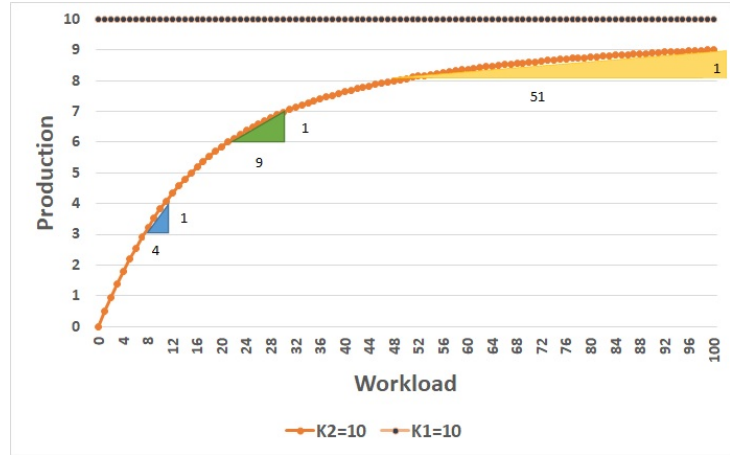
CF and Graves CF near the origin. When the production occurs near the CF's origin where  $\frac{\partial f}{\partial \lambda}$  is close to 1, the Missbauer CF-based model will become a Graves CF-based model; thus, the Missbauer CF-based model converges to the UCPI model. Figure A.7 illustrates the expected output for a given workload from the origin where the slopes of Missbauer and Graves CF are relatively close when  $\alpha = 0.5$  and  $K_2 = 10$  using Graves and Missbauer CF. Similarly, near the origin, the expected output is similar for Missbauer CF with  $K_2 = 30$  and Graves CF when  $\alpha = 0.25$ , respectively.



(a)

Figure A.7: A graphical illustration of the steep and flat Missbauer and Graves CF showing the expected output near the origin is similar that is determined by the shape parameter of the CFs

Based on the experimental evidence, we conclude that the shape parameter determines how much workload is necessary to produce the required output, given appropriate cost inputs without loss of generality to avoid a trivial "do nothing" policy.



(a)

Figure A.8: Marginal production: Blue color represents slope near origin requires four units of workload, Green color represents slope near the curvature requiring nine units of workload, and Yellow color represents slope near the horizontal segment requires 51 units of workload

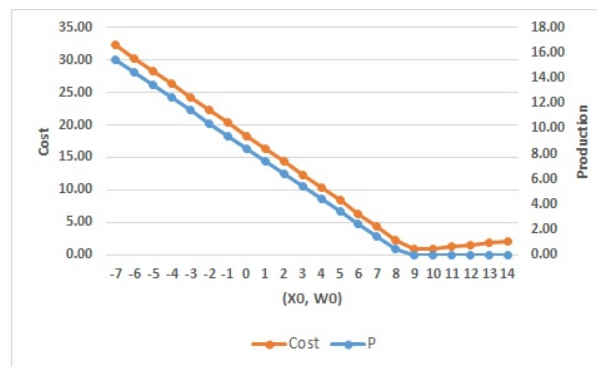
Figure A.8 represents a Missbauer CF illustrating the marginal increase in production concerning workload. A marginal production of one unit takes less workload near the origin, whereas a significant amount of workload is expected near the horizontal segment. We illustrate three regions on the CF to describe the relationship between expected production and workload requirements. As shown in blue, one production unit (3 to 4 units) requires four workload units near the origin. In contrast, 51 units are required when the curve reaches the horizontal segment where the high production shift is from 8 to 9, as shown in orange. The green color region shows moderate production, requiring nine workload units to produce from 6 to 7 units. Lastly, the shape of the CF determined by  $K_2$  in the case of Missbauer CF and  $\alpha$  in the case of Graves CF plays a crucial role in determining the production quantity. Based on the experimental results, we summarize that when the shape of the CF is flat, a large amount of workload is required to produce the expected output, and this scenario is similar in both CFs.

We now study the optimal solution of the uncapacitated Graves CF-based model with the UCPI model and discuss the key characteristic features based on input parameters, CF shape, and utilization.

