Stochastic Target Scheduling for Radar Resource Management

Threat Assessment and Optimal Threshold Policies

by

Erik J. Miehling

B.A.Sc, University of British Columbia, 2009

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

Master of Applied Science

in

THE FACULTY OF GRADUATE STUDIES
(Electrical and Computer Engineering)

The University Of British Columbia
(Vancouver)

September 2011

© Erik J. Miehling, 2011
Abstract

This thesis formulates a stochastic scheduler for use in adaptive resource management of a single Ground Moving Target Indicator (GMTI) radar when faced with tracking multiple, weakly maneuvering targets. The general problem involves first determining a priority allocation of the $L$ targets, then determining the optimal time to spend using the specified allocation. We present a framework for computing the threat estimate and associated priority of each target in a surveillance environment, termed the radar macro-manager. The threat level of each target is computed based on its heading and proximity relative to a set of user-specified static assets. We present a weight assignment algorithm based on the geography of the surveillance region and use eigenvector centrality to assign vulnerability weights to each asset. The error in the threat level is computed based on the error-covariance matrix of each target, provided by a Kalman filter. Both the threat level and threat error are used to compute the respective priority rank distributions. From the priority distributions of each target we specify a queue of tasks to maximize a reward function associated with processing the queue. The queue is determined with the aid of structural results from the field of optimal issuing which involves ordering the priority rank distributions with respect to the monotone likelihood ratio. In particular, we compute an optimal queue which specifies the order in which we observe individual targets. The length of each target observation is controlled by an external stochastic process, termed the radar micro-manager. The problem of determining this optimal stopping time is formulated as a sequential decision process, a type of Markov decision process. We provide conditions such that the optimal policy is characterized by a monotone threshold policy on the partially ordered set of positive definite error covariance matrices of each target. Given that the optimal policy
is monotone, we can efficiently approximate its form with an affine hyperplane using a hybrid random search - Simultaneous Perturbation Stochastic Approximation (SPSA) algorithm. Detailed numerical simulations evaluate the performance of both the radar macro-manager and radar micro-manager.
Preface

The research conducted over the duration of my Master’s degree concerned the field of stochastic control and optimization, primarily motivated by applications in defence. The work presented in this dissertation is based on the papers listed below.


The contributions of the dissertation’s author in the above publications is outlined as follows. Chapter 2 discusses the radar manager structure and the target model. The author’s contributions lie in analyzing the validity of modeling assumptions such as the LTI nature of the model and the inclusion of a probability of detection in the model to account for missed detections. Chapter 3 focuses on the task prioritization component of the radar manager and is largely comprised of material from the (in progress) paper, “Threat Assessment and Stochastic Target Prioritization for Radar Resource Management.” The majority of the work presented in this chapter is that of the dissertation’s author, namely, the formulation of the threat function, threat rank function, and the application of optimal issuing policies to the problem of stochastic task prioritization. Chapter 4 deals with the task scheduling component of the radar manager. The work presented in this chapter is based on the theory and ideas found in the paper, “Sequential detection with mutual
information stopping cost.” The author helped in the application of the monotone threshold policy to the GMTI radar problem. Much effort was made by the author to ensure seamless integration between the two levels of the radar manager. The numerical results in Chapter 5 were also solely the work of the dissertation author. All simulations were coded in Matlab with significant effort made to ensure that the simulation scenarios were relevant to real world surveillance situations.
# Table of Contents

Abstract . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . ii

Preface . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . iv

Table of Contents . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . vi

List of Tables . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . ix

List of Figures . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . x

Acknowledgments . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . xii

Dedication . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . xiii

1 Introduction . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1
   1.1 Background and Motivation . . . . . . . . . . . . . . . . . . . . . 2
       1.1.1 Why Dynamic Programming? . . . . . . . . . . . . . . . . . 7
       1.1.2 Motivation . . . . . . . . . . . . . . . . . . . . . . . . . . . . 8
   1.2 Contribution and Main Results . . . . . . . . . . . . . . . . . . . 9
       1.2.1 Main Results . . . . . . . . . . . . . . . . . . . . . . . . . . . 10
   1.3 Thesis Structure . . . . . . . . . . . . . . . . . . . . . . . . . . . . 12

2 Radar Manager and Target Dynamics . . . . . . . . . . . . . . . . . 13
   2.1 Radar Manager Structure . . . . . . . . . . . . . . . . . . . . . . 13
   2.2 Target Model . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 17
   2.3 Bayesian Tracker . . . . . . . . . . . . . . . . . . . . . . . . . . . 20
## 3 Radar Macro-manager

- 3.1 Radar Macro-manager Structure  
- 3.2 Macro-manager Scheduler
  - 3.2.1 Optimal Issuing Policy
  - 3.2.2 Target Prioritization
- 3.3 Threat Function
  - 3.3.1 Threat Estimate
  - 3.3.2 Asset Weight Assignment
  - 3.3.3 Threat Estimate Uncertainty
- 3.4 Threat Rank Function
  - 3.4.1 Distribution Assignment

## 4 Radar Micro-manager

- 4.1 Radar Micro-manager Structure
- 4.2 Optimal Stopping as a Sequential Decision Process
  - 4.2.1 Decision Epochs
  - 4.2.2 Target States and State Update
  - 4.2.3 Action Space
  - 4.2.4 Operating and Stopping Costs
  - 4.2.5 Objective: Optimal Stopping Time
- 4.3 Monotone Threshold Policy
- 4.4 Parameterized Policy
  - 4.4.1 Parameterizations
  - 4.4.2 Policy Estimation via Stochastic Approximation

## 5 Numerical Results

- 5.1 Macro-manager Numerical Results
- 5.2 Micro-manager Numerical Results
  - 5.2.1 Target Fly-by
  - 5.2.2 Persistent Surveillance

## 6 Conclusions and Future Work

- 6.1 Summary of Work Accomplished
- 6.2 Future Work
List of Tables

Table A.1 Rate of change of Jacobian matrix for various running times. . 107
Table A.2 Ratio of second-order term to first-order term of Taylor series expansion, $\zeta = 0.1$. . . . . . . . . . . . . . . . . . . . . . . 107
Table A.3 Ratio of second-order term to first-order term of Taylor series expansion, $\zeta = 0.8$. . . . . . . . . . . . . . . . . . . . . . . 108
## List of Figures

| Figure 1.1 | The categorization for radar resource management algorithms. | 4 |
| Figure 2.1 | A high-level schematic of the proposed radar manager. | 14 |
| Figure 2.2 | The two time scale set-up of the radar manager corresponding to the target priority queue. | 15 |
| Figure 2.3 | The three time scales of the radar manager corresponding to the target priority allocation vector. | 16 |
| Figure 2.4 | Typical fly-by scenario of a GMTI radar. | 17 |
| Figure 3.1 | A schematic of the radar macro-manager. | 25 |
| Figure 3.2 | Utility function, $d(x)$, representing the valuation of an observation at a given value of accumulated threat reward. | 29 |
| Figure 3.3 | Multi-target, multi-asset scenario. | 34 |
| Figure 3.4 | Radial velocity component and range vector used to determine threat of target to asset. | 34 |
| Figure 3.5 | A typical target-asset scenario. | 37 |
| Figure 3.6 | Threat level estimates (both first and second order) compared to the true threat value for $L = 4$ targets. | 40 |
| Figure 3.7 | Typical road network with asset locations. | 41 |
| Figure 3.8 | Road network represented as a graph. | 41 |
| Figure 3.9 | True threat error covariance compared to first-order approximation of threat error. | 46 |
| Figure 3.10 | True threat error covariance compared to second-order approximation of threat error. | 47 |
| Figure 3.11 | Form of target rank assignment functions. | 50 |
Acknowledgments

The research conducted over the course of my Master’s degree would not have been possible without the guidance and support of several individuals. First of all, my utmost gratitude is directed towards my Master’s supervisor, Professor Vikram Krishnamurthy. Your knowledge helped to guide my research along novel and interesting paths, while your patience provided me with the freedom to think through and solve interesting research problems. I would also like to thank the Department of National Defence (DRDC-Ottawa) for generously funding the research during my Master’s degree. I am honored to have had the opportunity to have worked directly with multiple defence scientists and engineers throughout my Master’s degree. In particular, Dr. Bhashyam Balaji and Dr. Jack Ding at Defence Research and Development Canada (DRDC) - Ottawa and Dr. Martie Goulding at MacDonald, Dettwiler and Associates (MDA) Corporation, were invaluable in bridging the gap between the theory and application of the algorithms developed. Furthermore, I would like to thank my family for their continuous encouragement throughout the course of my degree. Your endless support has always made it possible for me to find the courage to tackle intimidating new problems throughout my life. Lastly, I would like to extend my appreciation to all members of the Statistical Signal Processing laboratory at the University of British Columbia. The exceptional knowledge of each one of you helped me immensely throughout my degree.

– Erik J. Miehling
Dedication

To my loving parents
Chapter 1

Introduction

With the advent of state-of-the-art technology comes new capabilities and an associated need for advanced methodologies in order to realize the technology’s full potential. For example, with the introduction of the internet, came the ability for computers to communicate over a network. In order to ensure this communication was efficient and reliable, a set of communication protocols were developed, termed the TCP/IP (Transmission Control Protocol and Internet Protocol). Another example, reaching further into the past, concerns the introduction of the computer and the development of the operating system. With the initial introduction of the computer in the early 1950s, tasks were only able to be processed one at a time, forcing other queued tasks to wait until the completion of the previous job. The introduction of the concept of an operating system allowed the sharing of computer resources between multiple jobs, through the use of timesharing and batch processing, resulting in a much higher rate of use.

The need for novel algorithms is especially pertinent in the field of radars and surveillance. A radar is a system which sends pulses of electromagnetic (radio) waves towards an object and deduces characteristics of objects in its path based on properties of the reflected signal. The original motivating factor for the technology was for the use in military applications, but today, radar systems are used in a variety of applications including the areas of astronomy, air traffic control, meteorology, and surveillance, among others. Classically, in applications of surveillance, systems had multiple, mechanically steered radars, with each radar devoted
to a particular function (tracking, search, weapon assignment). However, with the development of Electronically Steered Phased Array Radars (ESPARs), it became possible for one radar to perform many functions, eliminating the need for multiple, dedicated radars. This was made possible by the ability of ESPAR systems to change radar parameters (beam position, dwell time, waveform, and energy level) instantaneously. Naturally, this raises the question: How does one optimally change the radar parameters to best fit the operating mode and to achieve the best possible performance of the radar? The techniques used to control the radar parameters are classified under the term **radar resource management (RRM)**.

1.1 Background and Motivation

The field of radar resource management has been an area of increasing interest over the past few decades. The specific purpose of any RRM algorithm is to analyze the mission profile and generate appropriate tasks through the assignment of particular radar operating modes. Due to the diversity of mission profiles and the corresponding tasks, modern multi-function radars (MFRs) may be faced with a multitude of possible jobs to perform. Since the radar has a finite amount of resources (time, energy, or processing power) to expend, clearly a challenge would arise in a scenario where the number of tasks outweighs the available resources. In this scenario, tradeoffs would need to be made. It would be of great interest to determine the “best” tradeoff by specifying a method to optimally allocate the finite amount of resources between the multiple tasks.

The method of determining an optimal tradeoff is dependent upon the optimization goal in mind. The general techniques of RRM are commonly considered to be either track maintenance, target search, or scheduling. This dissertation will focus on the last of the aforementioned methods, the issue of scheduling, as well as the associated task of determining the priority of each task. The need for determining task priorities is relevant in any scenario where there are more tasks to perform than can reasonably be completed within a limited amount of time (or with a limited amount of resources). It would then be logical to prescribe some sort of prioritization, such that the more important tasks are handled earlier than less important tasks. Before delving into the details of the formulation of models
and results of the thesis, a background of other algorithms is presented.

The complete problem of optimally allocating the ESPAR’s resources, while considering the possibility for a variety of operating modes, is a highly complex stochastic control problem. As a result, the work in the literature either make significant assumptions or focus on a subset of the full problem in order to obtain a solution. All radar scheduling techniques can be separated into two main categories: non-adaptive and adaptive [20, 99]. Non-adaptive techniques [29, 67], that is, heuristic schedulers, are based on rule-based design. The task prioritization and scheduling is typically pre-computed and static, enabling the convenience of real-time use of the scheduler, but coming at the sacrifice of unknown, potentially poor, performance. Advanced multi-function radars primarily use adaptive scheduling algorithms, thus the remaining portion of the literature review will be spent discussing various adaptive scheduling algorithms. An adaptive scheduling algorithm is one in which the prioritization and scheduling is determined dynamically to optimize system performance. The increased performance, over non-adaptive methods, comes at a higher (sometimes intractable) level of computation. Various algorithms work to alleviate the computation required to obtain an optimal solution. We categorize these algorithms based on the classification found in [20], which states that the vast majority of adaptive algorithms can be divided into four main categories: artificial intelligence, Quality of Service Resource Allocation Model (Q-RAM), resource-aided, and dynamic programming algorithms. The reader is directed to Figure 1.1 for a visual representation of the algorithm categorization.

The first main category within the field of adaptive scheduling algorithms is the well-researched area of artificial intelligence. Many methods in various subfields of artificial intelligence have been applied to the problem of both task prioritization and scheduling. First, techniques in the subfield of neural networks (models based on the interconnected network of nodes found in biological neural networks) have been used to solve both of the aforementioned problems: prioritization [46] and scheduling [39]. The discussion in [46] introduces a procedure for assigning priority values to targets based on the structure of an artificial neuron. The priority of a target is computed based on a weighted sum of five factors: range, radial velocity, acceleration, rank, and signal (friend or foe). The weights are computed via a learning method using back propagation. The associated problem of task scheduling has
also been studied with the use of neural networks. The discussion in [39] analyzes a formulation of task interleaving for track maintenance of multiple targets, taking advantage of the idle time between sending and receiving a pulse. The optimal interleaving network is based on a Hopfield network [36], a special type of neural network. Another subfield of artificial intelligence, termed an expert system, has been applied to the problems of prioritization and scheduling. An expert system is a system which includes a database of human (expert) knowledge used to resolve discrepancies in results and clarify uncertainties. One of the first applications of expert systems to the problem of radar resource allocation can be found in [94]. Here, the authors use an expert system to analyze the surveillance environment and modify the radar parameters and modes accordingly. The work of [70] has applied the theory to the selection of the transmit signal by controlling the signal’s type of modulation, carrier frequency, pulse duration, and pulse repetition interval (PRI). The last main subfield of artificial intelligence is the area of fuzzy logic. Fuzzy logic is a form of logic in which there exists more than just the binary outcomes: “true” and “false,” allowing for a truth value ranging anywhere in between. Multiple papers in the literature analyze how fuzzy logic can be applied to solve the
task prioritization problem [62–64, 95]. Characteristics of targets and tasks are inherently vague, thus descriptions such as friendly or dangerous or high-threat can be represented mathematically with the aid of fuzzy logic, providing flexibility in priority assignment and scheduling requirements [95].

Another category of algorithms which aim to allocate resources according to a measure of Quality of Service (QoS), termed Q-RAM algorithms, solve both the task prioritization and task scheduling problems simultaneously. The main idea behind Q-RAM techniques is, given a finite set of tasks, to assign an operating setpoint reflecting how many resources are allocated to each task such that the system utility is maximized [20]. The structure of Q-RAM algorithms consist of three main components [27]: 1) Q-RAM block, solves a convex optimization problem to maximize utility by considering many system parameters; 2) schedulability envelope, defines a region of schedulability as a convex constraint for the optimization problem; and 3) template-based scheduler, generates a dwell schedule based on parameters provided by the Q-RAM block. The scheduler in turn provides information on the scheduling of tasks to the Q-RAM block such that the scheduling constraint can be modified if necessary. The prominent work of [74] helped to provide the core foundation for the application of QoS management to resource allocation. Q-RAM can be treated as a nonlinear or integer programming problem, solved offline, resulting in the application to be restricted to static environments [35]. Classically, this prevented Q-RAM algorithms from being applied to the highly-dynamic problem of radar resource management; however, recent papers have adapted Q-RAM techniques such that they are more applicable [28, 34, 84]. In particular, [84] formulated a template-based dwell scheduler with timing and energy constraints. Tasks at run-time were then fit into the appropriate pre-computed templates. Other methods to improve optimization time [28, 34] have helped to make the radar application of Q-RAM more practical and relevant.

The next main category of radar resource management algorithms are termed resource-aided algorithms. The purpose of these algorithms is not to provide a specific task prioritization or schedule, but instead to modify the lower-level behaviour and parameters of the radar to obtain improved functionality. The two main approaches in this class of algorithms modify either, 1) The radar waveform, or 2) The update rate of the Bayesian tracker. The first category typically termed
*dynamic waveform algorithms* aim to modify the waveform signal to reduce the demand for resources (time, energy, or processing power). The precise choice of the waveform depends on the operating mode in question. The work of Kershaw and Evans discuss optimal waveform selection for target tracking purposes and probabilistic data association [43, 44]. Their results rely on the assumption of an optimal receiver, such that the one-step ahead waveform optimization problem can be solved to minimize the mean-squared tracking error at the next time step. Other work studies the waveform selection problem for the application of target classification [87]. The choice of waveform minimizes the decision time by essentially maximizing the information in the returned radar signal. Additional papers in the literature approach the waveform selection problem for target detection [52], by minimizing the time taken to detect new targets, and target tracking [53], by minimizing the tracking error of each target. The second category of resource-aided algorithms, termed *adaptive update rate* algorithms control the sampling rate of the tracker. Traditional trackers use uniform update rates for measurements; however, it is possible to use an adaptive update rate and achieve improved radar performance. The work of Keuk and Blackman [93] deals with optimizing radar parameters (SNR, track sharpness, detection threshold) with respect to radar load. They also discuss a calculation of the target revisit time, which is based upon the estimated lack of information regarding the respective target. An alternate adaptive update rate algorithm [85], is based upon an IMM model, the purpose of which is to estimate and predict the target states as well as estimate the process noise. The update interval is computed from the process noise estimation to reduce the number of track updates per unit time. Additional work [81] builds upon previous adaptive update rate algorithms by considering scenarios with high rates of missed detections and thus improving the performance of the tracker in the presence of clutter.

The final method of radar resource management discussed, and a method used in this dissertation, is the area of (stochastic) dynamic programming. Dynamic programming is a tool that allows optimization problems to be broken down such that decisions can be made in stages [10]. Current decisions are made to not only achieve a low present cost, but also low expected costs for all future decision epochs. Stochastic dynamic programming, a generalization of dynamic program-
ming which no longer assumes that the state transitions are deterministic, has been well researched in the area of radar resource management for the problem of both task prioritization and task scheduling. The work of Strömberg and Grahn [89] present algorithms for search and track functions based on dynamic programming; however, their solution method relies on a brute force optimization (computational requirements grow as the square of the number of tasks to be scheduled), making the algorithm intractable for large (realistic) surveillance applications. This computational burden of dynamic programming approaches is typically termed the curse of dimensionality. However, work has been done to drastically improve the computational requirements. In particular, the application of the theory of multi-armed bandits to sensor resource management [50, 97] makes stochastic dynamic programming a more attractive, practical approach. Specifically, these papers discuss how to compute the Gittins index, which allows the general multi-armed bandit problem to be separated into independent optimization problems, allowing the formulations of optimal and suboptimal beam scheduling algorithms. The aforementioned formulations assume that the states of the “un-played arms” stay constant; however, more recent work has offered a generalization of the multi-armed bandit problem to allow for the states of the un-played arms to vary over time. This is termed the restless bandit problem and has been applied to the problem of minimization tracking error [83] in a radar surveillance scenario.

1.1.1 Why Dynamic Programming?

As will be discussed later, the component of the radar manager in this dissertation which deals with the task scheduling problem is built upon the theory of stochastic dynamic programming. In order to provide justification for this choice, the disadvantages and shortcomings of other radar resource management algorithms are presented. First, a common theme in the artificial intelligence algorithms is the requirement for a form of a knowledge base. This necessitates learning methods, in the case of neural networks, or generation of fuzzy rules, in the case of fuzzy logic. The creation of these knowledge bases is typically slow and computationally expensive, making these methods largely impractical and inefficient for use in radar resource management [20]. The next class of algorithms that were
discussed were Q-RAM methods. Since these methods rely on assigning an operating setpoint, each setpoint typically needs to be evaluated before determining the optimal choice. For large or dynamic systems, this is not tractable [27]. Also, analysis has been carried out on the performance of Q-RAM techniques and it was shown that the resulting performance is far from optimal [37]. As stated in Section 1.1, resource-aided algorithms do not offer a solution to the task prioritization or scheduling problems, rather, they provide improved performance in systems already employing radar resource management techniques. As a result, a discussion of their performance drawbacks would not help to provide justification for our choice of a resource management algorithm. The final resource management algorithm that was discussed was the area of dynamic programming. As discussed in the previous section, dynamic programming algorithms typically suffer from the curse of dimensionality, making their application to the dynamic problem of radar resource management impractical. However, the approach offered in this dissertation builds upon structural results discussed in earlier work [49, 99] and applies the theory to resource management in GMTI systems. The structural results enable the dynamic programming problem to be efficiently solved and effectively applied to realistic radar applications.

1.1.2 Motivation

The motivation for the work presented in this dissertation is to offer a new, novel approach to radar resource management. This approach aims to offer a complete radar resource manager (in combination with the results of [51]). It consists of two main functions, task prioritization and task scheduling, and is based on structural results in the fields of optimal issuing policies and dynamic programming, respectively. The task prioritization component, termed the macro-manager, takes a semantic approach to prioritization, by ranking targets based on the operator’s incentive to observe them. The reason for this approach is that it is typically quite difficult to exactly quantify the reward for observing a particular target before an observation has been made. For example, consider an image of a scenario, it is not clear how to assign a number value to the amount of information obtained from
The other component of the proposed radar resource management algorithm, termed the micro-manager, is responsible for the task scheduling component. Based on the structural results of [49], we extend the theory to the application of observation scheduling in a GMTI scenario. By presenting the algorithms in terms of structural results, we avoid the typical drawbacks of the resource management techniques, allowing our algorithms to be applied in realistic systems.

The goal for developing radar resource management algorithms is to improve one’s ability to make conclusions on the intent of targets based on observation of their states. However, the reader should keep in mind that the algorithms presented in this dissertation are intended to aid the human operator. One of the goals of the scheduler, from a design perspective, is to compute the threat of multiple targets more quickly and more accurately than a human operator would otherwise be able to do so without the aid of this system. For this reason, the scheduler proposed is thought to be more of an addition, rather than a replacement, to the human element of target surveillance.

1.2 Contribution and Main Results

In this thesis, we consider a general scenario where a single electronically steered GMTI radar is tasked with tracking multiple targets. Under these conditions, one would be interested in determining a management scheme for optimally tracking all targets with a certain degree of accuracy. We propose a dual time-scale method of radar resource management which performs task prioritization and task scheduling simultaneously. The component of the radar manager responsible for task prioritization, operating on the slow time-scale, is termed the radar macro-manager. In particular, the macro-manager specifies one of two prioritizations: 1) a priority allocation vector, or 2) a priority queue of target observations, although the main focus of this dissertation is on the latter of the two prioritization choices. The micro-manager operates on the fast time-scale and is responsible for determining the length of the scheduling interval for a given task prioritization. This is achieved

\footnote{For readers who are interested in the conflict between semantic importance and resource utilization, see [34] for a good discussion.}
by using structural results in the field of dynamic programming and Markov decision processes. There is an inherent tradeoff in the micro-management problem. Since it is assumed that our radar can only observe one target at a given time, by observing one target and improving the accuracy of its measurements, we are simultaneously predicting the states of all other targets, resulting in poorer measurements over time. The goal is to minimize the uncertainty of the current target through a function of not only its own level of uncertainty, but also the uncertainty of all other targets in the system. The resulting dual time-scale radar manager performs two objectives, prioritization and scheduling, concurrently.

1.2.1 Main Results

The main results of this dissertation are summarized in the following points,

- **Full formulation of dual time-scale radar manager:** The proposed scheduler offers a full formulation of a dual-time scale system for radar resource management. The macro-manager, responsible for determining the task prioritization, operates on the slower of the two time scales. When the prioritized list has been computed (based on target states estimated via a Kalman filter) it is then passed to the micro-manager. The micro-manager operates on the fast time scale and is in charge of determining the optimal stopping time associated with the prescribed task prioritization.

- **Target Prioritization Queue:** The work here builds upon the ideas developed in [49, 51] by presenting the addition of a complete formulation of the radar macro-manager. We define the prioritized task list, computed using a structural result from the field of optimal issuing policies [78], as a prioritized queue of target observations. The queue is computed from a measurement of threat of each target based on the following three factors:
  1) Target proximity and heading relative to sensitive assets; The threat for each target is computed based on the target state (position and velocity) relative to every asset in the surveillance region. We build upon the formulation found in [42] by allowing for targets with non-constant velocity and also consider the range of the target to the asset in the scenario where the velocity is directed away from the asset (unlike in [66]).
target-asset pair, we compute a weighted sum based on the vulnerability to compute the threat level of each target.

2) Measure of vulnerability of each asset; We use a method from the field of link analysis [21] termed eigenvector centrality. Given a road network and known asset locations, we construct a graph and the associated adjacency matrix based on a measure of each road’s accessibility. From this, the dominant eigenvector is computed, representing the centrality of each node. Knowing which of the nodes correspond to assets allows us to determine the relative vulnerability of each asset from the dominant eigenvector. The total threat of each target to all assets is then computed as the weighted (vulnerability) sum of each target-asset threat.

3) Threat estimation error of each target; Out of the papers in the literature dealing with threat assessment of a target, almost all do not consider the both the error in the target state estimation as well as the target estimate as part of the threat computation. In this dissertation, when computing the priority of a target’s observation, we consider both the threat estimate and its associated error, reflecting the fact that, given two targets with an equivalent threat estimate, we should be more concerned about a target with a high uncertainty, than that of a low uncertainty. The threat estimate error is computed based on a second-order Taylor expansion of the threat model, using the error covariance matrix returned from the Kalman filter tracker.

• **Optimal threshold policies in GMTI systems:** The results of [49] discussing conditions for the existence of an optimal monotone threshold policy are extended to the application of target tracking using a GMTI radar. In GMTI systems, the radar is mounted on a moving aircraft and measurements are taken in terms of range, range rate, and azimuth, which are all nonlinear in the Cartesian target states (position and velocity). As a result, we linearize the measurement model and show that it is nearly linear and time-invariant for typical operating conditions, thus warranting the use of a Kalman filter. We also model the fact that the radar will inherently miss detections by modifying the Kalman filter updates accordingly. Missed detections are due to multiple factors [61] such as an excess of external noise, adverse target
behaviour (velocity of target drops below the minimum detectable velocity [76]), or the presence of measurement clutter. Details of the target model can be found in Section 2.2.

- **Numerical results:** We present a thorough analysis of the performance of the proposed scheduler. In order to accurately evaluate the performance of each component of the radar manager, we run numerical simulations for both the radar macro-manager and micro-manager, comparing the performance of each to other methods. Simulations are motivated by typical scenarios in surveillance, such as a fly-by scenario, where the aircraft follows a constant-velocity trajectory, and a persistent surveillance scenario where the target region is continually observed.

### 1.3 Thesis Structure

The remaining chapters of the thesis are laid out as follows. Chapter 2 will discuss the general radar manager, the observation and target model, and the legacy tracker. Chapter 3 analyzes the problem of determining a task prioritization, termed the radar macro-manager. The process of computing the prioritized task list involves formulating the threat function, computing asset vulnerabilities, and calculating the threat estimate uncertainty. Chapter 4 discusses radar micro-management problem, responsible for determining the optimal time to use the specified task prioritization provided by the macro-manager. We derive conditions to ensure the optimal policy takes a monotone (threshold) form. We state a stochastic approximation algorithm that forms an affine hyperplane approximation to the policy. Detailed numerical simulations can be found in Chapter 5. Simulations analyze the performance of both levels of the radar manager. Finally, Chapter 6 presents conclusions and potential directions for future work. Derivations of models and proofs of the main results of the thesis are located in Appendices B and C, respectively.
Chapter 2

Radar Manager and Target Dynamics

2.1 Radar Manager Structure

The radar manager is divided into two main levels of abstraction. These two layers are termed the macro-manager and the micro-manager. The macro-manager contains three elements: the threat function, the threat rank function, and the macro-manager scheduler. The macro-manager is responsible for determining a target prioritization of the $L$ targets in the system. Depending on the type of radar and general surveillance structure in question, this prioritization could correspond to one of a variety of alternatives. Two potential choices are listed below.

1. Target priority queue; queue of $L$ targets in the system; this queue specifies the order in which targets should be observed,

2. Target priority allocation vector; $\nu = [\nu^1, \nu^2, \ldots, \nu^L]$, where $\sum_{\forall l} \nu^l = 1$. The value $\nu^l \geq 0$ corresponds to the assigned priority of target $l$.

The target prioritization is determined from the Bayesian state estimates of the target states (from the Bayesian tracker contained in the radar micro-manager) as well as the location of sensitive assets in the observation environment. The details of the macro-manager are discussed in Chapter 3. The micro-manager block
contains three elements: the electronically steered multi-function radar, a Bayesian tracker, and the micro-manager scheduler itself. The micro-manager is responsible for determining the optimal stopping time associated with the current assigned target prioritization. This decision is based off of calculating the optimal stopping set using the system and model parameters. The micro-manager is discussed in Chapter 4. A high-level schematic outlining the interaction between the micro-manager and macro-manager can be seen in Figure 2.1.

![Schematic Diagram](image)

**Figure 2.1:** A high-level schematic of the proposed radar manager. The macro-manager (see Chapter 3), which determines the optimal priority allocation and specifies the desired queue (observation order) of the targets, \( l \in \{1, 2, \ldots, L\} \), operates on the slow time-scale, \( n \). The micro-manager (see Chapter 4) determines the optimal time to continue using the specified priority allocation and operates on the fast time-scale, \( k \).

It is necessary to provide some high-level details of the radar operation before proceeding to the following sections. In particular, we differentiate between the operation of the radar when a particular choice of target prioritization is made. The first choice considered for the target prioritization is the target priority queue. In
In this case, the macro-manager, which operates on the slow time-scale, $n$, specifies a queue of the $L$ targets which maximizes a defined reward. In this prioritization choice, the micro-manager receives measurements on the fast time-scale, $k$, and processes the queue sequentially, devoting all resources to the current target in the queue (filtering) while devoting no resources to the other $L - 1$ targets (predicting). The time-scale interaction for this prioritization can be seen in Figure 2.2. The length of a given iteration at the macro-manager level, $n$, is defined by the sum of the optimal tracking times that the micro-manager specifies for each target in the queue $\tau_{o1}^n, \tau_{o2}^n, \ldots, \tau_{oL}^n$, where each $o_l$ takes on a unique target index $l \in \mathcal{L}$. The length of a tracking interval, $n$, is highly dependent upon the level of uncertainty in the targets, but is typically on the order of 10 to 50 iterations, which corresponds to 1 to 5 seconds at the typical sampling time of $T = 0.1s$. It should be noted that the iterations of this time scale are not uniformly spaced, unlike that of the micro-manager time scale, $k$.

The operation of the radar manager is slightly different in the case of the target prioritization taking the form of a priority allocation vector, $\nu$. We introduce a higher resolution tracker time-scale, $t$, at which individual observations are re-

---

1 This corresponds to the case of the target priority vector specifying $\nu^\ell = 1$ for the highest priority target, $\ell$, and $\nu^l = 0$ for all of the $L - 1$ other target, $l \neq \ell$. 

---

Figure 2.2: The two time scale set-up of the radar manager corresponding to the target priority queue. Targets are observed in an order specified by the macro-manager while the $L - 1$ other targets are predicted based on their respective models.
ceived. Each iteration of $t$ is assumed to be on the order of milliseconds. The inclusion of the tracker time scale can be seen via the following argument. It is useful to consider the priority vector, $\nu$, as a vector specifying the ratio of the micro-management cycle spent looking at each target. For example, for a priority vector $\nu = [\nu^1, \nu^2] = [0.7, 0.3]$, the radar would observe target $l = 1$ for 70% of the tracking time, $\tau$, and target 2 the remainder of the time (30%).\footnote{Equivalently, we can interpret $\nu$ as the ratio the radar’s SNR devoted to each target.} Since the stopping time $\tau$ is unknown, we switch between the targets on a much faster time-scale, termed the tracker time scale, such that the tracking time ratios specified in the priority vector $\nu$ are satisfied for each time $k$. Since a decision to stop or continue with the current priority vector is made at the micro-manager time-scale, $k$, we are ensured to satisfy the prescribed priority allocation, regardless of the time we release control back to the macro-manager. The reader is instructed to refer to Figure 2.3 for a graphical representation of the various time-scales. A detailed analysis of the micro-management problem for the target priority vector case is presented in [51].

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure2.3}
\caption{The three time scales of the radar manager corresponding to the target priority allocation vector. Measurements are received at the tracker time scale.}
\end{figure}

Before delving into further details of the radar manager’s components, it is useful to provide context by discussing the details of our target model and Bayesian
tracker, as well as any fundamental assumptions and approximations.

2.2 Target Model

Given that the main application of the theory presented in this thesis is for GMTI systems, we assume that our targets are confined to a two-dimensional physical world. See Figure 2.4 for a typical tracking scenario with measurements from an aircraft-based GMTI radar.

\[
A_0 = (0, 0, 0)
\]

Figure 2.4: Typical fly-by scenario of a GMTI radar.

The quantity \(A_0\) is defined to be the center of our surveillance region, \(\Lambda_p\), and is set to be located at the point \((x, y, z) = (0, 0, 0)\). The term \(r_s\) is called the range swath. The platform is assumed to be located at \((x, y) = (\xi_x^k, \xi_y^k)\) and moving at a velocity \((\dot{x}, \dot{y}) = (\dot{\xi}_x^k, \dot{\xi}_y^k)\) with platform altitude defined to be constant at \(\xi_z^k = \sqrt{\xi_{x}^2 + \xi_{y}^2 \tan(\theta_d)}\), where \(\theta_d\) is the angle of depression, typically \(10^\circ - 15^\circ\). The quantity, \(\psi\), is termed the squint angle, with a value of \(90^\circ\) corresponding to the broadside-looking geometry [76].

The tracker assumes that all targets \(l \in \{1, 2, \ldots, L\}\) follow a model based on the discrete white noise acceleration (DWNA) model [8, 11]. The form of this
discrete-time dynamic system is as follows [51],

\[
\begin{align*}
    s_{k+1}^l &= F^l s_k^l + Gw_k^l \quad \text{(2.1)} \\
    z_k^l &= \begin{cases} 
        h(s_k^l, \xi_k) + \frac{1}{\sqrt{\nu_k^l}} v_k^l & \text{with probability } p_{d_k}^l \\
        0 & \text{with probability } 1 - p_{d_k}^l
    \end{cases} \quad \text{(2.2)}
\end{align*}
\]

As stated, the targets are assumed to be moving on a two-dimensional plane, thus the state vector contains information regarding the position, \(x_k, y_k\) and velocity, \(\dot{x}_k, \dot{y}_k\), resulting in a state vector for target \(l\) of \(s_k^l = [x_k^l, y_k^l, \dot{x}_k^l, \dot{y}_k^l]^T\). Also, \(z_k^l\) is the observation vector, and \(Gw_k^l, \frac{1}{\sqrt{\nu_k^l}} v_k^l\) are zero-mean Gaussian random vectors with covariance matrices \(Q \geq 0\) (positive definite ordering) and \(R(v^l \Delta) \geq 0\), respectively. The quantity \(\nu_k^l\) represents the priority allocation given to target \(l\). It is assumed that the measurement noise, \(v_k\), is independent from the state, \(s_k\). Also, the process noise, \(w_k\), is assumed to be independent of \(s_0, \ldots, s_k\) and \(z_0, \ldots, z_k\).

The platform state information at time \(k\) is captured in the vector \(\xi_k\). Similar to the state vector for each target, the platform state vector includes information about both the position and velocity, but now regarding the GMTI platform, thus \(\xi_k = [\xi_x, \dot{\xi}_x, \xi_y, \dot{\xi}_y]^T\), denoting the \(x\) and \(y\) positions and velocities, respectively.