### Effect of cost inputs

In this section, we study the input cost parameters  $\in \{E1, E2, E3\}$  and compare the results using the UCPI and Graves CF-based models. We compare the optimal  $R, P,$

and the expected cost between the uncapacitated Graves CF-based and UCPI models. A CF-based model brings additional restrictions despite being an uncapacitated version of the Graves CF. We use E1 cost parameters in a Graves CF with  $\alpha \in \{0.5, 0.25\}$ ,  $\mathcal{N}(7, 1)$  with  $b=30$ . We summarize the following results based on the cost inputs. Using E1, similar to the previous analysis; we witness that an optimal release is a large number (close to infinity) across all models. We observe no change in the output for the initial WIP. Using E1, optimal  $R$  is trivial, and hence these results are not shown. Therefore, we show only the results of  $P^*$  and the objective function that are constant across the models irrespective of initial WIP. The following graph shows the optimal  $P$  and objective function.



(a)

Figure A.9: Optimal production and objective function using E1

In the second experiment, we use the same demand distribution with E2 cost inputs, where  $C^r > 0$  and other cost parameters are zero except  $h$  and  $b$ . We compare the results for two different initial WIP levels  $W_0 \in \{0, 80\}$  to capture the release requirement based on the CF parameter. We observe a wide range of values for the production and releases when  $W_0=0$ . Figure A.10 illustrates the  $R^*$ ,  $P^*$ , and objective function when  $W_0 = 0$ .

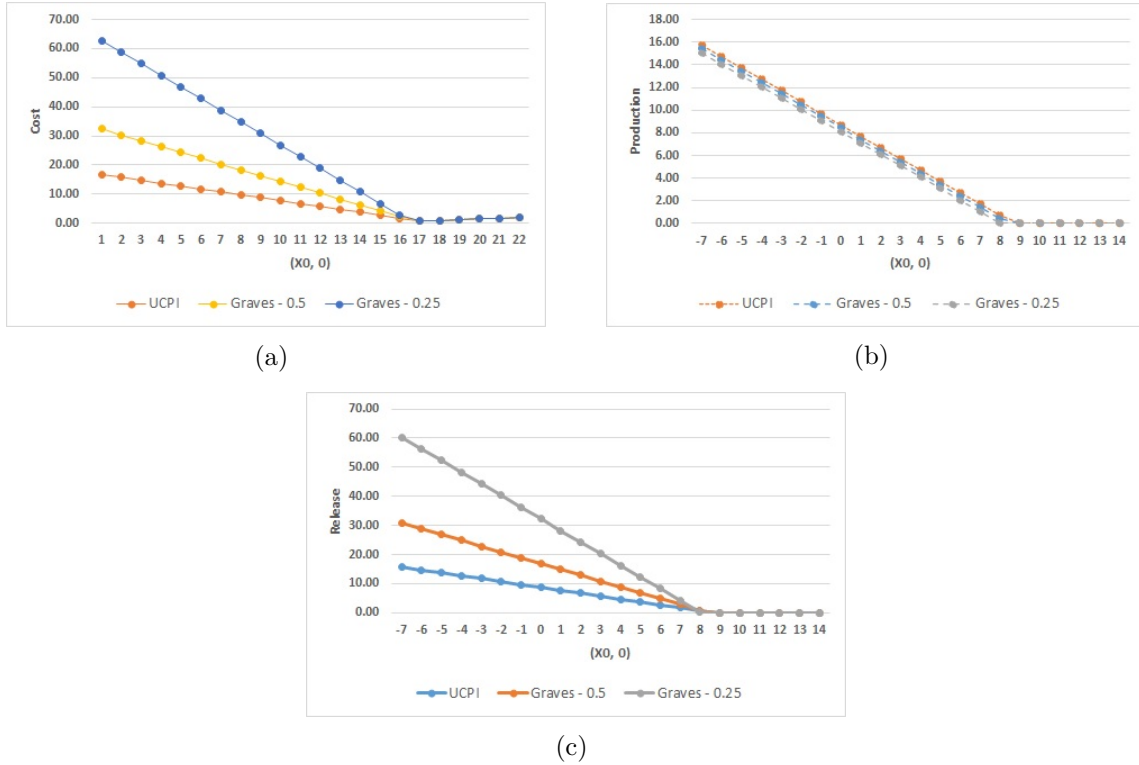


Figure A.10: Using E2 when  $W_0 = 0$  (a) Objective cost (b)  $P^*$  (c)  $R^*$

On the other hand, we visualize the optimal release are zero ( $R^* = 0$ ) across all models when  $W_0=80$ , i.e., the sufficient workload is available at the beginning of the period, and any additional release increases the expected cost unnecessarily. This is due to  $P_{UCPI}^* < \alpha(W_0)$ , then  $P^*$  across Graves CF-based models converging to the UCPI model. Figure A.11 shows the optimal  $P$  and the expected cost when  $W_0 = 80$ .