The state matrix, \(F\), which models the physics of the non-maneuvering target, and the process noise gain matrix, \(G\), are defined as follows,

\[
F = \begin{bmatrix} 1 & T & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & T \\
0 & 0 & 0 & 1 \end{bmatrix}, \quad G = \begin{bmatrix} \frac{1}{2} T^2 & 0 \\
T & 0 \\
0 & \frac{1}{2} T^2 \\
0 & T \end{bmatrix} \quad \text{(2.3)}
\]

where \(T\) is the sampling interval. The process noise covariance matrix, \(Q\), is de-

\[\text{[The justification for these matrices and a derivation of the model can be found in Appendix B.1.]}\]
fined as,

\[
Q = \begin{bmatrix}
\frac{1}{4}T^4\sigma^2_x & \frac{1}{2}T^3\sigma^2_x & 0 & 0 \\
\frac{1}{2}T^3\sigma^2_x & T^2\sigma^2_x & 0 & 0 \\
0 & 0 & \frac{1}{4}T^4\sigma^2_y & \frac{1}{2}T^3\sigma^2_y \\
0 & 0 & \frac{1}{2}T^3\sigma^2_y & T^2\sigma^2_y
\end{bmatrix}
\]

Where the values of the process noise standard deviations, \(\sigma_x\) and \(\sigma_y\), control the intensity of the process noise and are dependent upon the (expected) target being tracked.\(^4\) The random movements of the targets are accounted for via an injected (white) process noise \([11]\). The measurement noise is defined as,

\[
R(\nu) = E \left[ \left( \frac{1}{\sqrt{\nu}}v_k \right) \left( \frac{1}{\sqrt{\nu}}v_k \right)^T \right]
\]

(2.4)

\[
= \frac{1}{\nu}E \left[ v_kv_k^T \right] 
\]

(2.5)

\[
= \frac{1}{\nu} \begin{bmatrix}
\sigma_r^2 & 0 & 0 \\
0 & \sigma_a^2 & 0 \\
0 & 0 & \sigma^2_r
\end{bmatrix}
\]

(2.6)

where the variances, \(\sigma_r^2\), \(\sigma_a^2\), and \(\sigma^2_r\) representing the variance in range, azimuth, and range rate, respectively, are chosen depending on the application and radar in question \([92]\). As will be discussed in Section B.3, we assume that we are receiving measurements from a radar which has a probability of detection less than one, \(p_d < 1\), accounting for the possibility of missed detections. As can be seen in Equation (2.2), the observation for target \(l\) at time \(k\), \(z^l_k\), takes on one of two values. If the detection is missed, then \(z^l_k = 0\), which can be interpreted as an observation being received, but with an infinite noise variance. On the other hand, if the measurement is successfully received, then \(z^l_k = h(s^l_k, \xi^l_k) + \frac{1}{\sqrt{\nu}}v^l_k\). The nonlinear measurement function, \(h(s^l_k, \xi^l_k)\), maps the target state and platform state to range, azimuth, and range rate (Doppler) values. It is defined as follows (with the target

\(^4\)It should be noted that in radar tracking applications, in order to fit a model to the targets, one needs to have an idea of the type of target being tracked before measurements are actually made. As stated earlier, we have assumed a non-maneuvering target model, which for the purposes of GMTI and ground moving targets, is commonly accepted.
Notice that the above range, azimuth, and range rate calculations are based on the relative positions and velocities of the target and the platform. This is equivalent to (the hypothetical situation of) treating either the target or the platform as being fixed in space, then taking measurements.

We can approximate the above non-linear measurement model, by a Linear Gaussian State Space Model (LGSSM). The validity of this approximation is achieved by considering typical target behaviour and showing that the measurement model, namely the non-linear measurement function, \( h(s_k, \xi_k) \), can be well approximated by a linear, time-invariant function. The resulting LGSSM is as follows, the details of which can be found in Appendix B.2.

\[
\begin{align*}
{s'}_{k+1} & = Fs_k + Gw_k \\
{z'}_k & = \begin{cases} 
Hps'_k + \frac{1}{\sqrt{\Delta}}v'_k & \text{w.p. } p'_d \\
0 & \text{w.p. } 1 - p'_d
\end{cases}
\end{align*}
\] (2.8) (2.9)

### 2.3 Bayesian Tracker

The prevailing feature of any control system is that it is able to be added to the existing black-box system, to modify and regulate the output to something that is desired. Thus, the radar control system proposed in this thesis must be able to seamlessly interact with the pre-existing, legacy Bayesian tracker. In common GMTI systems, state estimation performed by the Bayesian tracker is achieved via a Kalman filter. As can be seen in the Appendix (Section B.2), the system model, Equation (2.9), is approximately a linear and time-invariant, Gaussian model on the micro-management time-scale, thus a Kalman filter is warranted.

The Bayesian tracker receives measurements in polar coordinates, correspond-
ing to range, azimuth, and range rate (see Equation (2.7)), from the multifunction radar and aims to create “tracks” of each of the targets in the surveillance environment. These tracks are represented by two quantities, the filtered state estimate of each target \( l \), denoted \( \hat{s}_{l|k} \), and a measure of “how good” the state estimate is, represented by an associated error covariance matrix, \( P_{l|k} \). These two quantities are defined as follows,

\[
\hat{s}_{l|k} = \mathbb{E}\left\{ s_{l|k} \mid z_{0}, z_{1}, \ldots, z_{k} \right\} (2.10)
\]

\[
P_{l|k} = \mathbb{E}\left\{ (s_{l|k} - \hat{s}_{l|k})(s_{l|k} - \hat{s}_{l|k})^T \mid z_{0}, z_{1}, \ldots, z_{k} \right\} (2.11)
\]

which are computed via a Kalman filter using the measurements \( \{z_{1}, z_{2}, \ldots, z_{k}\} \). For notational simplicity, we will write the sequence of measurements as \( Z_{k} = \{z_{0}, z_{1}, \ldots, z_{k}\} \). Using a Kalman filter, the solutions to the above equations are, termed the measurement update equations [3], are

\[
\hat{s}_{l|k} = \hat{s}_{l|k-1} + P_{k|k-1} - P_{k|k-1} H \left( H^T P_{k|k-1} H + R \right)^{-1} \left( z_{k} - H^T \hat{s}_{l|k-1} \right) (2.12)
\]

\[
P_{l|k} = P_{l|k-1} - P_{l|k-1} H \left( H^T P_{l|k-1} H + R \right)^{-1} H^T P_{l|k-1} (2.13)
\]

where \( \hat{s}_{l|k-1} = \mathbb{E}\left\{ s_{l|k-1} \mid Z_{k-1} \right\} \) and \( P_{l|k-1} = \mathbb{E}\left\{ (s_{l|k-1} - \hat{s}_{l|k-1})(s_{l|k-1} - \hat{s}_{l|k-1})^T \mid Z_{k-1} \right\} \) are the conditional mean and error covariance, respectively. These quantities are initialized by \( \hat{s}_{0|0} = \hat{s}_{0} \) and \( P_{0|0} = P_{0} \geq 0 \). We wish to obtain representation for the quantities \( \hat{s}_{l+1|k} = \mathbb{E}\left\{ s_{l+1|k} \mid Z_{k} \right\} \) and \( P_{l+1|k} = \mathbb{E}\left\{ (s_{l+1|k} - \hat{s}_{l+1|k})(s_{l+1|k} - \hat{s}_{l+1|k})^T \mid Z_{k} \right\} \) as functions of the conditional mean, \( \hat{s}_{l|k-1} \), and covariance, \( P_{l|k-1} \). The following equations, termed the time update equations, express solutions for \( \hat{s}_{l+1|k} \) and
\[ P_{k+1|k} = F \hat{s}_{k|k} + 1 \]
\[ \hat{s}_{k+1|k} = F \hat{s}_{k|k} + 1 \]
\[ P_{k+1|k} = FP_{k|k-1}F^T + Q \]
\[ \hat{s}_{k+1|k} = \hat{s}_{k|k-1} \]
\[ P_{k+1|k} = FP_{k|k-1}F^T + Q \]

where Equation (2.17) is termed the *Riccati equation*. In the absence of any measurements, the state and covariance are updated by the following equations,

\[ R(P_{k|k}, \hat{z}_{k}) \text{ def } = FP_{k|k}F^T + Q - 1_{\text{non}}(\hat{z}_{k})FP_{k|k}H^T(HP_{k|k}H^T + R)^{-1}HP_{k|k}F^T \]
\[ L(P_{k}) \text{ def } = FP_{k|k}F^T + Q \]

The details regarding the derivation of the above equations can be found in Appendix B.3.
Chapter 3

Radar Macro-manager

The purpose of the radar macro manager is to develop an automated, systematic method for characterizing and predicting potential high threat situations and consequently be able to take appropriate preemptive measures. Given that the theme of this document is motivated by surveillance applications, these preemptive measures are completely passive and refer to how to control the radar.

Recall the two choices stated in Section 2.1 for target prioritization: 1) Target priority queue, and 2) Target priority allocation vector. The analysis in this section will be focused upon the former of the two prioritization choices, the target priority queue. Recall that this queue specifies the order in which the radar observes each of the targets. In particular, we wish to specify the optimal order in which targets should be observed as to maximize the immediate reward for performing target observations.

3.1 Radar Macro-manager Structure

The ultimate goal of radar surveillance is to determine the intent of each target in our system. Computing the threat level of each target based on its local behaviour provides some useful insight into its true intentions. The process of drawing conclusions about a target based on its interaction with its environment can be classified within the field of situational awareness. Situation awareness (occasionally referred to as situation analysis in the literature) is a topic which is concerned with
measuring spatial and temporal characteristics of the current environment and extracting relevant information to make optimal decisions and predict the future state of the environment. In other words, in the context of the problem discussed in this document, given information about the current state of each target in our system, we can calculate and predict how it will likely behave in the near future, resulting in information which the radar scheduler can use to make opportunistic scheduling decisions.

The radar macro-manager in this document consists of three main components. First, the threat function receives Bayesian estimates from the tracker from which it computes an estimate of the threat level as well as an associated uncertainty estimate. Second, the threat rank function receives the threat estimate for each target, defined as the pair \((\hat{r}_l, \sigma'_l)\), and ranks the targets according to their priority by assigning a random variable, \(X^l\), and associated probability distributions, defined by a mean and a variance. From the mean and variance values, we assign a specific distribution to each target’s rank such that it belongs to a class of likelihood ratio orderable distributions. Lastly, the macro-manager scheduler receives the \(L\) priority ranks, \(\{X^1_n, X^2_n, \ldots, X^L_n\}\), and produces a target prioritization. Specifically, we use the ranks to determine an optimally ordered queue specifying the order to observe the \(L\) targets in the system. We also specify a definition of a tracking time constraint vector, computed from the optimal queue, which modifies the behaviour of the micro-manager by imposing a maximum tracking time constraint dependent upon the priority of each target. A schematic of the macro-manager can be seen in Figure 3.1. Since the main theoretical result of the macro-manager is the application of the optimal issuing policy to the problem of determining an optimal observation queue, we will present this result first. Immediately following, we will state how the threat and priority of each target is computed based on the target information.

### 3.2 Macro-manager Scheduler

As stated earlier, the macro-manager could specify one of two prioritizations: a target priority queue, or a target priority vector. The theory presented in this section is devoted to determining this optimal priority queue, whereas the presentation of
In systems where the total value obtained from processing a queue of jobs is dependent upon the order in which the tasks are performed, it would be of great interest to come up with the order that maximizes the total value. This is precisely the role of the macro-manager scheduler; to determine the optimal order to observe each of the \( L \) targets in our environment. Since the total number of orders for \( L \) targets is \( L! \), it would be intractable to perform an exhaustive search of all orders in systems with a even a modest number of targets. Fortunately, we can make use of structural results developed in the literature to compute this optimal order.

The theory behind optimal issuing was largely motivated by problems in inventory management. The problem of optimal depletion of a finite inventory has been the topic of research for over half a century. Some of the early papers in this
field, [5, 19, 32] worked to provide conditions on the items (when the supply and demand are assumed known) such that one of either Last-In-First-Out (LIFO) or First-In-First-Out (FIFO) ordering policies was optimal. Later research focused on the issue of optimal stock depletion under the assumption of random item rankings. Derman and Klein in particular were among the first to study the problem of optimal issuing when the value of each item was considered to be stochastic. Pierskalla and Roach [69] extended the problem to deal with stochastic supply and demand for an application to perishable inventory (inventory control for a blood bank).

The theory from optimal stock depletion has made its way in many other lines of research over the years. One of the applications involves attempting to determine the optimal order such to maximize the total field-life (life-time) of a group of items. This problem was first approached assuming each item had a deterministic field-life [12], then was generalized to allow for stochastic field-lives [13, 78]. In the case of stochastic field-life of an item, optimal issuing orders were achieved by ordering the field-life distributions of each item with respect to both monotone likelihood ratio ordering [13, 78] and hazard-rate ordering [22]. The field-life of each item, in both the deterministic and stochastic cases, is assumed to be controlled by a so-called field-life survival function, which models how the field-life distributions change over time. Albright [1] worked to derive results for many different forms of the field-life survival function, considering two scheduling cases: independent and dependent. The independent case assumes that the process that defines at what time the items are used is independent on the field lives of each item and the order in which they are used. Conversely, the dependent case considers the fact that the field lives of successive items are dependent upon their locations in the queue. Even though many different forms of the field-life survival function were considered, results for the dependent case (more interesting than the independent case; however, considerably more complex) were limited to fairly simplistic survival functions. The theory of optimal issuing has also been applied to maximizing the net present value of a list of R&D projects [31], maximizing accuracy of predictive form filling [54], and minimizing the makespan (total processing time) of a queue of jobs [14, 16, 30, 71].

We first outline the results of the problem of optimal issuing of items with stochastic field-lives originally discussed in [78]. This involves introducing the
definitions and lemmas used to obtain the results. We then apply Ross’ result to
the problem of determining an optimal order for observing the \( L \) targets in our
system, and outline the main proofs in the context of our problem.

3.2.1 Optimal Issuing Policy

We start by introducing three types of stochastic orders, used to determine if one
random variable dominates another random variable. Most stochastic orders are
not complete orders, but rather partial orders, since not every pair of distributions
are orderable. It is necessary, for the purposes of the following theory, to state the
definition of three types of (partial) stochastic orders, expected value dominance,
first-order stochastic dominance, and likelihood ratio dominance.\(^1\)

\begin{definition}
Let \( X \) and \( Y \) be random variables with positive support.

1. Expected Value Dominance: The random variable \( X \) is said to dominate
the random variable \( Y \) in expectation if,

\[ \mathbb{E}\{X\} \geq \mathbb{E}\{Y\} \quad (3.1) \]

2. First-order Stochastic Dominance: The random variable \( X \) is said to be
stochastically larger than the random variable \( Y \) if,

\[ P(X > w) \geq P(Y > w) \text{ for all } w. \quad (3.2) \]

This is denoted by \( X \succeq_{st} Y \).

3. Likelihood Ratio Dominance: Let \( X \sim f(x) \) and \( Y \sim g(x) \), where “\( \sim \)” means
“distributed as.” Then \( X \) dominates \( Y \) in a likelihood ratio sense if,

\[ \frac{f(x)}{g(x)} \text{ increases in } x. \quad (3.3) \]

This is denoted by \( X \succeq_{LR} Y \).
\end{definition}

\(^1\)For the purposes of this paper, we will only need to consider random variables that are confined
to a positive domain.
The likelihood ratio order is a stronger condition than first-order stochastic dominance, thus \( X \geq_{LR} Y \Rightarrow X \geq_{st} Y \), but the converse, in general, is not true. First-order stochastic dominance also implies dominance in expectation. Thus,

\[
X \geq_{LR} Y \Rightarrow X \geq_{st} Y \Rightarrow E\{X\} \geq E\{Y\}
\]

(3.4)

The problem of maximizing the field-life of a stockpile of \( n \) items is discussed in [78](p.118) and will be restated here. Each of the \( n \) items has a ranking, represented by the random variables \( X_i, i = 1, 2, \ldots, n \). The field-life of the \( i \)-th item if released into the field at time \( t \) is \( L_i(t) = X_i d(t) \). The items are assumed to be placed into the field sequentially and replace immediately the moment the previous item expires. That is, when item \( i \) reaches the end of its field-life, item \( i + 1 \) will immediately be placed into the field. The function \( d(t) \), termed the field-life survival function, models how the rankings change over time. For the example in [78], it is assumed that \( d(t) \) is a non-negative, non-decreasing, concave function, thus the items are assumed to improve with time. The logic behind this particular choice of \( d(t) \) is clearly dependent upon the particular application. A relevant example for this form of \( d(t) \) is a stockpile of batteries, each in their own charger, where the charge of each battery increases more quickly at a low charge than when the battery is reaching a full charge. Conversely, the case of \( d(t) \) being decreasing function would fit an example where the products are worsening in time in the stockpile, such is the case in perishable inventory.\(^2\) The main result is that given a likelihood ratio ordering of the ranks, \( X_1 \leq_{LR} X_2 \leq_{LR} \cdots \leq_{LR} X_n \), and non-negative, non-decreasing, concave function \( d(t) \), the policy which issues the first item, then the second item, and so on, until the last item, will maximize the probability of a maximum total field-life. That is, the theory specifies to issue items in order of increasing likelihood ratios of their respective distributions.

The alternate case, for \( d(t) \) a non-negative, non-decreasing, convex function, implying that items are improving at an increasing rate while in the stockpile, is

\(^2\) This is clearly the more relevant case for the problem of perishable inventory; however, there is a surprising lack of literature that analyze decreasing \( d(t) \) in the stochastic case.
modified and applied in the following section to determining an optimal target prioritization. For this case, the optimal order to maximize the total expected field-life is to issue items in decreasing ranks; $n \rightarrow n-1 \rightarrow \ldots \rightarrow 1$. The proof follows from a similar argument discussed in Ross [78] but is presented in the Appendix C in the context of our problem for intuition behind the operation of the scheduler.

### 3.2.2 Target Prioritization

The optimal issuing policy provided by Ross [78] can be applied to the problem of determining an optimal order to observe the $L$ targets. First, we define the threat rank of a target as a quantity which is assigned based on the target’s rank, $X^l_n$. As can be seen later, in Section 3.4, a target, $l$, with a high level of incentive (a high level of threat) at time $n$ is assigned a high rank mean, $\mu X^l_n$. The incentive (immediate reward) of observing a particular target is analogous to the field-life of an item, as discussed in the optimal issuing problem statement in Section 3.2.1. The immediate reward of observing target $l$ at an accumulated reward value, $x$, is defined as $T^l_n = X^l_n d(x)$. The function $d(x)$, seen in Figure 3.2, represents the valuation (utility) of the immediate reward of observing the current target, based on a given level of accumulated reward, $x$. The form of $d(x)$ is positive ($d(0) > 0$), non-decreasing, convex function, indicating that we value a gain, $\bar{x} > 0$, in threat reward more at a higher accumulated reward value, $x_2$, than an equivalent gain at a lower value, $x_1$. That is,

$$d(x_2 + \bar{x}) - d(x_2) > d(x_1 + \bar{x}) - d(x_1) \quad \text{for} \quad x_2 > x_1$$

![Utility function](image)

**Figure 3.2:** Utility function, $d(x)$, representing the valuation of an observation at a given value of accumulated threat reward.
The intuition behind the shape of this function can be understood by the following argument. Notice that \( x \) is an accumulated variable, as is time, \( t \), in the example discussion in [78]. As targets in the queue are observed, the accumulated reward, \( x \), is incremented accordingly. It is clearly evident that an accumulation of a threat reward \( \bar{x} \) requires a positive amount of time \( t \). As a result, a higher accumulated reward translates into a longer time before some targets are observed. Thus, the perceived threat of these targets should be higher and as a result, a higher utility for their observation is assigned the longer we wait to make the observation, thus justifying the increasing slope of the function \( d(x) \). In summary, the goal of the macro-manager is to find an order to maximize the total accumulated reward, which is accumulated at an amount dependent upon the threat estimate and threat error of the target under observation. Thus, essentially, we are ordering our target observations to maximize the reward proportional to the incentive for observing each target.

The ordering problem is now formally presented in the context of the target scheduling problem. The set \( \mathcal{O} \) contains all \( L! \) possible permutations of the orders of target observations. We define an order as \( o = (o_1, o_2, \ldots, o_L) \in \mathcal{O} \) which corresponds to one of the permutations of \( \mathcal{O} \). Each of elements \( o_1, o_2, \ldots, o_L \) is assigned a unique target index from 1, 2, \ldots, \( L \) such that no two elements have the same target index. The optimal order, denoted \( o^* \), maximizes the expected total threat reward, denoted \( T^o_{nL} \). That is,

\[
onumber
o^* = \arg \max_{o \in \mathcal{O}} \{ \mathbb{E}\{T^o_{nL}\} \} \quad (3.6)
\]

\[
\bar{T}^o_n = \max_{o \in \mathcal{O}} \{ \mathbb{E}\{T^o_n\} \} \quad (3.7)
\]

where the reward \( T^o_{nL} \) is defined recursively as follows,

\[
T^{o_1}_n = \bar{X}^{o_1}_n d(0) \quad (3.8)
\]

\[
T^{o_2}_n = T^{o_1}_n + \bar{X}^{o_2}_n d(T^{o_1}_n) \quad (3.9)
\]

\[
\vdots
\]

\[
T^{o_L}_n = T^{o_{L-1}}_n + \bar{X}^{o_L}_n d(T^{o_{L-1}}_n) \quad (3.10)
\]
where \( \tilde{X}_n^{o_l} \) is the realization of the random threat rank variable, \( X_n^{o_l} \). Note that \( T_n^{o_l} \) itself is a random variable, and dependent upon the order in which we observe targets, making the problem of determining the optimal order non-trivial.

The optimal ordering policy for the target observations is stated formally in the following theorem.

**Theorem 3.2.2.** Suppose that there are \( L \) targets with ranks, \( X_1^n, X_2^n, \ldots, X_L^n \) which can be arranged according to the likelihood ratio order (Definition 3.2.1) as follows,

\[
X_1 \leq_{LR} X_2 \leq_{LR} \ldots \leq_{LR} X_L
\]  

(3.12)

Then for a non-negative, non-decreasing, convex utility function \( d(x) \), the policy that maximizes the probability for all \( x \) that the total reward exceeds \( x \), is to observe the targets in the following order,

\[
o^* = (L, L - 1, \ldots, 1)
\]  

(3.13)

and is termed the optimal order.

**Proof.** See Appendix C.1.4.

The above theorem states that the optimal order, defined in Equation (3.6), corresponding to the maximum threat reward, Equation (3.7), is achieved by ordering the observations according to decreasing likelihood ratios of their respective ranks, \( X_1^n, X_2^n, \ldots, X_L^n \).

We now state the details of the modeling of the threat and priority of each target, necessary for the computation of the aforementioned priority ranks and optimal observation queue.
3.3 Threat Function

The definition of the problem of threat assessment, and of threat itself, can vary widely depending on the context. A fairly general definition of threat assessment, provided by Allouche et al. [80], which seems to apply to all treatments of threat analysis in the literature, is as follows: “The analysis of the past, present and expected actions of external entities, and their consequences, to identify menacing situations and quantitatively establish the degree of their impact on the mission, the intents, the plans, the actions and the human and material assets of some valuable units to be protected, taking into account the defensive actions that could be performed to reduce, avoid or eliminate the identified menace.” The precise interpretation of threat is dependent on the particular situation in question. In [68], the author focuses on a scenario where an UAV must reach a destination by taking the path of least threat. A high threat value in this context corresponds to a scenario where the UAV is likely to be detected by a radar. Threat in [72] is defined from the perspective of vehicle tracking systems, and thus refers to a scenario where two objects may collide. The definition of threat in the context of target tracking and radar systems typically refers to the perceived risk that a target presents to the radar itself. Since in our application, the radar measurements are assumed to be of ground-moving targets performed via an aircraft-mounted GMTI radar, we do not maintain the same definition of target threat. Instead, we consider a set of sensitive assets; a collection of areas which we wish to protect from enemy targets, and compute the threat of each target in relation to these assets.

Some papers in the literature deal with the the problem of threat evaluation in the context of military applications and defence. The problem of asset protection is discussed in [18, 80] (termed threat reference point); however, both references lack an explicit mathematical formulation for determining the threat to these assets based on information on the target behaviour. Their analysis is thorough, but only in a qualitative sense; with the work of [79] providing a thorough review of threat evaluation and weapon assignment. More detailed formulations of threat functions are discussed in [42, 64, 66]. The analysis for threat evaluation in [42] is broken down into three main components: 1) Proximity: A quantity Time-to-Closest-Point-of-Approach (TPCA) is defined which represents the expected time
for the target in question to reach a particular region centered around an asset (with a lower TPCA value representing a higher threat); 2) Capability: based on estimations of the target’s weapons systems, fuel capacity, and general lethality of a target; 3) Intent: a value which can be inferred from various factors such as heading, range rate, and altitude. The discussion in [64] is slightly more general than that of [42] and includes a decision tree for target priority assessment which takes into account factors such as target trajectory (heading, range rate), hostile/friendly ID, weapon systems, and track quality. The threat function formulation discussed in [66] makes an assumption that the target poses no threat to an asset if its rate of motion is directed away from the given asset. In reality, due to a potential ability for the target to change directions in a short time frame, the proximity of the target-asset pair should still be taken into account, even if the velocity is directed away from the asset. This extension is considered in our formulation. It is the aim of the following sections to provide an explicit, mathematical link between target behaviour and fixed assets and the respective levels of threat, building upon the work already developed throughout the literature.

3.3.1 Threat Estimate

In this section, we address the factors that can be used to determine a target’s level of threat. We are provided with estimates of the targets’ states, both position and velocity, as well as knowledge of any sensitive assets in the environment. We start with determining the threat of a given target to a particular asset in our system. An asset could be a friendly target that we are trying to protect or simply just a region we wish the target to stay away from. In any case, the closer the target is to a given asset, the higher we should consider its threat. Note that assets are defined as points in $\mathbb{R}^2$ and are chosen by the system designer. Consider the following scenario, with two targets and three assets as seen in Figure 3.3.

The coordinates $(x_{p1}^p, y_{p1}^p), (x_{p2}^p, y_{p2}^p), ..., (x_{pA}^p, y_{pA}^p)$ represent the locations of our assets, where $A$ is the total number of assets. The quantity $\mathbf{r}_l = [\mathbf{x}_l, \mathbf{y}_l]$ represents the instantaneous velocity vector of target $l \in \mathcal{L} = \{1, 2, ..., L\}$ and $\mathbf{r}_{a,l}$ represents the range vector from target $l$ to asset $a$. We wish to calculate the component of a given target’s velocity in the direction of a particular asset. Consider Figure 3.4 for the
structure of a sample calculation of the threat of target $l$ to asset $a$.

Define the target-asset threat (TAT) function as,

$$\tilde{\gamma}_{a,l} = \frac{\hat{r}_{a,l}}{r_{a,l}}$$  \hspace{1cm} (3.14)

Figure 3.3: Multi-target, multi-asset scenario.

Figure 3.4: Radial velocity component and range vector used to determine threat of target to asset.
where,

\[
\hat{r}_{a,l} = \frac{[x_a^p - x_l, y_a^p - y_l] \cdot [\dot{x}_l, \dot{y}_l]}{\sqrt{(x_a^p - x_l)^2 + (y_a^p - y_l)^2}}
\]

(3.15)

\[
r_{a,l} = \sqrt{(x_a^p - x_l)^2 + (y_a^p - y_l)^2}
\]

(3.16)

The vector \(\hat{r}_{a,l}\) is the magnitude of the projection of the target’s velocity vector onto the direction vector between target \(l\) and asset \(a\). Also, the value \(r_{a,l}\) is the distance between the target and the asset. This TAT function is directly proportional to the velocity of the target with respect to the asset, that is, \(\hat{\gamma}_{a,l}\) will be positive when the target is moving towards the asset and negative when the target is moving away from the asset. Also, the function is inversely proportional to the distance between the target and asset. This makes intuitive sense, as when the target is close to the asset, it should be considered to be more of a threat.

There are some issues with the current formulation of the TAT function that should be addressed. Consider the case where we have two assets \((x_1^p, y_1^p)\) and \((x_2^p, y_2^p)\), and the target is located equidistant between the two but moving directly towards, say, the first asset. At this moment, the TAT functions values between the target and the two assets will exactly cancel out, that is,

\[
\hat{\gamma}_{a,1} = \hat{\gamma}_{a,2} = 0
\]

Thus, in the logical next step where we wish to combine the TAT functions for multiple targets and assets to compute the total threat of a target, a simple addition would result in the threat of the above target to be zero, when in reality, it is still a very real threat to the asset located at \((x_1^p, y_1^p)\). To alleviate this problem, we restrict the TAT functions to map to only positive numbers. One way of doing this is to use the exponential function as follows,

\[
\gamma_{a,l} = e^{\hat{\gamma}_{a,l}}
\]

(3.18)

The threat value is exponential in the projected velocity \(\hat{r}_{a,l}\), and thus negative quantities are mapped to a dampened positive value. Since the TAT function for
each target-asset pair is guaranteed to be a positive number, we can perform a weighted sum to determine the threat of the $i^{th}$ target to all of the assets in our system,

$$\Gamma(s^l) = \sum_{a=1}^{A} w_a \gamma_{a,l}$$

(3.19)

where $w_a > 0$, $\sum_a w_a = 1$, are the importance weights of each asset in our system. A high value of $w_a$ implies that we consider asset $a$ as an important asset. In essence, the TAT function considers both the proximity threat as well as the threat associated with the direction and velocity in relation to all assets in our surveillance region. The formal definition for the TAT function is as follows,

**Definition 3.3.1 (Target-Asset Threat Function).** The Target-Asset Threat (TAT) function, denoted $\Gamma$, maps the state vector of target $l$, $s^l \in \mathbb{R}^{4 \times 1}$ to a positive threat value, for each target, $r^l \in \mathbb{R}^+$. That is, $\Gamma: \mathbb{R}^{4 \times 1} \rightarrow \mathbb{R}^+$,

$$r^l = \Gamma(s^l) = \sum_{a=1}^{A} w_a \gamma_{a,l}$$

(3.20)

where $\gamma_{a,l}$ is defined in Equation (3.18).

For some intuition behind the calculated $\Gamma(s^l)$ value, consider the following argument. By computing this quantity, we are essentially calculating the target’s capability to be a hazard to the assets in the system. Following the argument outlined in [18], the logic here is that, in general, a target will not have an intent to attack unless it is capable of one. Thus by computing the capability we are thus gaining information on its intent and can use $\Gamma(s^l)$ as a measure of the target’s threat. A typical target-asset scenario can be seen in Figure 3.5. As can be seen from the figure, the state estimates do not provide completely accurate information of the true target tracks, that is, there is an error associated with the state estimates. As a result, we would also expect a discrepancy between the true threat and the estimated threat. This is indeed the case, as seen in Figure 3.6.
Figure 3.5: A typical target-asset scenario. True target tracks are represented by the colored (solid) tracks, whereas the dashed lines represent the state estimates. There are a total of three assets in the system, which are assumed to be fixed with coordinates (all in units of meters) \((x_1^p, y_1^p) = (220, -150), (x_2^p, y_2^p) = (-170, 230),\) and \((x_3^p, y_3^p) = (400, 20)\) with importance weights \(w_1 = 1/3, w_2 = 1/2, w_3 = 1/6,\) respectively.

The radar platform is moving along a linear, constant velocity path with states, \([\xi_x k, \dot{\xi}_x k, \xi_y k, \dot{\xi}_y k]^T = [10\text{km}, 53\text{m/s}, -30\text{km}, 85\text{m/s}]^T\) at a constant altitude of \(\xi_z = 5575.\)

Consider a set of optimal target state estimates, denoted \(\hat{s}_n^l,\) provided from our Bayesian tracker. Then one may expect the optimal mean, \(\hat{r}_n^l,\) of the threat estimate to simply be,

\[
\hat{r}_n^l = E\{r_n^l\} \approx \Gamma(\hat{s}_n^l)
\]  

(3.21)

However, the above expression is not accurate if the function, \(\Gamma,\) is highly nonlinear or the level of noise is high [60]. This arises from the fact that the expectation of a function of a random variable is, in general, not equal to the function of the expected value of a random variable. That is, \(\hat{r}_n^l = E\{r_n^l\} = E\{\Gamma(\hat{s}_n^l)\} \neq \Gamma(E\{\hat{s}_n^l\}) = \)
\( \Gamma(s_l^n) \), where \( s_l^n \) denotes the true, unknown state vector of each target.\(^3\) Thus we wish to find a better approximation of the threat estimate to the true threat value. One way of doing this is to expand \( \Gamma(s) \) in a Taylor series expansion. That is,

\[
\Gamma(s_l^n) = \Gamma(\hat{s}_l^n) + \nabla_s \Gamma(\hat{s}_l^n)(s_l^n - \hat{s}_l^n) + \frac{1}{2} \nabla^2 \Gamma(\hat{s}_l^n)(s_l^n - \hat{s}_l^n) \otimes (s_l^n - \hat{s}_l^n) + \ldots
\]

(3.22)

Where, for the sake of notation, we have omitted the notation denoting that the partial derivative Jacobian, \( \nabla_s \Gamma \in \mathbb{R}^{1 \times 4} \), and Hessian, \( \nabla^2 \Gamma \in \mathbb{R}^{1 \times 4^2} \), are dependent upon the target, \( l \), and time, \( n \), indices since they are evaluated at the state estimate, \( s_l^n = \hat{s}_l^n \). The Jacobian vector is as follows (the Hessian is omitted due to its large size),

\[
\nabla_s \Gamma = \sum_{a=1}^{A} \left( w_a e^{-(\bar{y}_a - \hat{y}_a)\bar{r}_a} \bar{x}_a \right)
\]

(3.23)

where \( \bar{x}_a = x - x^a, \bar{y}_a = y - y^a \), and \( r_a = \sqrt{\bar{x}_a^2 + \bar{y}_a^2} \). It should be noted that the Jacobian and Hessian can be computed numerically, where we would only rely on function evaluations to approximate the entire function, the essence of Taylor series expansions. However, in this problem, since we have complete knowledge of the threat function, we are able to analytically compute the Jacobian and Hessian offline as a function of the state, providing the advantage of an immediate value for a given state.

If we now only consider the first order approximation, that is, keeping the first two terms of the RHS of Equation (3.22), then take expectations of both sides, we obtain,

\[
E\{r_l^n\} = \hat{r}_l^n = E\left\{ \Gamma(s_l^n) + \nabla_s \Gamma(\hat{s}_l^n)(s_l^n - \hat{s}_l^n) \right\}
\]

(3.24)

\[
= E\{\Gamma(s_l^n)\} + \nabla_s \Gamma E\left\{ (s_l^n - \hat{s}_l^n) \right\}
\]

(3.25)

\[
= \Gamma(\hat{s}_l^n)
\]

(3.26)

Since \( (s_l^n - \hat{s}_l^n) \) is assumed to be zero-mean, then \( E\{(s_l^n - \hat{s}_l^n)\} = 0 \). We can see that

\(^3\)This should be clear for readers who are aware of Jensen’s inequality.
the above does not give us any more information than Equation (3.21). This first-order estimate can be seen in Figure 3.6 as the black-dotted line. We now consider the second order approximation, by keep the first three terms of Equation (3.22) and taking expectations,

\[ E \{ r_n \} = \hat{r}_n = E \left\{ \Gamma(\hat{s}_n) + \nabla \Gamma(\hat{s}_n) (s_n - \hat{s}_n) + \frac{1}{2} \nabla^2 \Gamma (s_n - \hat{s}_n) \otimes (s_n - \hat{s}_n) \right\} \] (3.27)

\[ = \Gamma(\hat{s}_n) + \frac{1}{2} \nabla^2 \Gamma E \left\{ (s_n - \hat{s}_n) \otimes (s_n - \hat{s}_n) \right\} \] (3.28)

\[ = \Gamma(\hat{s}_n) + \frac{1}{2} \nabla^2 \Gamma \text{vec} \left( (s_n - \hat{s}_n)(s_n - \hat{s}_n)^T \right) \] (3.29)

\[ = \Gamma(\hat{s}_n) + \frac{1}{2} \nabla^2 \Gamma \text{vec} \left( P_n^I \right) \] (3.30)

Where the steps (3.28) and (3.29) are obtained from [60]. The \text{vec()} operator stacks the columns of a matrix argument, thus the quantity \text{vec} \left( P_n^I \right) \in \mathbb{R}^{4^2 \times 1} is a vector created by stacking the columns of the covariance matrix \( P_n^I \). Equation (3.29) offers a refined estimate; the accuracy of which can be observed as the red-dotted line in Figure 3.6. The additional term, \( \frac{1}{2} \nabla^2 \Gamma \text{vec} \left( P_n^I \right) \), an approximation of the bias, is necessary in the presence of nonlinearities of \( \Gamma \) and noise in the state \( s_n \) to ensure that filter estimates are unbiased [6, 26, 55, 60].

Given the knowledge of the uncertainty in our state estimates (contained in the error covariance matrix \( P_n^I \)), and now that we have reasonable estimates of the threat level, we can better estimate the error in these estimates. But first, we analyze a small extension to problem of assigning asset weights.

### 3.3.2 Asset Weight Assignment

We now discuss a short extension of the problem of assigning weights to the assets. The weight of an asset is dependent upon a measure of its vulnerability. In typical tracking scenarios, chiefly in urban-surveillance scenarios, the motion of targets is restricted to roadways. Assuming that the assets we wish to protect are located on roadways as well, we aim to compute the weight of each asset as a representation of its accessibility from nearby roadways. The tools used to determine this vulnerability weight are obtained from the field of eigenvector centrality. Google’s PageRank algorithm is a form of eigenvector centrality, discussed in [21].
Figure 3.6: Threat level estimates compared to the true threat value for $L = 4$ targets. Starting in the top-left and moving right, plots correspond to target indices $l = 1, 2, 3, 4$. The solid (colored) line represents the true threat value, whereas the black-dotted and red-dotted lines represent the first-order and second-order approximations respectively. Measurement iterations of length $T = 0.1$s.

Consider the following road network, in Figure 3.7, representing a typical urban-surveillance scenario with assets situated at either the intersections of roads or at the termination of a road. Assuming a ground map of all access routes to each asset is known, we can construct a graph $\mathcal{G} = \{V, W\}$ with a node set $V$ and a weighted adjacency matrix $W$, as in Figure 3.8. The node set $V$ consists of all assets, all intersections of routes, and all route termination points. The weighted adjacency matrix is $W \in \mathbb{R}^{\left|V\right| \times \left|V\right|}$, where $\left|V\right|$ (represents the cardinality of set $V$) is the total number of nodes in the graph. The matrix $W$ contains a positive element $W_{ij}$ if and only if there is an access route from node $j$ to $i$. We assume that all
Figure 3.7: Typical ground map with 14 nodes. The three red dots indicate asset locations. There are three road types; large, medium and small, representing the accessibility of the respective road way.