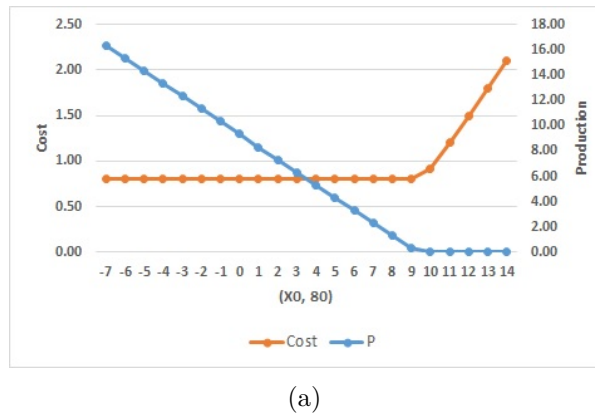


Figure A.11: Using E2 when  $W_0 = 80$ :  $P^*$  and objective cost

In the third experiment, we use E3 cost inputs to capture the effect of all cost parameters concerning the shape of the CF. When  $W_0 = 0$ , it is interesting to notice that the optimal production is close to the  $P^*$  given by the UCPI model. As expected, a large amount of release is necessary when the initial FG state falls below zero; hence, the objective function increases when the shape parameter of the Graves CF decreases.

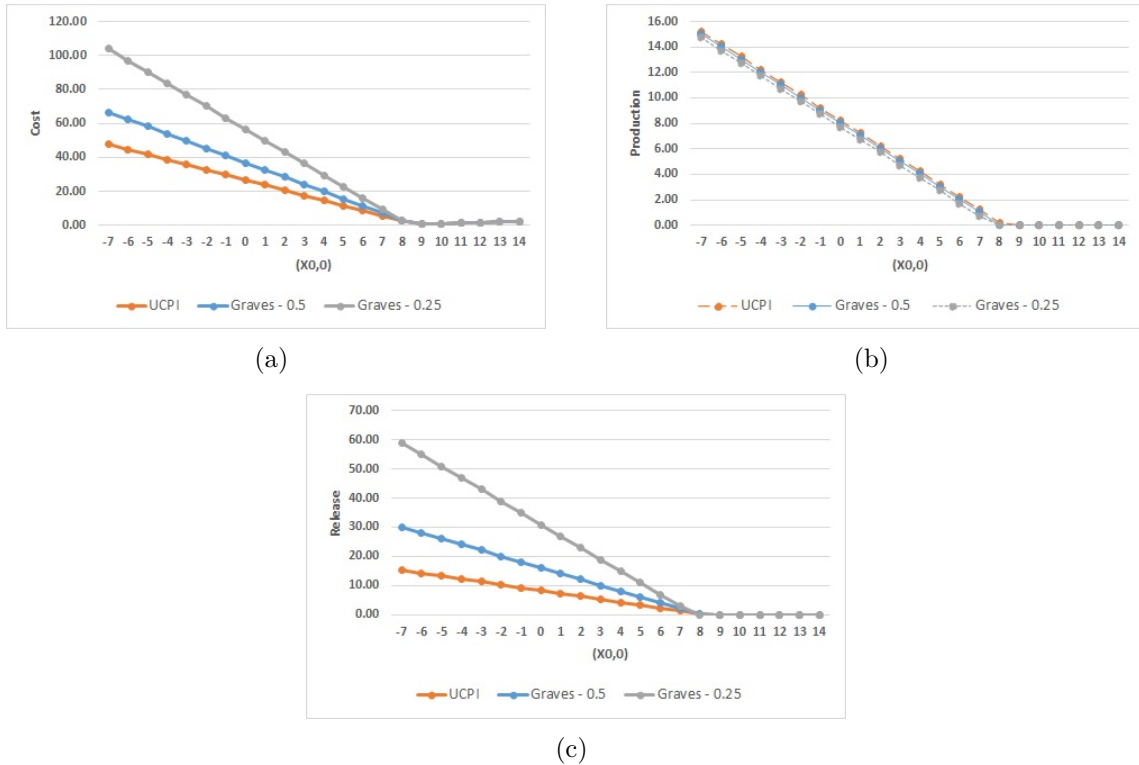
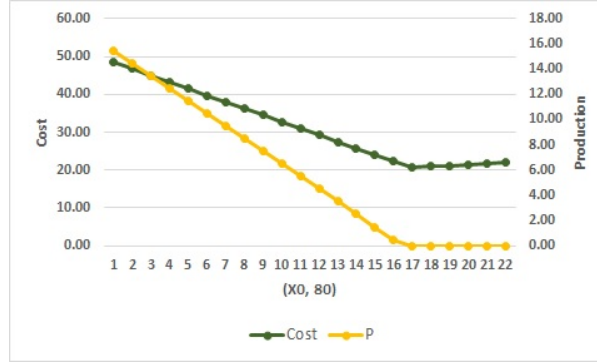