Figure 3.8: Graphical representation of Figure 3.7. Filled nodes indicate asset locations, defined by the asset index set $\vec{i} = (1, 8, 11)$, where hollow dots indicate road termination points or intersections. The three road types are assigned respective weights $(\hat{w}_1, \hat{w}_2, \hat{w}_3)$. 

41
weights $W_{ij}$ are contained in $[0, 1/U]$, and that each path weight, represented by the values $(\hat{w}_1, \hat{w}_2, \hat{w}_3)$, is defined by the system-designer to be proportional to the ease with which a target can travel along the corresponding route. In other words, a narrow alley-way would be assigned a weight near zero, whereas a super high-way would be assigned a weight near $1/U$. To account for the possibility that a target might approach an asset from a non-access route (i.e. over a rocky terrain, across a river, or through a series of interconnecting buildings), we assign to the map a constant $d \in [0, 1/U]$ indicating the possibility that a target might travel along a non-access route. For instance, if all assets were located in the middle of a desert, then $d$ would be near $1/U$ since there would be little need for a target to remain on a designated road.

We now use the weighted adjacency matrix to compute the eigenvector centrality. We define a new matrix, termed the adjusted weighted adjacency matrix as $\hat{W} \in \mathbb{R}^{\left|V\right| \times \left|V\right|}$ where each element satisfies $\hat{W}_{ij} = \max_{i, j \in V} \{W_{ij}, d\}$. For any choice of $d \in (0, 1/\left|V\right|]$, the matrix $\hat{W}$ is a positive matrix, and thus by the Perron-Frobenius Theorem, $\hat{W}$ will have a unique positive right-eigenvector, $m \in \mathbb{R}^{\left|V\right| \times 1}$, corresponding to the eigenvalue equal to the spectral radius of $\hat{W}$. The vector $m$ is thus the principal eigenvector of $\hat{W}$. We define $\bar{m}$ as the normalized eigenvector, $m$, such that $\bar{m}_i = m_i / (\sum_{i \in V} m_i)$, resulting in $\sum_{i \in V} \bar{m}_i = 1$. Now, we can define the well-known measure of the graph centrality of node $i \in V$ as the element $\bar{m}_i$.

Recall, the precise weights of the roads are chosen by the designer to represent how easily a target can move along a respective path. We define them arbitrarily as follows, $(\hat{w}_1, \hat{w}_2, \hat{w}_3, d) = (0.8, 0.5, 0.1, 0.05)/\left|V\right|$. For these values, the graph in Figure 3.8, has the associated weighted adjacency matrix and corresponding normalized principal eigenvector of,
\[ W = \begin{bmatrix}
    d \hat{w}_2 & d & d & d & d & d & d & d & d & d & d & d \\
    \hat{w}_2 & d & \hat{w}_1 & d & d & d & d & d & d & d & d & d \\
    d & \hat{w}_1 & d & d & d & d & d & d & d & d & d & d \\
    d & d & \hat{w}_1 & d & d & d & d & d & d & d & d & d \\
    d & d & d & \hat{w}_1 & d & d & d & d & d & d & d & d \\
    d & d & d & d & \hat{w}_1 & d & d & d & d & d & d & d \\
    d & d & d & d & d & \hat{w}_1 & d & d & d & d & d & d \\
    d & d & d & d & d & d & \hat{w}_1 & d & d & d & d & d \\
    d & d & d & d & d & d & d & \hat{w}_1 & d & d & d & d \\
    d & d & d & d & d & d & d & d & \hat{w}_1 & d & d & d \\
    d & d & d & d & d & d & d & d & d & \hat{w}_1 & d & d \\
    d & d & d & d & d & d & d & d & d & d & \hat{w}_1 & d \\
\end{bmatrix}, \quad \tilde{m} = \begin{bmatrix}
    0.032 \\
    0.046 \\
    0.054 \\
    0.071 \\
    0.041 \\
    0.055 \\
    0.113 \\
    0.054 \\
    0.051 \\
    0.102 \\
    0.129 \\
    0.090 \\
    0.059 
\end{bmatrix} \tag{3.31} \]

We now index the above vector, \( \tilde{m} \), by the asset index set \( \tilde{i} = [1, 8, 11] \) to obtain a measure of the vulnerability the assets in our system. Notice that the asset vulnerabilities in this example are \( \tilde{m}_i = [0.032, 0.113, 0.102] \) indicating that the assets indexed by node index \( i = 8 \) and \( i = 11 \) are roughly twice as vulnerable as the asset indexed by \( i = 1 \). Using the importance weight, \( w_a \), defined in the previous section, we now define the weighted vulnerability value for each asset \( a \) as,

\[
\tilde{w}_a = \frac{\tilde{m}_{i_a} w_a}{\sum_a \tilde{m}_{i_a} w_a} \tag{3.32}
\]

such that \( \sum_a \tilde{w}_a = 1 \).

Recall the definition of the TAT function from Section 3.3.1, but now with the updating asset weights \( \tilde{w}_a \),

\[
\rho' = \Gamma(s') = \sum_{a=1}^{A} \tilde{w}_a \gamma_{a,l} \tag{3.33}
\]

where \( \gamma_{a,l} \) is defined in Equation (3.18). The TAT function now considers the proximity of the threat, the accessibility of the asset, and also the threat associated with the target velocity in relation to all assets in our surveillance region.
3.3.3 Threat Estimate Uncertainty

From the previous sections, we have a function which maps the estimate of the target states to an estimate of the target threat. Since the state estimates themselves are not precise, that is, there is an associated error covariance matrix, the resulting threat estimate also has a degree of uncertainty. Using the error covariance matrix provided at the end of the previous micro-manager tracking interval, $P_l$ (beginning of the current macro-manager interval), and knowledge of the exact form of the threat function, we wish to derive the associated threat variance, $\varepsilon_n$, corresponding the the threat estimate, $\hat{r}_n$. The general problem of determining the propagation of uncertainty can be stated formally as follows,

Given a nonlinear function $f : \mathbb{R}^N \rightarrow \mathbb{R}^M$ and a random vector $x \in \mathbb{R}^N$ with an associated mean $\hat{x}$ and error covariance matrix $P_x \in \mathbb{R}^{N \times N}$, define $y = f(x)$ where $y \in \mathbb{R}^M$. Determine the resulting mean $\hat{y}$ and error covariance matrix $P_y \in \mathbb{R}^{M \times M}$ of the random vector $y$.

There are many different approaches one can take to obtain a solution to the above problem. The methods found in the literature can be divided into two main categories: empirical and analytical. Empirical methods, and the most general of all the approaches, largely fall under the classification of Monte Carlo methods [4, 33]. However, these approaches are very computationally expensive and cannot be performed in real-time. Conversely, the analytical methods of finding a solution to the above problem rely on finding a valid approximation of moments of the function of random variables. The usual method for this approximation is via Taylor series expansion. Some methods in the literature only include up to the first-order term [65], which are reasonable approximations assuming the function is roughly linear in the region near the linearization. However, when the function exhibits highly nonlinear behaviour, as in the case of the threat function, we are required to retain higher-order terms of the series to achieve more accurate analytic estimates. Various papers in the literature discuss second-order moment approximations [25, 56, 96, 100], albeit, many of them contain awkward notation and complicated representations. A very nice treatment of expressing higher order ap-
proximations to moments of functions of random variables is seen in [60] where an explicit representation of the Taylor expansion is presented using Kronecker products and partial derivative block matrices.

We start by defining \( e = s_n^l - s_n^\hat{l} \), where \( e \in \mathbb{R}^{4 \times 1} \) is assumed to be a zero-mean random vector. We wish to compute the error covariance of the threat, defined as \( P_\Gamma = \mathbb{E}\{(r - \mathbb{E}\{r\})(r - \mathbb{E}\{r\})^T\} \)

where \( r \) is the true threat value. If we only take the first-order approximation of Equation (3.22) and compute the error covariance matrix\(^4\) then the resulting approximation is,

\[
P_\Gamma \approx \mathbb{E}\{(r - \mathbb{E}\{r\})(r - \mathbb{E}\{r\})^T\} = E \left\{ (\Gamma(\hat{s}) + \nabla_s \Gamma e - \Gamma(\hat{s}) + \nabla_s \Gamma e - \Gamma(\hat{s}))^T \right\}
\]

where \( P \) is the error covariance matrix \( P_n^l \) with indices \( l \) and \( n \) omitted. The result of this error covariance approximation can be seen in Figure 3.9, which is evidently not a very good estimate of the true error.

We now present the second-order approximation to the error covariance of the threat. The method is similar; however, we now keep two terms of the expansion in Equation (3.22). Assuming Gaussian noise, the expression for the approximation of the error covariance of the threat is,

\[
P_\Gamma \approx \nabla_s \Gamma P (\nabla_s \Gamma)^T + \frac{1}{4} \nabla_s^2 \Gamma (D - ss^T) (\nabla_s^2 \Gamma)^T
\]

where \( \nabla_s \Gamma : \mathbb{R}^{4 \times 1} \rightarrow \mathbb{R}^{m \times m} \), with \( m = 1 \). In general the error covariance matrix of the output variable is of dimension \( m \times m \).
Figure 3.9: True threat error covariance compared to first-order approximation of threat error. True threat error is computed by taking the absolute difference between the true threat value and the first-order approximation to the threat mean. Colored (not red) lines represent the true threat error where red-dotted lines represent the estimated value.

With the details of the derivation presented in the Appendix (Section B.4). The second-order error estimate accuracy compared to the true threat error can be seen in Figure 3.10.
Figure 3.10: True threat error covariance compared to second-order approximation of threat error. The true threat error is computed in the same way as the first-order case (see caption of Figure 3.9); however, we now use the second-order approximation to the threat mean. Colored (not red) lines represent the true threat error where red-dotted lines represent the estimated value.

Comparing the first-order error approximation (Figure 3.9) to the second-order approximation (Figure 3.10) for each target clearly shows the improvement in accuracy. Note that the “true” threat error is not the same in the two plots, since the threat mean estimate was different in the first and second order cases.

3.4 Threat Rank Function

From the previous section, we now have an estimate of the threat value for each target \( l \) as well as an associated estimation in the uncertainty of the computed threat value. Given a target with a particular threat estimate, denoted as the pair \((\hat{r}_n^l, e_n^l)\), it would be useful to assign a priority rank to the target. We denote the rank of
targets in our system, at time $n$, by the random variables,

$$X_{1n}^1, X_{2n}^2, \ldots, X_{Ln}^L$$

(3.42)

where $X_{ln}^l$ refers to the rank of target $l$. The rank can be interpreted as a measure of the incentive (the immediate reward) to observe a particular target. Thus the rank assigned to a target is proportional to the level of threat and the uncertainty of that target. The threat rank function function, see Figure 3.1, serves the purpose of assigning this rank to each target. With knowledge of the rank of each target in the surveillance region, we aim to compute an optimal policy to maximize the total (immediate) reward associated with the observations of targets at various levels of threat.

It is important to notice that the level of threat of a target is a quantity that is completely controlled by the state of the targets. Since it is assumed that the targets behave independently of the actions of the radar (as opposed to a game theoretic formulation [47, 82]), we have no effect on the value of the threat. We do however, have an effect on the value of the threat uncertainty. By observing a particular target, we consequently reduce the uncertainty in the state estimates, and since the threat error, $e_{ln}^l$, is based off this uncertainty, we reduce the threat error as a result.

The purpose of the macro-manager scheduler is not to simply reduce the uncertainty in the threat error of all targets, regardless of the threat value, but instead, maximize a reward for observing a target at a specific threat error and the threat value. This follows the same argument regarding the semantic approach to prioritization (as discussed in the motivation section of the introduction). Consider the case where we have an equivalent threat error in two targets of differing threat estimates; this should translate into different incentives for observing each target, and thus different rewards. Observe the following example scenario for justification. Consider two targets, $l = 1$ and $l = 2$, with threat estimates and errors obeying the following relationships,

$$\hat{r}_{ln}^1 > \hat{r}_{ln}^2 \quad \text{and} \quad e_{ln}^l = e_{ln}^2.$$  

(3.43)

That is, the value of the threat estimate of target $l = 1$ is higher than that of target
but they both have the same threat error. From a rather intuitive standpoint, it is evident that an uncertainty in a high threat should be of more concern than an uncertainty in a low threat. As mentioned earlier, since a target’s rank of a target is a measure of its level of threat, our threat reward function should capture this concept and assign a higher rank to target \( l = 1 \).

The threat rank function is responsible for specifying the rank of each target. We consider both the level of threat and the threat variance in the definition of the rank,

\[
X_n^l = b r_n^l + e_n^l
\]  (3.44)

Given the random variable, we wish to define it by a specific distribution. For simplicity, we consider the family of probability distributions that can be defined by the two-parameters, mean and variance. The mean and variance of the above rank can be computed as follows, assuming \( r_n^l \) is the only random quantity and \( e_n^l \) is completely deterministic,

\[
\mu_{X_n^l} = \mathbb{E}\{X_n^l\} = \mathbb{E}\{b r_n^l + e_n^l\} = b \mathbb{E}\{r_n^l\} + e_n^l
\]  (3.45)

\[
\sigma^2_{X_n^l} = \text{Var}\{X_n^l\} = \text{Var}\{b r_n^l + e_n^l\} = b^2 \text{Var}\{r_n^l\} + e_n^l
\]  (3.46)

where the coefficient \( b \geq 0 \) is chosen by the system-designer to define which factor, threat or error, they consider to be a higher threat.

The justification for the form of rank mean and rank variance functions is as follows. First, we qualitatively describe the labeled vertices in Figure 3.11 as follows: (1) Low error in a low threat value, (2) Low error in a high threat value, (3) High error in a low threat value, (4) High error in a high threat value. Recall that
Figure 3.11: The general form of the functions specifying the target rank mean, $\mu_{X^l_n}$, and target rank variance, $\sigma^2_{X^l_n}$. The target rank mean increases in both the threat estimate, $\hat{r}^l_n$, and the threat estimate error, $e^l_n$. The target rank variance increases monotonically in $e^l_n$ and is independent of $\hat{r}^l_n$.

On the contrary, the target with the lowest error (lowest uncertainty) in the lowest threat estimate should be considered the safest target and be given the lowest rank mean. This intuition is reflected in Figure 3.11(a), with the highest and lowest ranks represented by vertices (4) and (1), respectively. The vertices (2) and (3), corresponding to the cases low error in a high threat, and high error in a low threat, respectively, are shown to be assigned roughly the same rank mean, with a slightly higher rank assigned to case (3). If we are indifferent between the cases (2) and (3), we would set the coefficient $b = 1$. However, if we consider case (2) to be a higher threat than case (3), we would set $b > 1$. Conversely, the case $b < 1$ corresponds to considering case (3) to be a higher threat than case (2). The rank mean function exhibits the following dependency on the threat estimate, $\hat{r}^l_n$, and threat error, $e^l_n$.

1.) Consider two targets with equivalent threat estimates, $\hat{r}^1_n = \hat{r}^2_n$. Then $\mu_{X^l_1} >$
\[ \mu_{X^2_n} \text{ for } e_{n}^1 > e_{n}^2. \]

2.) Consider two targets with equivalent threat error estimates, \( e_{n}^1 = e_{n}^2 \). Then \( \mu_{X^1_n} > \mu_{X^2_n} \) for \( \hat{r}_{n}^1 > \hat{r}_{n}^2 \).

The above relationships are intuitive: the target rank mean increases in both the threat estimate and the level of threat error. The rank variance is only dependent upon the threat error, by Equation (3.46).

### 3.4.1 Distribution Assignment

Given that we have values for the rank mean, \( \mu_{X^i_n} \), and rank variance, \( \sigma^2_{X^i_n} \), of the random variables, \( X^1_n, X^2_n, \ldots, X^L_n \), we now aim to specify the probability distributions of the target ranks. To provide rationale for assigning a particular distribution to the target rank, we perform numerical simulations. We consider a fixed true target trajectory and compute the track estimate multiple times, with each evaluation resulting in a threat value. We then produce a histogram of the threat ranks from the threat estimates, the results of which can be seen in Figure 3.12.

![Figure 3.12: The distribution of threat rank values for a given fixed trajectory.](image)

The threat rank evaluations result in a unimodal, symmetric distribution. For simplicity, we quantize the threat rank level into \( N = 10 \) discrete levels. Thus the rank distribution of each target will have the same (discrete) support. As a result,
we consider the class of discrete distributions. Consider the binomial distribution, a discrete probability distribution with the following PMF,

\[
Bin(N, p) = \binom{N}{r} p^r (1 - p)^{N-r}
\]  

with mean, \( \mu_b = N p \), and variance, \( \sigma_b^2 = N p (1 - p) \). For a fixed \( N \), the dependence of the mean and variance on the parameter, \( p \), can be seen in Figure 3.13.

\[a\]  

From Equations (3.45 and 3.46) we have expressions for both the rank mean and rank estimate. Given that we have fixed the possible threat values to \( N = 10 \) for every target, we now must choose the remaining parameter, \( p \in [0, 1] \), to best fit \( \mu_{X_l^n} \) and \( \sigma_{X_l^n}^2 \). We do this by computing the least squares estimation of the parameter, \( p \). We start by defining the objective: we wish to obtain the least squares estimate of the probability parameter, \( p^{l_s}_n \), to minimize the sum of the square of residuals. That is,

\[
p^{l_s}_n = \arg \min_{p_l^n \in [0, 1]} \| \rho^{l}_n \|_2^2
\]  

The quantity, \( \rho_{n}^{l} \), is defined as,

\[
\rho_{n}^{l} = \left[ \frac{\mu_{X_l^n}}{\sigma_{X_l^n}^2} \right] - \left[ \frac{N p_{n}^{l}}{N p_{n}^{l}(1 - p_{n}^{l})} \right] = \left[ \frac{\mu_{X_l^n} - N p_{n}^{l}}{\sigma_{X_l^n}^2 - N p_{n}^{l}(1 - p_{n}^{l})} \right]
\]  

For simplicity, we will omit the time, \( n \), and target, \( l \), indices for the remainder of
the calculations. Given the form of $\rho$, Equation (3.48) represents a nonlinear least-squares problem, with $\|\rho\|_2^2 = (\mu - Np)^2 + (\sigma^2 - Np(1 - p))^2$. The following lemma characterizes the solution of the nonlinear least-squares problem.

**Lemma 3.4.1.** For any $n > 0$, mean $\mu \in \mathbb{R}$ and variance $\sigma^2 \geq 0$, the non-linear least-squares optimization,

$$p^* = \arg\min_{p \in \mathbb{C}} (\mu - np)^2 + (\sigma^2 - np(1 - p))^2$$

has the following unique real solution,

$$p^* = \frac{1}{2} + \frac{a}{3(2^{2/3})n(b + \sqrt{b^2 + 4a^2})^{1/3}} - \frac{(b + \sqrt{b^2 + 4a^2})^{1/3}}{3(2^{4/3})n}$$

where $a = 3n(4\sigma^2 + n)$ and $b = 54n^2(-2\mu + n)$.

**Proof.** See Appendix C.1.1. \qed

We refer to the real solution as the least-squares estimate of the parameter $p$, denoted by $p_n^{ls} \in \mathbb{R}$. As can be seen in Figure 3.14, the value is confined to the range $[0, 1]$. We show that this is indeed true in the following lemma.

**Lemma 3.4.2.** The least-squares estimate $p_n^{ls} \in \mathbb{R}$ is confined to the range $[0, 1]$ for $N, \sigma^2 > 0$ and $\mu \in [0, N]$.

**Proof.** See Appendix C.1.2. \qed

The fit of the least-squares estimate, $p_n^{ls}$, is now analyzed. We plot each term of the objective function, $\|\rho_n^{l}\|_2^2$, in Figure 3.15. The absolute error values are defined...
Figure 3.14: The least-squares estimate of the parameter $p_n^I$ of a binomial distribution for a given rank mean, $\mu_{X_n^I}$, and variance, $\sigma_{X_n^I}^2$. Notice that for $\mu_{X_n^I} \in [0, N]$ and $\sigma_{X_n^I}^2 \in [0, N/4]$ the least-squares estimate is confined to $p_n^I \in [0, 1]$.

Figure 3.15: The relative errors in the (a) rank mean, $\mu_{X_n^I}$, and (b) rank variance, $\sigma_{X_n^I}^2$, obtained from the least-squares estimate, $p_n^I$. 
as,

\[ |e_\mu| = \frac{|\mu_{X_n^l} - Np_n^{l^*}|}{\mu_{X_n^l}}, \quad (3.52) \]

\[ |e_{\sigma^2}| = \frac{|\sigma_{X_n^l}^2 - Np_n^{l^*}(1 - p_n^{l^*})|}{\sigma_{X_n^l}^2}, \quad (3.53) \]

Note that since the least-square estimate \( p_n^{l^*} \) is dependent on both the rank mean and rank variance, the error quantities \( |e_\mu| \) and \( |e_{\sigma^2}| \) are also dependent upon these variables.

Given the least-squares estimate, \( p_n^{l^*} \), the binomial approximation to the rank of each target is defined as,

\[ \hat{X}_n^l \sim \text{Bin}(N, p_n^{l^*}) \quad (3.54) \]

Recall, from Theorem 3.2.2, we are required to order the ranks of all targets, \( X_1^1, X_1^2, \ldots, X_1^L \) according to the likelihood ratio order. Since we have assigned a binomial distribution to approximate each target’s threat rank, denoted \( \hat{X}_n^1, \hat{X}_n^2, \ldots, \hat{X}_n^L \), we are now interested in conditions such that we can order the binomial distributions. Recall that we have quantized the threat for each target to a finite number of discrete levels, thus all targets’ binomial distributions have a common parameter, \( N \). We present the following lemma stating conditions on a binomial distribution’s parameter \( p \), such that likelihood ratio dominance between two binomial random variables is achieved.

**Lemma 3.4.3.** Consider two random variables, denoted \( K_1 \sim \text{Bin}(N, p_1) \) and \( K_2 \sim \text{Bin}(N, p_2) \) where \( N \) is a positive integer. Then \( K_1 \geq_{LR} K_2 \) if and only if,

\[ p_1 \geq p_2 \quad (3.55) \]

**Proof.** See Appendix, Section C.1.3. \( \square \)
As a result of Lemma 3.4.3, by ordering the binomial parameters we are or-
dering the corresponding binomial random variables, $\hat{X}_1, \hat{X}_2, \ldots, \hat{X}_L$, with respect
to the likelihood ratio order. Consequently, the corresponding target observation
order is specified, the performance of which is analyzed in the numerical results in
Section 5.

We now introduce a constraint for the operation of the micro-manager which
takes into account the priority rank of each target.

**Maximum Tracking Time Ratio Vector**

The purpose of this section is to discuss a method to incorporate the threat level of
each target into the operation of the micro-manager. As the reader will see later,
the goal of the micro-manager controller is to maximize the mutual information be-
tween a target’s observations and states. That is, the micro-manager minimizes the
error in the track estimates, defined by their respective error covariances, of each
target. An assumption of the macro-manager is that we will process all items in the
queue. Consider the case where the micro-manager does not consider the threat
of the targets as a factor in its operation. This would result in the micro-manager
achieving optimal (in a sense defined by the micro-manager) state estimates of
each target, regardless of its level of threat. It is clear that this is not ideal in most
scenarios, as we would care much more about the accuracy of states estimates of
a high threat target than the accuracy of estimates of a low threat target. Thus, the
micro-manager needs to take the threat level of each target into account.

We consider the threat of each target at the level of the micro-management
controller by imposing a **maximum tracking time** constraint for each queued target,
$\bar{\tau}_l$, dependent upon its level of threat. The justification here is that a target should
be allowed a shorter tracking time if it is not considered a high threat, reflecting the
fact that we wish to not spend many resources on a low threat target. The maximum
tracking time of the $l$th target in the queue is defined as,

$$\bar{\tau}_l = \max \{ \lceil \tau_1^l \rceil, \tau_l \}$$ (3.56)

where $\tau_1^l$ is the optimal stopping time specified by the micro-manager for the first
target in the queue. The quantity, $\tau_n$, is termed the tracking time constraint vector with each element defined as,

$$t_n^{o_1} = \frac{\hat{\tau}_n^{o_1}}{\tau_n^{o_1}} = \frac{\mathbb{E}\{\hat{\tau}_n^{o_1}\}}{\mathbb{E}\{\hat{\tau}_n^{o_1}\}}$$

(3.57)

which specifies the ratio of the tracking time allotted to each target relative to the stopping time of the first target in the queue, $\tau_n^{o_1}$. We define this ratio as the relative expected rank of the respective target to the expected rank of the first target in the queue.

Recall the optimal ordering theory presented in Section 3.2.2, and the result in Equation (3.4); we know that the following is satisfied,

$$\hat{\tau}_n^{o_1} \geq_{LR} \hat{\tau}_n^{o_2} \geq_{LR} \ldots \geq_{LR} \hat{\tau}_n^{o_L} \Rightarrow \mathbb{E}\{\hat{\tau}_n^{o_1}\} \geq \mathbb{E}\{\hat{\tau}_n^{o_2}\} \geq \ldots \geq \mathbb{E}\{\hat{\tau}_n^{o_L}\}$$

(3.58)

for the order $(o_1, o_2, \ldots, o_L)$. Thus, $\mathbb{E}\{\hat{\tau}_n^{o_1}\}$ is the maximum expected rank of all targets. As a result, we can write the tracking time ratios, in Equation (3.57), as,

$$t_n^{o_1} = 1 \geq t_n^{o_2} \geq t_n^{o_3} \geq \ldots \geq t_n^{o_L}$$

(3.59)

The above specifies for the first target in the queue, $o_1$, the micro-manager should track it for 100% of the specified stopping time, defined as $\tau_n^{o_1}$, and targets $o_2, o_3, \ldots, o_L$ for ratios of $\tau_n^{o_1}$ specified by the respective values $t_n^{o_2}, t_n^{o_3}, \ldots, t_n^{o_L}$.

By imposing maximum tracking time constraints on the lower priority targets, we are making the estimates of the respective target suboptimal; however, this is by design as we care less about the accuracy of states of lower priority targets. This sacrifice comes at the reward of processing the target queue in a shorter time-frame, freeing up the resources of the radar for additional functions.

---

5 The operation $\lceil \cdot \rceil$ is termed the ceiling function and specifies to round the argument up to the nearest integer.
Chapter 4

Radar Micro-manager

The target prioritization provided by the macro-manager is computed at a particular instant in time, using the current (and potentially past) observed target states. Since the targets evolve over time, their respective threat to assets will vary, and the ideal prioritization will also change in time. This implies that at some point, we wish to stop using the current target prioritization and give control back to the macro-manager to provide an updated prioritization. The decision on whether to continue or stop with the provided allocation is based upon noisy, Bayesian estimates of the target states, with the stochastic nature of the stopping time problem arising from the fact that the estimates are assumed to be received with a non-unity probability of detection. The purpose of the micro-manager is to determine the optimal time to return control back to the macro-manager, thus providing a solution to the task scheduling problem.

In general, this allocation can refer to a variety of prioritizations. The allocations mentioned throughout the dissertation are: 1) a priority vector, or 2) a prioritized queue of target observations. Referring to the target and measurement model from Section 2.2, in the case of the prioritized queue, the tracker devotes all of its resources to the current target in the queue and no resources to the remaining targets. Mathematically, assuming target \( \bar{l} \) is the current target in the queue, this is represented by \( \nu_{\bar{l}} = 1 \) and \( \nu_{l} = 0 \) for \( l \neq \bar{l} \), stating that target \( \bar{l} \) is devoted a Kalman filter where all other targets are updated using Kalman predictors. It is evident that the priority vector allocation is a more general form than that of the prioritized
queue, thus we will present the micro-management optimization problem in terms of the priority vector, following the results presented in [51].

The problem of resource management using POMDPs has been well researched. Multiple papers have formulated the resource management problem as a POMDP [15, 41, 75]; however, the solution techniques used for the formulations are only tractable for small, unrealistic scenarios. In realistic applications, solutions to POMDPs are intractable (the problem is PSPACE hard [58]). For this reason, structural results are obtained to make the solutions of POMDPs reflecting realistic scenarios tractable. In particular, we are concerned with determining necessary and sufficient conditions such that the optimal solution varies monotonically with the system parameters (a concept termed monotone comparative statistics\(^1\)).

There has been a large amount of work presented in regard to structural results for POMDPs, with the prominent works being that of Albright [2] and Lovejoy [59]. In particular, Albright [2] provides conditions for a two-state system such that the optimal reward and optimal action are both monotone in the current information state. Lovejoy [59] generalizes the results of Albright to \(n\) states by invoking the monotone likelihood ratio (partial) order on the set of probability distributions on the information state. Krishnamurthy [48] further generalizes the results of Lovejoy by considering non-standard POMDPs (nonlinear in the information state) and by introducing an ordering that results in less restrictive sufficient conditions to achieve a monotone form of the optimal policy.

The interest in this chapter is the application of the previously formulated structural results to the problem of radar resource management in GMTI systems. Work has been presented using structural results in the context of radar resource management [48, 49]. Specifically, [48] presents sufficient conditions on the cost functions, dynamics of the Markov chain, and target and observation probabilities such that the optimal policy has a threshold structure with respect to monotone likelihood ratio ordering on lines spanning the information simplex. The work of [49] applies the results of [48] to the problem of determining the optimal time to stop observing a particular target. Recent work [51] generalizes the results of [49] by determining conditions to characterize a monotone threshold policy for determining conditionals.

\(^1\)Monotone comparative statistics are concerned with optimal solutions which vary monotonically with respect to its parameter, with the paramount source being the work of Topkis [90].
the optimal stopping time associated with a prescribed priority allocation (instead of prescribing the optimal time to spend observing a single target). The presentation for the sufficient conditions, such that the optimal policy takes a monotone threshold form, will follow the discussion in [51]. Once we have shown the monotone structure, we are able to efficiently compute a parameterized approximation to the policy. The parameterized policy is computed by minimizing the sample-path cost, a process which is based on the target and observation model and the system parameters. Since the computation is only based on the model, it is computed offline, allowing the optimal action to be obtained (in real-time) solely on the current state of the system.

4.1 Radar Micro-manager Structure

The radar micro-manager block consists of three main components, as can be seen in Figure 4.1. First, the multi-function radar, with an electronically steered beam, is responsible for transmitting and receiving radar pulses, from which target data is generated. Second, the Bayesian tracker obtains the information for all targets from the multi-function radar and forms track estimates and associated error covariance matrices. If necessary, we direct the reader to Section 2.3 where the details of the Bayesian tracker were originally discussed. Lastly, the micro-manager scheduler is responsible for specifying the optimal action (from the optimal policy) based on the Bayesian estimates received from the tracker and the target prioritization provided by the radar macro-manager. The action generated by the scheduler controls for how long the radar uses the allocation provided by the macro-manager. The focus of this chapter is on the contents of the micro-manager scheduler block, primarily the sufficient conditions such that the optimal policy takes a monotone threshold form.

4.2 Optimal Stopping as a Sequential Decision Process

We formulate the radar micro-manager as a multivariate sequential decision process, a form of a POMDP. We start by introducing the components of a sequential decision process, which is comprised of the following five ingredients [73].
Figure 4.1: A schematic of the radar micro-manager.

1. **Decision epochs**: defines when decisions can be made.

2. **State space**: the set of allowable states that the system can reside in.

3. **Transition probabilities**: the probability that a particular state-to-state transition will occur given a particular action.

4. **Action space**: the set of actions available to the decision maker.

5. **Costs and Rewards**: the immediate costs and rewards associated with choosing a particular action in a particular state.

In addition to the above five components, we define an optimization objective, such that the actions are chosen to minimize a cost (or equivalently maximize a reward), the details of which are provided in Section 4.2.5.

A Markov decision process (MDP) is a specific type of sequential decision model where we place the restriction that the choice of an action for time $m + 1$, 
is based solely on the information available at time \( m \). Also, for the purposes of the micro-manager, we focus on a generalization of an MDP, where it is assumed that we cannot directly observe the underlying states, termed a Partially Observed Markov Decision Process (POMDP). This is justified due to the fact that we are unable to observe the true tracks of or targets due to the presence of noise and measurement errors.

### 4.2.1 Decision Epochs

The decision epochs refer to the times at which the micro-manager can specify an action. For both listed target prioritizations (Section 2.1), the decision epochs are the iterations \( k \) on the micro-manager time scale (see Figures 2.2 and 2.3). That is,

\[ k = \{1, 2, \ldots \}. \quad (4.1) \]

The above implies that the decision epochs are an infinite set; however, in Section 4.2.5 we formulate the micro-manager optimization as an optimal stopping time problem, with a stopping rule. Thus the decision epoch set is indeed finite, but since the stopping time is unknown (stochastic) we define at set of decision epochs as stated.

### 4.2.2 Target States and State Update

The information required to make a decision concerning the optimal action is contained within the state of the system. We define the state of the micro-manager problem to be the set of error covariance matrices \( \mathbf{P}_k \).

\[ \mathbf{P}_k = (\tilde{\mathbf{P}}_k, \mathbf{P}_k) = (\{\tilde{\mathbf{P}}^1_k, \tilde{\mathbf{P}}^2_k, \ldots, \tilde{\mathbf{P}}^L_k\}, \{\mathbf{P}^1_k, \mathbf{P}^2_k, \ldots, \mathbf{P}^L_k\}) \quad (4.2) \]

where \( \tilde{\mathbf{P}}^l_k = \mathbb{E}\left\{ (s^l_k - \mathbb{E}(s^l_k)) \left( s^l_k - \mathbb{E}(s^l_k) \right)^T \right\} = \mathcal{L}^{k-1}(\mathbf{P}^l_0) \) is the purely predicted (measurement-free) estimate and can be computed for any time \( k \) solely from the initial error covariance matrix \( \mathbf{P}_0 \). The notation \( \mathcal{L}^{k-1}(\cdot) \) refers to the \( k - 1 \) iteration of the Lyapunov function. The quantity \( \mathbf{P}^l_k \) is measurement-dependent and is defined as \( \mathbf{P}^l_k = \mathbb{E}\left\{ (s^l_k - \mathbb{E}(s^l_k|z^l_{k-1})) \left( s^l_k - \mathbb{E}(s^l_k|z^l_{k-1}) \right)^T \right\} = \mathcal{R}(\mathbf{P}^l_{k-1}, z^l_{k-1}) \), termed the filtered estimate. The quantity \( \tilde{\mathbf{P}}^l_k \) can be interpreted as the error covariance
of the state estimate when not conditioned on any measurements; whereas, \( P^l_k \) is conditioned on the measurement \( z^l_{k-1} \).

The state update is dictated by the modified Riccati update and Lyapunov update. Based on the current state \( P^l_k \) and the received measurement \( z^l_k \), we compute the next state \( P^l_{k+1} \) using one of the following equations,

\[
P^l_{k+1} = \mathcal{R}(P^l_k, z^l_k) \overset{\text{def}}{=} FP^l_k F^T + Q - 1_{\mathbb{R}^d} FP^l_k H^T (HP^l_k H^T + R)^{-1} H P^l_k F^T \tag{4.3}
\]

\[
P^l_{k+1} = \mathcal{L}(P^l_k) \overset{\text{def}}{=} FP^l_k F^T + Q \tag{4.4}
\]

where the modified Riccati update, Equation (2.20, 4.3), is used if we are filtering a particular target, and conversely, the Lyapunov update, Equation (2.21, 4.4), is used if we are predicting a particular target, first defined in Section 2.3. The complete state update is dependent upon the target prioritization specified by the macro-manager.

1. Target Priority Allocation Vector

The covariances \( \bar{P}^l_k \) are updated, as stated previously, using the Lyapunov equation. The covariances \( P^l_k \) are updated using the Riccati equation, taking into account the priority vector \( \nu = [\nu^1, \nu^2, \ldots, \nu^L] \). Thus,

\[
P_{k+1} = \left( \mathcal{L}(\bar{P}^l_k), \mathcal{R}(P^l_k, z^l_k) \right) \text{ for each } l \in \mathcal{L} \tag{4.5}
\]

2. Target Priority Queue

Let \( o_l \) be the current target in the observation queue. For a given macro-management cycle, \( n \), and micro-manager tracking interval \( k = 1, 2, \ldots, \tau^{o_l}_n \), target \( o_l \) is updated via the Riccati equation,

\[
P^{o_l}_{k+1} = \mathcal{R}(P^{o_l}_k, z^{o_l}_k) \tag{4.6}
\]

whereas the states of all other targets \( l \neq o_l \) are updated via the measurement-free Lyapunov equation,

\[
P^l_{k+1} = \mathcal{L}(P^l_k) \text{ for all } l \neq o_l \tag{4.7}
\]
Similarly, as the queued targets are processed, state updates are made accordingly, with the radar manager only filtering the current queued target and predicting all other targets.

The rationale for listing multiple choices for the target prioritization is to illustrate the fact that the micro-manager specifies the optimal time to terminate any allocation provided by the macro-manager, and does not just serve the purpose of specifying the optimal time to stop tracking a particular target (although this is one of the choices of the prioritization).

A decision on the action for the next iteration $k + 1$ is dictated by an optimal policy $u_{k+1} = \eta^*(P_{k+1})$ (discussed in Section 4.2.5) and is based on the updated state $P_{k+1}$. The timeline of events for micro-manager can be seen in the following diagram, Figure 4.2.

![Micro-manager timing diagram](image)

**Figure 4.2:** Micro-manager timing diagram.
4.2.3 Action Space

Recall that the objective of the micro-manager is to determine the optimal time to stop using the current target prioritization and release control back to the macro-manager. Given that the problem is to find the optimal stopping time to terminate tracking the current target or the current priority allocation, the action set is simply,

\[ \mathcal{U} = \{ u^1, u^2 \} \]  

where, \( u^1 \rightarrow \text{stop} \) and \( u^2 \rightarrow \text{continue} \), which refers to whether the scheduler chooses to stop or continue with the current prioritization.