Figure A.12: Using E3 when  $W_0 = 0$  (a) Objective cost (b)  $P^*$  (c)  $R^*$

On the other hand, we witness the optimal release is zero across all models when  $W_0=80$ . Like the E2 case, we observe no change in the optimal  $P$  and the expected cost across all the models when  $W_0=80$ . The models converge to the UCPI case when sufficient WIP is present at the beginning of the period. For instance, for  $(0, 80)$ , we get  $P^* = 8.49$ , and the expected cost is 36.21, respectively.



(a)

Figure A.13: Using E3 when  $W_0 = 80$ :  $P^*$  and objective cost

### Effect of Missbauer and Graves CF

In this study, we analyze how the shape parameters of the capacitated and uncapacitated CF-based models affect the optimal solution. We compare Missbauer and Graves CF with two different shape parameter values representing a steep and a flat CF. For experimentation, we take  $\mathcal{N}(7, 2)$ ,  $b=30$  and E3 cost inputs. We study the CF-based models with two different initial WIP levels  $\in \{0, 80\}$  to illustrate the workload requirements between flat and steep CFs. We consider  $\alpha = 0.5$  and  $K_2 = 10$  to represent the steep and  $\alpha = 0.25$  and  $K_2 = 30$  to represent the flat curvature of the CF.

Figure A.14 shows the optimal cost,  $P^*$  and  $R^*$  when  $W_0 = 0$ . Under suitable cost assumptions, irrespective of the CF shape and type of CF, if  $X_0 < 0$ , additional releases are required to bring the workload to yield output. Furthermore, Missbauer CF requires significant releases in comparison with the Graves CF due to the non-decreasing concave shape of the CF with decreasing slope ( $\frac{\partial f}{\partial \Lambda}$ ) from the origin to the maximum production capacity. The shape of the CF determines the workload requirements in terms of the Graves CF. Based on the production and release requirements, the objective function increases when the CF becomes flat in the Missbauer CF. We witness an increase in the cost when the CF is flat compared to a steep one. On the other hand, when the average cycle time increases, Graves CF requires a large number of releases to produce output.