4.2.4 Operating and Stopping Costs

Operating Cost

It is now necessary to define the operating cost and stopping cost of the radar, corresponding to the actions \( u^2 \) and \( u^1 \), respectively. The operating cost is assumed to be fixed in time \( k \), and independent of the current state and measurements. We denote the operating cost as,

\[ \bar{C}(s_k, Z_k, u^2) = c_\nu > 0 \]  

Mutual Information Stopping Cost

The stopping cost is dependent upon both the states of the targets, \( s^I_k \), and the measurements, \( Z^I_k \). We define the cost in terms of a measure of mutual information between the target states and the target measurements. The definition of mutual information is as follows,

**Definition 4.2.1** (Cover and Thomas [17]). The mutual information between two random variables, \( X \) and \( Y \), is defined as,

\[ I(X; Y) = \int_x \int_y p(x, y) \log \left( \frac{p(x, y)}{p(x)p(y)} \right) \]
where \( p(x) \) and \( p(y) \) represents the probability distributions of \( X \) and \( Y \) respectively with \( p(x,y) \) representing the joint probability distribution.

With some manipulation, the mutual information can be written as,

\[
I(X;Y) = \int_x \int_y p(x,y) \log \left( \frac{p(x,y)}{p(x)p(y)} \right) dydx 
\]

(4.11)

\[
= \int_x \int_y p(x,y) \log \left( \frac{p(x|y)}{p(x)} \right) dydx 
\]

(4.12)

\[
= \int_x \int_y p(x,y) \left[ \log(p(x|y)) - \log(p(x)) \right] dydx 
\]

(4.13)

\[
= \int_x \int_y p(x,y) \log(p(x|y)) dydx - \int_x \int_y p(x,y) \log(p(x)) dydx 
\]

(4.14)

\[
= -\int_x p(x) \log(p(x)) dx - \left( -\int_x \int_y p(x,y) \log(p(x|y)) dydx \right) 
\]

(4.15)

The entropy (representing the amount of hidden information in a random variable) of a continuous random variable \( X \), termed the differential entropy, is defined as,

\[
H(X) = -\int_x p(x) \log(p(x)) dx 
\]

(4.16)

The conditional entropy of a random variable \( X \), conditioned on another random variable \( Y \) is defined as,

\[
H(X|Y) = -\int_x \int_y p(x,y) \log(p(x|y)) dydx 
\]

(4.17)

The conditional entropy \( H(X|Y) \) represents the amount of uncertainty “left-over” in \( X \) given knowledge about \( Y \). Equation (4.15) can be written in terms of differential entropies as,

\[
I(X;Y) = H(X) - H(X|Y) 
\]

(4.18)

Note, that mutual information is **commutative**, in the sense that the uncertainty reduction in \( X \) by knowing \( Y \) is the same as the uncertainty reduction in \( Y \) by
Knowing $X$, that is, $I(X;Y) = I(Y;X)$.

Given that the Bayesian tracker is providing the scheduler with target state estimates that are distributed as multi-variate Gaussian random variables, it is necessary to state the following result on the entropy of a multi-variate Gaussian distribution.

**Theorem 4.2.2** (Entropy of a multi-variate Gaussian distribution, [17] p.249). Let $X \in \mathbb{R}^n$ have a multi-variate Gaussian distribution with mean $\mu_X$ and covariance $K_X$. Then

$$H(X) = \frac{n}{2} \log (2\pi e |K_X|)$$

(4.19)

where $|K_X|$ denotes the determinant of $K_X$.

Define $\bar{l}$ as the highest priority target. In the case of the macro-manager assigning a priority vector, $\bar{l}$ would be defined as $\bar{l} = \arg\max_{l \in \mathcal{L}} (\nu_l)$. If the macro-manager has specified a priority queue, the current target in the queue is assigned the index $\bar{l}$.

From the definition of $P_k$ in Section 4.2.2, and Theorem 4.2.2, the mutual information, in Equation 4.18, at time $k$ for target $l$ (indices omitted), can now defined in terms of the covariances $\bar{P}_l$ and $P_l$ as,

$$I(s^l_k;Z^l_k) = H(s^l_k) - H(s^l_k|Z^l_k)$$

(4.20)

$$= \frac{n}{2} \log (2\pi e |P_l|) - \frac{n}{2} \log (2\pi e |\bar{P}_l|)$$

(4.21)

$$= \frac{n}{2} \left[ \log |P_l| + \log (2\pi e) - \log |\bar{P}_l| - \log (2\pi e) \right]$$

(4.22)

$$= \frac{n}{2} \left[ \log |P_l| - \log |\bar{P}_l| \right]$$

(4.23)

$$= 2 \left[ \log |P_l| - \log |\bar{P}_l| \right]$$

(4.24)

We define a more general form of the above mutual information equation.
through the inclusion of positive constants $\alpha, \beta \in \mathbb{R}^L_+$,

$$\hat{I}\left(P^i_k, \alpha^i, \beta^i\right) = \alpha^i \log|\bar{P}^i_k| - \beta^i \log|P^i_k|.$$  \hfill (4.25)

The constants $\alpha, \beta$ are chosen to affect the behaviour of the scheduler. Note that when $\alpha^i = \beta^i = 2$, for all $l \in \mathcal{L}$, the expression in Equation (4.25) agrees with the classical definition of mutual information in Equation (4.24).

We now define a stopping cost of the following form [51],

$$\bar{C}(P_k, \alpha, \beta, u_k = u^1) = -\hat{I}\left(P^i_k, \alpha^i, \beta^i\right) + F\left[\hat{I}\left(P^i_k, \alpha^i, \beta^i\right) : l \neq \bar{l}\right]$$ \hfill (4.26)

where $F : \mathbb{R}^{L-1} \to \mathbb{R}^{L-1}$ is a monotone (order preserving) function for each of its $L - 1$ elements. The stopping cost, Equation (4.26), is an intuitive choice. For a clear understanding, consider a scenario where we do not allow for the possibility of missed detections, that is $p_{id} = 1$. As the radar observes target $l = \bar{l}$, the conditional differential entropy $H(s^i_k|Z^i_k)$ will decrease as we condition upon more measurements. Thus, by the definition of mutual information, see Equation (4.20), the mutual information between the target states $s^i_k$ and the measurements $Z^i_k$ will monotonically increase over time, that is,

$$\hat{I}\left(P^i_{k+1}, \alpha^i, \beta^i\right) \geq \hat{I}\left(P^i_k, \alpha^i, \beta^i\right)$$ \hfill (4.27)

Thus, resulting in the first term of the stopping cost to decrease as $k$ increases. Simultaneously, the mutual information between the states of the unobserved targets, $s^j_k, j \neq i$, and their respective measurements, $Z^j_k, j \neq i$, will decrease in time $k$. As a result, since $F$ is assumed to be monotone, the stopping cost will decrease in $k$,

$$\bar{C}(P_k, \alpha, \beta, u^1) \geq \bar{C}(P_{k+1}, \alpha, \beta, u^1)$$ \hfill (4.28)

The cost for stopping becomes “cheaper” the longer we spend observing target $l = \bar{l}$. Given that the operating cost of the radar is a fixed constant in time $k$, see Equation (4.9), the cost for stopping will eventually become less than the cost for continuing, and it would be logical for the scheduler to specify the stop action, $u^1$.

Recall that in general, we do allow for the possibility of missed detections, thus
$p_d < 1$. As a result, the state $P_k$ is a random variable, and the inequalities stated in Equations (4.27) and (4.28) are no longer guaranteed to hold, thus stopping in finite time ($\tau < \infty$) is not assured. To alleviate this problem, we impose a maximum stopping time, $\bar{\tau}$, such that if $k = \bar{\tau}$, action $u_k = u_\bar{\tau} = u^1$ is automatically chosen.\footnote{This maximum stopping time is encoded in the $\iota_n$ vector defined by the macro-manager.} Thus we have defined a stopping rule and a decision to stop will be made at some time $\tau < \infty$ almost surely (with probability one).

Recall the discussion from Section 2.1 concerning the purpose of the macro-manager. The specific stopping cost is dependent upon the prioritization provided by the macro-manager,

1. **Target Priority Allocation Vector**
   In this case, the macro-manager provides the micro-manager with a vector, $v = [v^1, v^2, \ldots, v^L]$ corresponding to the priority allocated to each target. In this case, the stopping cost is a function of the mutual information (and follows directly from Equation (4.26)),
   \[
   \bar{C}(P_k, \alpha, \beta, u^1) = -\alpha^I \log |P^{\bar{l}}_k| + \beta^I \log |P^l_k| + F \left[ \alpha^l \log |P^l_k| - \beta^l \log |P^l_k|; l \neq \bar{l} \right]
   \]
   (4.29)

2. **Target Priority Queue**
   In the case of the macro-manager specifying a target priority queue, the radar is devoting all of it’s resources to target $\bar{l}$ and none of its resources to targets $l \neq \bar{l}$, that is, $v^l = 1$ and $v^l = 0$ for all $l \neq \bar{l}$. Since we are only filtering target $\bar{l}$ and predicting all other targets $l \neq \bar{l}$, the following holds,
   \[
   P^l_k = P^l_k \text{ for all } l \neq \bar{l}
   \]
   (4.30)
   and the mutual information in Equation (4.24) is zero. As a result, we set $\alpha^l = 0$ for all $l$, and consider the difference in differential entropy as the stopping cost for the case of the target priority queue,
   \[
   \bar{C}(P_k, u^1) = -I \left( P^l_k, 0, \beta^l \right) + F \left[ I \left( P^l_k, 0, \beta^l \right); l \neq \bar{l} \right]
   \]
   (4.31)
\[ \beta^l \log |P_k| - F \left[ \beta^l \log |P_k|; l \neq \bar{l} \right] \]  

(4.32)

where 0 is a vector of zeros of length \( L - 1 \). The above expression is of the same form (up to constants) to the difference in the marginal differential entropy and the conditional differential entropy.

\[ \bar{C}(P_k, u^l) = H(P_k^l) - F \left[ H(P_k^l); l \neq \bar{l} \right] \]  

(4.33)

\[ = 2 \log \left( 2\pi e |P_k^l| \right) - F \left[ 2 \log \left( 2\pi e |P_k^l| \right); l \neq \bar{l} \right] \]  

(4.34)

\[ = 2 \log (2\pi e) + 2 \log |P_k^l| - F \left[ 2 \log (2\pi e) + 2 \log |P_k^l|; l \neq \bar{l} \right] \]  

(4.35)

The subsequent optimization problem involving the aforementioned stopping cost does not affect by the presence of additive constants, thus we can safely ignore these and denote the stopping cost as Equation (4.32).

The precise choice of the function \( F \) can vary depending on the scenario as long as the monotonicity property is preserved. The choice of \( F \) will affect the behaviour of the micro-manager scheduler. A variety of forms are,

- **Cumulative Function:** \( F \left[ (-); l \neq \bar{l} \right] = \frac{1}{L-1} \sum_{l \neq \bar{l}} \{ (-) \} \)

- **Maximum Function:** \( F \left[ (-); l \neq \bar{l} \right] = \max_{l \neq \bar{l}} \{ (-) \} \)

- **Minimum Function:** \( F \left[ (-); l \neq \bar{l} \right] = \min_{l \neq \bar{l}} \{ (-) \} \)

The optimization problem, termed the **optimal stopping time problem** is now formally presented.

### 4.2.5 Objective: Optimal Stopping Time

The purpose of the radar micro-manager is to determine the optimal time to continue tracking with the target prioritization provided by the macro-manager. This can be formulated as a class of problems termed **optimal stopping time problems**. An optimal stopping time problem is any problem where we are able to choose
when to stop a particular process in order to minimize a cost (or equivalently, maximize a reward).

We provide a simple example of an optimal stopping time problem to provide background for the reader. The following example presents a scenario where a seller must choose whether to accept or reject consecutive offers on an asset.

**Example 4.2.3 (The Asset Selling Problem [10]).** Consider a person selling an asset for which they are offered an amount of money, $m_k \in [0, M]$ (independent random variables), at each time period, $k = 1, 2, \ldots, N - 1$. The person’s choices at each period $k$ are “sell,” for which they sell the asset at price $m_k$, and “do not sell,” causing them to wait until the next period for the subsequent offer. It is assumed that the person cannot later accept previously rejected offers, and that they also must accept the offer $m_{N-1}$. Given that the accepted offer will gain a constant interest rate of $r > 0$ until time $k = N$, what is the policy which maximizes the personal profit at the $N^{th}$ period?

The optimal policy in the previous example specifies to accept the offer (action “sell”) if it is greater than the expected revenue discounted to the current time, otherwise “do not sell”. The solution was obtained by rewriting the problem in the form of Bellman’s equation [9] and using standard dynamic programming techniques to calculate the optimal policy. A complete worked solution can be found in [10].

Before discussing the full micro-manager optimal stopping time problem, we first present the following deterministic scenario with a unity probability of detection, $p_d = 1$, to offer some insight into the objective of the micro-manager.

**Example 4.2.4.** For this framework, consider the stopping time problem with a unity probability of detection, $p_d = 1$, and fixed initial error covariance matrices, $P_0 = \hat{P}_0$, $\bar{P}_0 = \hat{P}_0$. The optimal policy aims to determine the optimal time to stop...
tracking using the current allocation, $\tau_d$, which satisfies the following,

$$\tau_d = \arg\min_{\tau \in [0, \tau_{\text{max}}]} \left\{ (\tau - 1)c_V + \tilde{C}(P_\tau, u_1) | P_0 = \hat{P}, \tilde{P}_0 = \hat{P} \right\}$$

(4.36)

where $\tau_{\text{max}}$ is the maximum allowable stopping time imposed to ensure the micro-manager tracking cycle stops in finite time. We define the following quantity,

$$D(P, \tau) = (\tau - 1)c_V + \tilde{C}(P_\tau, u_1)$$

(4.37)

We solve this optimization numerically by plotting $D(P, \tau)$ versus $\tau$ for the fixed initial error covariances. The following plot, Figure 4.3, shows the dependence of $D(P, \tau)$ on $\tau$ for various operating costs, $c_V$.

Figure 4.3: Deterministic sample-path cost, $D(P, \tau)$, versus stopping time $\tau$. 
Analyzing Figure 4.3, we can see that there exists a time at which the accumulated operating cost exceeds the stopping cost, corresponding to the minimum of the respective plot. The optimal deterministic stopping time for each value of $c_\nu$ is achieved at the minimum of the respective plots. The optimal deterministic policy thus assigns the optimal stopping times $\tau_d = (13, 10, 8, 7, 6, 6)$ for $c_\nu = (0.5, 0.6, 0.7, 0.8, 0.9, 1.0)$, respectively.

The previous example illustrates that the optimal policy in the deterministic case is to simply find the time associated with the minimum of a deterministic function. However, in the stochastic case, where $p_d < 1$, noise is introduced, thus causing the given function, $D(P, \tau)$, to not have a well-defined minimum.

One may question why the policy for the stochastic case would not just follow a similar procedure as the one discussed above, but now plot the expectation of the function, $D(P, \tau)$, for each stopping time. There is an inherent loss of performance associated with doing this: we would be averaging $D(P, \tau)$ for each deterministic stopping time, thus removing the stochastic aspect of the optimal policy. That is, the nature of the optimal policy for the stochastic case is that it will not always assign a fixed stopping time given a particular set of initial covariance matrices, but instead, it will prescribe an optimal stopping time that is dependent upon the sequence of measurements up until the stopping time. For further justification of this argument, we formally introduce the stochastic optimal stopping time problem.

Let $\eta$ be a stationary policy which maps the error covariance set, $P_k$, at time $k$ to an action $u_k$,

$$\eta : P_k \rightarrow u_k \in \{u^1, u^2\} \quad \text{(4.38)}$$

where, as defined earlier, $u^1 :$ (stop) and $u^2 :$ (continue). We wish to find the policy that minimizes the sample-path cost, defined as,

$$J_\eta(P) = \mathbb{E}^\eta \left\{ \sum_{k=1}^\tau \bar{C}(P_k, u_k) \big| P_0 = P, \bar{P}_0 = P \right\} \quad \text{(4.39)}$$

$$= \mathbb{E}^\eta \left\{ (\tau - 1)c_\nu + \bar{C}(P_\tau, u^1) \big| P_0 = P, \bar{P}_0 = P \right\} \quad \text{(4.40)}$$
That is,

\[ \eta^\ast(P) = \arg\min_{\eta \in \eta} \{ J_\eta(P) \} \quad (4.41) \]

The policy \( \eta^\ast(P) \) is termed the *optimal policy*.

The typical solution method for dynamic programming problems begins by writing the problem in terms of a set of equations termed Bellman’s dynamic programming equations [9]. Writing the optimal stopping time problem in this form results in the following,

\[
\bar{V}(P) = \min \left\{ \bar{C}(P), c_v + \mathbb{E}_z \left[ V(\mathcal{R}(P^\ell, z^\ell), \mathcal{L}(\bar{P}^\ell), \mathcal{R}(P^\ell, z^\ell), \mathcal{L}(\bar{P}^\ell)) \right] \right\} \quad (4.42)
\]

\[
\eta(P) = \arg \min \left\{ \bar{C}(P), c_v + \mathbb{E}_z \left[ V(\mathcal{R}(P^\ell, z^\ell), \mathcal{L}(\bar{P}^\ell), \mathcal{R}(P^\ell, z^\ell), \mathcal{L}(\bar{P}^\ell)) \right] \right\} \quad (4.43)
\]

where \( l \neq \bar{l} \) indexes the remaining \( L - 1 \) targets. The optimal stopping set is defined by,

\[
S_{\text{stop}} = \{ P : \eta^\ast(P) = u^1 \} \quad (4.44)
\]

It is the goal of the micro-manager to characterize the optimal policy, \( \eta^\ast(P) \), which generates the optimal stopping set, such that the sample-path cost, Equation (4.39), is minimized. Techniques typically used to solve the Bellman equation rely on properties that are unfortunately not satisfied by the above formulation. For example, the state space, defined as the set of error covariance matrices \( P_k \), is uncountably infinite. As a result, in order to solve Equations (4.42) and (4.43), we rely on structural results, formulated in [51] and outlined in the following section.

### 4.3 Monotone Threshold Policy

The main theoretical results behind the form of the optimal stopping time policy are now presented, based on the recent work of Krishnamurthy et al [51]. The main theorem of this section states the existence of an optimal threshold policy, \( \eta^\ast(P) \), which is monotone in the error covariance matrices of the targets, \( P \). Sufficient conditions for this monotone policy are obtained by exploiting monotone properties of the Lyapunov and Riccati filter updates, stopping cost (stochastic observability),
and value function.