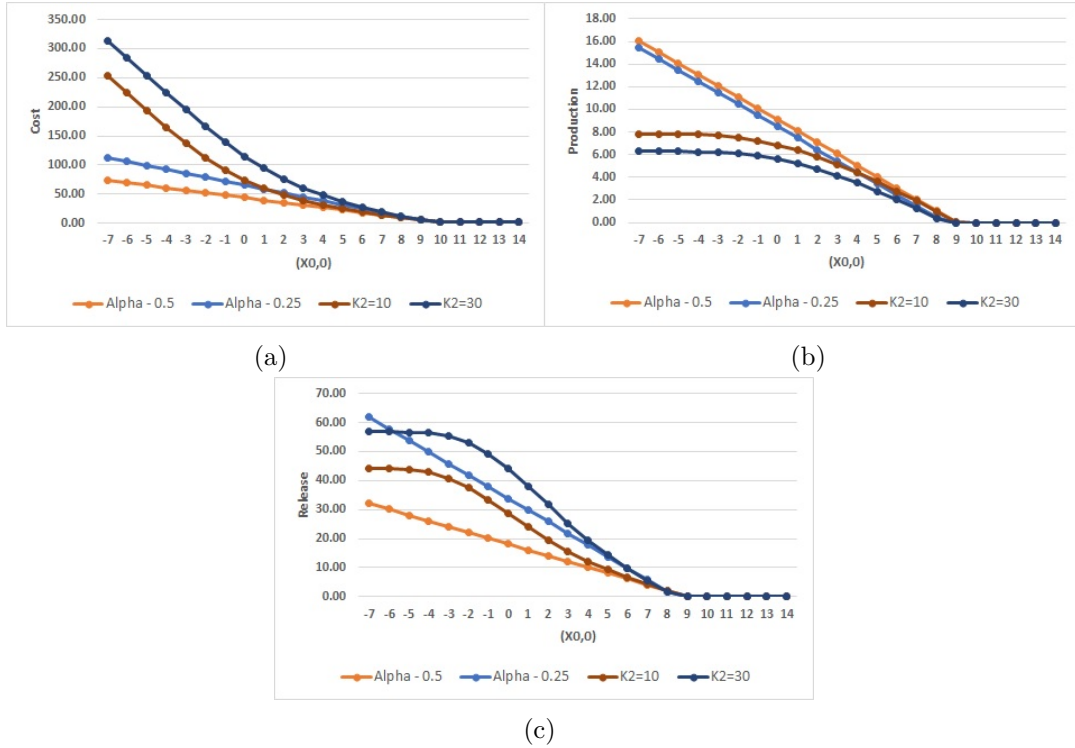


Figure A.14: When  $W_0 = 0$ , (a) Objective cost (b)  $P^*$  (c)  $R^*$

We observe a do-nothing policy when  $X_0 > 9$  in all the CFs. We observe a linear decrease in the release and production defined by the linear shape of the Graves CF. However, we find a non-linear trend in the Missbauer CF that depends on the non-decreasing CF with the appropriate shape parameter. The higher the shape parameter in a Missbauer CF, the larger the workload requirement to produce output. Thus, we get an increase in the objective function of the Missbauer CF due to a large amount of releases requirement.

Figure A.15 shows the optimal cost,  $P^*$ , and  $R^*$  when  $W_0 = 80$ . We confirm that the Graves CF-based model simplifies to become the UCPI model and the optimal  $R$ ,  $P$ , and the expected cost becomes a constant irrespective of the shape parameter ( $\alpha = 0.5, 0.25$ ). However, this behavior primarily depends on the CF shape ( $\alpha$ ). i.e., when  $\alpha(W_0) \leq P_{UCPI}^*$ , Graves CF-model will become a UCPI model, following the optimal solution given by the UCPI model. No additional release is required as the initial WIP is sufficient to produce the desired output. However, Missbauer-based CF produces less than the maximum capacity and therefore does not follow the optimal production and cannot achieve the minimum objective cost when initial WIP=80. Since Graves CF supports the

workload requirement when  $W_0 = 80$ , we get the minimum cost objective the same as the UCPI model.

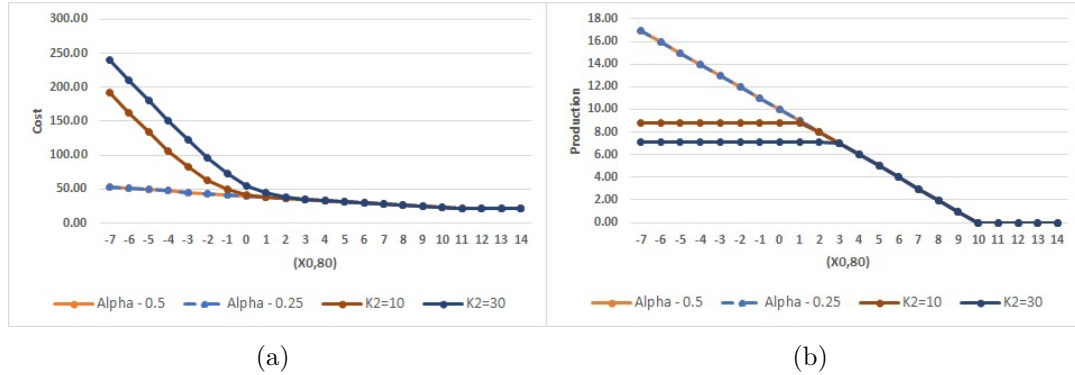


Figure A.15: When  $W_0 = 80$ , (a) Objective cost (b)  $P^*$

Despite no additional releases, the objective function using Missbauer-based CF is significantly high. This outcome is due to the large ending WIP, less production yielding more backorder cost at the end of the period. Furthermore, we observe a large increase in the objective function value in a flat Missbauer CF compared to a steep one. Refer to Figure A.15(a). All the objective function costs converge to a single number when the initial FGI state is 3 or higher when the initial WIP is 80. This result means that a sufficient workload given by  $W_0$  exists in the case of the Graves CF, and existing FGI removes additional production needed in which models converge to a single number. However, Missbauer CF requires a huge workload (supplied by  $W_0 = 80$ ) to get the expected output. Therefore, at high  $W_0$  when  $P_{UCPI}^* < P_{max}$  and  $P_{UCPI}^* \leq f(W_0)$ , then  $P_{missbauer}^* = P_{UCPI}^*$  and the cost will converge to a single critical number irrespective of the shape parameter of the CF. Without loss of generality, under certain cost assumptions, we observe similar behavior in a do-nothing policy where  $X_0 \gg \mathbb{E}[D]$ .