Before stating the main theory of the micro-manager scheduler, it is necessary to introduce some notation. We define $\mathcal{M}$ to be the set of all $S \times S$ real, symmetric, positive semi-definite covariance matrices. We say that one matrix, $A \in \mathcal{M}$ dominates another $B \in \mathcal{M}$ in a positive-definite sense if $x^T A x \geq x^T B x$ for all non-zero $x \in \mathbb{R}^{S \times 1}$ denoted as $A \succeq B$. Also, we denote $A \succ B$ if $x^T A x > x^T B x$ for non-zero $x$. Note that this is a partial order, since not all matrices can be ordered with respect to the positive-definite order, thus $[\mathcal{M}, \succeq]$ is a partially ordered set.

The following theorem is the main result of the micro-management scheduler and concerns the structure of the optimal policy $\eta^*(\mathbf{P})$.

**Theorem 4.3.1** (Krishnamurthy et al. [51]). *Given the sequential decision problem with sample-path cost,

$$ J_{\eta}(\mathbf{P}) = \mathbb{E}^\eta\{(\tau - 1)c_\nu + \bar{C}(\mathbf{P}_{\tau}, u^1)|\mathbf{P}_0 = \mathbf{P}, \bar{\mathbf{P}}_0 = \mathbf{P}\} $$

and stopping set,

$$ S_{\text{stop}} = \{\mathbf{P} : \eta^*(\mathbf{P}) = u^1\} $$

with a general stopping cost (stochastic observability) of,

$$ \bar{C}(\mathbf{P}_k, \alpha, \beta, u^1) = -\alpha^l \log |\bar{P}_k^l| + \beta^l \log |P_k^l| + F \left[ \alpha^l \log |\bar{P}_k^l| - \beta^l \log |P_k^l|; l \neq \bar{l} \right] $$

The following results hold,

1. **Target Priority Allocation Vector**: The optimal decision policy $\eta^*(\mathbf{P})$ is increasing in $P^l$, decreasing in $\bar{P}^l$, and decreasing in $\bar{P}^l$ and increasing in $P^l$ for each $l \neq \bar{l}$, on the poset $[\mathcal{M}, \succeq]$.

2. **Target Priority Queue**: The optimal policy $\eta^*(\mathbf{P})$ is increasing in $P^\bar{l}$ and, for each $l \neq \bar{l}$, decreasing in $P^l$ on the poset $[\mathcal{M}, \succeq]$.

*Proof.* See Krishnamurthy et al. [51].
The proof of Theorem 4.3.1 relies on the following structural monotonicity results on the partially ordered set $[M, \succeq]$. The details of how these results are used can be found in the complete proof in [51].

1) **Monotone properties of Lyapunov and Riccati update equations:**
   - Two sets results are necessary, (i) For $P_1 \succ P_2$ then $\mathcal{R}(P_1, z) \succ \mathcal{R}(P_2, z)$ and $\mathcal{L}(P_1) \succ \mathcal{L}(P_2)$ [3]; and (ii) The functions $|\mathcal{R}(P, z)/|P|$ and $|\mathcal{L}(P)|/|P|$ are monotone non-increasing in $P$.

2) **Monotone cost function:**
   - Using the above result (1)(ii), it can be shown that the stopping cost is monotone non-increasing in $P^l, \bar{P}^l$ and monotone non-decreasing in $P^l, \bar{P}^l$ for all $l \neq \bar{l}$.

3) **Monotone policy:**
   - Using properties of supermodularity [90], namely the fact that for supermodular (submodular) functions $f(x, u)$, the arg min$_u \{f(x, u)\}$ is non-increasing (non-decreasing) in argument $x$, it is shown that the optimal policy $\eta^*(P)$ is non-decreasing in $P^l$ and non-increasing in $P^l, l \neq \bar{l}$.

The monotonicity results are shown using induction on the value iteration algorithm. In particular, the value iteration algorithm, a fixed point iteration of the Bellman equation, is applied recursively over the iterations $k = 1, 2, \ldots$. The procedure uses the aforementioned structural results to prove the existence of an optimal monotone threshold policy, $\eta^*(P)$.

### 4.4 Parameterized Policy

Given that we have proved the existence of the optimal policy $\eta^*(P)$ and shown it is monotone, we are able to parameterize and efficiently compute an approximation. The knowledge of the monotone structure of the optimal policy still requires us to define the precise form of the approximation. The two main categories of parameterizations discussed are linear and affine hyperplanes. Since both of these are inherently only approximations to the true policy, the behaviour of the parameterized policy will indeed be suboptimal, the performance of which can be seen in the numerical results presented in Chapter 5.
4.4.1 Parameterizations

There are multiple parameterizations that could be used to approximate the optimal policy. The only restriction is that the parameterized policy follows the conditions stated in Theorem 4.3.1. The following parameterization, Equation (4.45), is an affine hyperplane approximation which, upon observation, can be seen to satisfy the aforementioned conditions of Theorem 4.3.1. Both linear and affine approximations were computed; however, the affine policy approximations offered much better performance than their linear counterparts.

\[
\eta_\theta(P_k) = \begin{cases} 
  u^1 & \text{if } -\theta^T P_k^T \theta^l + \theta^T P_k^T \theta^j + \sum_{l \neq i} \left[ \theta^T P_k^T + \theta^T P_k^T \theta^l \right] \geq 1 \\
  u^2 & \text{otherwise}
\end{cases}
\]

(4.45)

The optimal policy is thus parameterized by the parameter vectors, \(\theta^l, \theta^j\), with \(l = \{1, 2, \ldots, L\}\). We define the complete parameter vector, \(\theta\), as the vertical concatenation of the 2L parameter vectors, such that \(\theta\) is of dimension \(2SL \times 1\).

In order to ensure that the parameterized policy, Equation (4.45), satisfies the conditions in Theorem 4.3.1, we need to specify allowed values for \(\theta\). A necessary and sufficient condition for the parameterized policy \(\eta_\theta(P)\) to be monotone increasing in \(P^l, \bar{P}^l\) and monotone decreasing in \(\bar{P}^l, \bar{P}^l\) for \(l \neq \bar{l}\), is \(\theta \in \Theta = \mathbb{R}^{2SL \times 1}\). Also, to ensure that the optimal parameter vector is unique (to arbitrary scaling), we require that for each \(l\), \(\theta^T \theta^l = 1\) (similarly for \(\theta^j\)). This is guaranteed by parameterizing each according to a spherical coordinate parameterization,

\[
\theta^l = \left[ \cos(\phi^l_1) \sin(\phi^l_1) \cos(\phi^l_2) \sin(\phi^l_2) \cos(\phi^l_3) \sin(\phi^l_3) \sin(\phi^l_4) \sin(\phi^l_4) \right]^T
\]

where \(\phi^l \in \mathbb{R}^{3 \times 1}\). We define \(\phi\) as the vertical concatenation of all \(\phi^l\) and \(\phi^j\), thus \(\phi \in \mathbb{R}^{6L \times 1}\). Stochastic approximation algorithms can now be used to estimate the optimal parameter vector, \(\phi^*\), and in turn, the optimal parameterized policy \(\eta_{\phi}^*(P)\).

4.4.2 Policy Estimation via Stochastic Approximation

There are many methods available to determine the optimal parameter vector. Methods such as gradient search, simulated annealing, random search, genetic algo-
rithms and neural networks, are just a few of the many algorithms out there. The gradient search methods obtain a local minimum of the cost function by moving in a decreasing direction, determined by the gradient. This is a relatively efficient method; however, since the algorithm only finds a local minimum, many initial conditions must be used in order to have a reasonable chance of finding the global minimum. The method of simulated annealing attempts to intentionally move in locally “worse” directions (opposing the gradient) to prevent being caught in local minima. A guaranteed way to ensure that the search does not get trapped in a local minimum, given a large enough search space, is to use a basic random search method. The issue with random search is that the necessary search space grows exponentially as the size of the parameter being estimated increases. That is, for a discrete-valued parameter vector of length 8 where each element could take on 100 possible values, there are $100^8$ possible combinations. For our problem, the parameter vector is continuously valued, not discrete, therefore the search space becomes uncountably large. Adding noise into the problem requires a certain degree of averaging to find the true minimum, thus increasing search time further and making this method rather impractical for our problem. As a result, we decided to pursue a hybrid gradient search and random search method. Genetic algorithms and neural networks are also used for stochastic search and optimization; however, they will not be discussed here. For a good overview of simulation optimization and stochastic approximation algorithms see [23].

Typically it is difficult to come up with a fair metric for measuring the performance of the various stochastic approximation algorithms, thus making it difficult to make comparisons and make the most appropriate choice for the problem at hand. This is due to the fact that due to the stochastic nature of the problem; we cannot obtain an exact value for a given minima, but rather just an estimate. The only way to reduce the variance of this estimate is to run multiple averaging runs, which can be very time consuming, and also leaves the possibility for making false conclusions when comparing different approximation methods if the variance is not sufficiently reduced. Following the reasoning in [88], we measure efficiency of each method through how many function evaluations the algorithm requires. For gradient search algorithms, the best known techniques are perturbation analysis and likelihood ratio/score function methods. We chose a perturbation analysis.
method, simultaneous perturbation stochastic approximation (SPSA) to be exact, since it only requires two evaluations of the cost function (independent of the size of the parameter being estimated), thus making it more efficient than the common Kiefer-Wolfowitz algorithm which requires $2^n$ evaluations, where $n$ is the length of the parameter vector being estimated. The algorithm to determine the approximation to the threshold policy using SPSA is outlined below,

**Algorithm 1: Multi-Linear Threshold Policy Approximation**

**Step 0: Initialization**
Choose highest priority target $l \in \mathcal{L}$.
Define $P^l_0$ and $P^{\bar{l}}_0$ for $l \neq \bar{l}$ initial covariance matrices.
Define SPSA parameters.
Set $n = 1$

**Step 1: Stochastic Optimization (SPSA Loop)**
Compute initial threshold coefficients, $\phi_0$ by sampling from the set $\Phi = \mathbb{R}^{6L \times 1}$.
For $n = 0, 1, 2, \ldots, N_{max}$

Define SPSA parameters, $\varepsilon_n = \frac{\varepsilon}{(1+n)^{\gamma}}$, $\delta_n = \frac{\delta}{(1+n)^{\gamma}}$, $\omega_n(i) = \pm 1$ w.p. 0.5 where $i = \{1, 2, \ldots, SL\}$.
Perturb parameter vector, $\phi_n^\pm = \phi_n \pm \delta_n \omega_n$.
Simulate $J_{\phi_n^+}$ and $J_{\phi_n^-}$,

For $k = 1, 2, \ldots, K_{max}$

Update states accordingly (see Section 4.2.2).
Choose action based on the policy, $u_{k+1}^+ = \eta_{\phi_n^+}(P_{k+1})$.
If $u_{k+1}^+ = stop$, then $J_{\phi_n^+} = \sum_{m=1}^{k+1} \tilde{C}(P_{k+1}, u_m^+)$.
If $u_{k+1}^- = stop$, then $J_{\phi_n^-} = \sum_{m=1}^{k+1} \tilde{C}(P_{k+1}, u_m^-)$.
End For;
End Simulate;
Compute gradient estimate, $\hat{\nabla}_\phi J = \frac{J_{\phi_n^+} - J_{\phi_n^-}}{2\mu_0 \omega_n}$.
Update parameter vector, $\phi_{n+1} = \phi_n - \varepsilon_n \hat{\nabla}_\phi J$.
End For;
**Step 2: Termination**

The parameter vector corresponding to the best multi-linear approximation to the threshold policy is given by, $\phi^* = \arg\min_{\phi \in \Phi} J_\phi = \phi_{N_{\text{max}}}$. In practice there may be differences between the true (no noise) minimum and the terminal estimate provided by the algorithm. The above only holds if $N_{\text{max}}$ is sufficiently large, of which is typically impractical for most applications. Thus the above procedure should be monitored by a human operator to ensure convergence to a (possibly local) minimum.

An example convergence plot of the SPSA algorithm can be seen in Figure 4.4,

![Figure 4.4: Convergence of the SPSA algorithm to a minimum of the objective function.](image)

**Figure 4.4:** Convergence of the SPSA algorithm to a minimum of the objective function.
Chapter 5

Numerical Results

In this section, we present numerical results to display the performance of the radar manager. We first generate scenarios for the radar macro-manager and evaluate the performance of the computed optimal order. Then, we analyze the performance of the micro-manager by evaluating its performance in two common radar applications, the target fly-by problem and the persistent surveillance problem. Comparisons to other algorithms are made in both the macro-manager and micro-manager simulation sections.1

Below is the general protocol outlining the operating and interaction of the radar manager. The simulations are based on the following operation.

**Protocol 1: Dual Time-scale Radar Manager Operation**

**Step 0: Initialization**
Define asset locations, \((x_1^p, y_1^p), (x_2^p, y_2^p), \ldots, (x_A^p, y_A^p)\).
Construct graph \(G = \{V, W\}\) and compute vulnerability weights, \(\bar{w}_i\) (Section 3.3.2).

**Step 1: Target Prioritization (Macro-manager)**

Compute threat estimate, \(\hat{r}_n^l\), of each target.

---

1All simulations were carried out for \(L = 4\) targets.
Compute threat estimate error, \( e_n^l \), of each target.
Assign priority distributions, \( \hat{X}_n^l \), from least-squares binomial parameters, \( p_n^{l*} \).
Order binomial parameters to construct optimal observation queue, \( o^* \).
Compute maximum tracking time vector, \( t_n \).

**Step 2: Target Scheduling (Micro-manager)**

For \( i = 1, 2, ..., L \),

Observe \( i^{th} \) target in observation queue \( o^* \), and set \( k = 1 \).
While \( k < t_n(i) \),

Update covariance matrices of each target accordingly (Section 4.2.2).
Compute action \( u_{k+1} = \eta^* (P_{k+1}) \).
If \( u_{k+1} = \text{stop} \)

break;
End If;
Increment fast time step, \( k = k + 1 \).
End While;
End For;
Increment slow time step, \( n = n + 1 \), and return to Step 1.

### 5.1 Macro-manager Numerical Results

The performance of the macro-manager is now analyzed. Recall that the purpose of the macro-manager was to determine a target prioritization, specifically an optimal target observation queue, based on the target state estimates, targets’ error covariances, and asset locations. To evaluate the performance of the macro-manager, we compute the optimal order associated with various sets of initial target scenarios and compare the optimal order’s performance with the performance of all other possible observation orders\(^2\) on each of the initial target scenarios. We assume \( M \) sets of initial conditions, indexed by \( m \). This process allows us to illustrate the

\(^2\)For a system with \( L \) targets, this translates into \( L! \) possible orders.
flexibility of the application of the optimal order policy. For simplicity, each one of the scenarios is assumed to have the same fixed asset locations.

The initial target conditions used for the following simulations were generated as follows. The true target positions, $x_0^1, x_0^2, \ldots, x_0^L$ and target velocities, $\dot{x}_0^1, \dot{x}_0^2, \ldots, \dot{x}_0^L$, were generated by sampling from the uniform distributions $U(-100, 100)$ and $U(-4, 4)$, respectively. For simulation purposes (to construct the initial error covariance matrices) it was necessary to also generate initial target state estimates. These estimates were generated by the inclusion of an additive Gaussian noise term. The initial position estimates were computed as, $\hat{x}_l^0 = x_l^0 + w_x^l$ and velocity estimates, $\dot{\hat{x}}_l^0 = \dot{x}_l^0 + w_{\dot{x}}^l$, where $w_x^l \sim N(0, 10^2)$ and $w_{\dot{x}}^l \sim N(0, 2^2)$ for all $l$. Similarly, the true states and state estimates for the $y$ component were generated using the same parameters. The true target state vectors and state estimate vectors were then constructed as $s_l^0 = [x_l^0, \dot{x}_l^0, y_l^0, \dot{y}_l^0]^T$ and $\hat{s}_l^0 = [\hat{x}_l^0, \dot{\hat{x}}_l^0, y_l^0, \dot{\dot{y}}_l^0]^T$, allowing the initial error covariance matrix to be computed, $P_0^l = (s_l^0 - \hat{s}_l^0)(s_l^0 - \hat{s}_l^0)^T$.

The optimal order was then computed following the procedure outline stated in Step 1 of Protocol 1. The stopping time of each target observation was considered to be a random time, $\tau$, sampled from the uniform distribution $U(\underline{\tau}, \bar{\tau})$ where $\underline{\tau} = 5$ and $\bar{\tau} = 10$ iterations, which corresponds to random stopping time (at $T = 0.1s$) of 0.5–1 seconds. The following plot, Figure 5.1, presents a comparison of the average reward associated with processing each queue item using various policies. The reward for each policy (order) is computed over all initial target states, $m \in M$, and averaged. Notice that the average reward of the optimal policy dominates all other policy rewards for each target observation. Practically, this is of interest since if the user decides to terminate the queue before completion, the optimal policy will have achieved the highest reward (out of all policies) for the targets already observed.

Figure 5.1 shows the average reward over all initial target states, but it would be of interest to also analyze the performance of the optimal order for each $m \in M$. Figure 5.2 shows the reward associated with the optimal order for each set of initial target states. Notice that the optimal policy achieves the maximum reward for nearly all of the initial conditions. The few cases where an alternate order results in a higher reward is due to the random motion of the target resulting in a different (than the initial) priority rank realization by the time the corresponding target is
Figure 5.1: Average gain in reward for each target observation using various ordering policies.

Figure 5.2: Performance of optimal policy for each initial target set, \( m \).
observed.

The performance of the optimal order policy is dependent upon how close the actual threat rank realization (by the time the target is observed) is to the initial threat ranks. This is due to the fact that the computation of the optimal order is based solely on the initial estimates of the target states. The performance would then be optimal if the targets were assumed to be static over a macro-management interval, \( n \) (since in this scenario, the threat rank realizations will be equivalent to the initial random variables used to compute the optimal order). In reality, the targets are not fixed in time; however, at the time-scale at which the macro-manager operates and the nature of the (slow moving) targets, the states of the targets do not change at a rate such that the updated order (if it were computed at the time the target was observed) would be different than the initial optimal order.

5.2 Micro-manager Numerical Results.

To illustrate the micro-management algorithms proposed in this paper, we consider two important GMTI surveillance problems – the target fly-by problem and the persistent surveillance problem.³

5.2.1 Target Fly-by

We consider 4 ground targets that are tracked by a GMTI platform, as seen in Figure 5.3. The nominal range from the GMTI sensor to the target region is approximately \( \tilde{r} = 30\text{km} \). For this example, the initial (at the start of the micro-manager cycle) estimated and true target states of the four targets are:

\[
\begin{align*}
\hat{s}_0^1 &= [130, 5.5, 84, 8.1]^T, \\
\hat{s}_0^2 &= [-47.88, -2.38, 210.41, 0.418]^T, \\
\hat{s}_0^3 &= [55.84, 2.37, 121.74, 9.56]^T, \\
\hat{s}_0^4 &= [-55.13, 5.75, -68.41, -6.10]^T,
\end{align*}
\]

\[
\begin{align*}
s_0^1 &= [100, 3, 40, 7]^T, \\
s_0^2 &= [-20, -4, 200, 1]^T, \\
s_0^3 &= [50, 2, 95, 10]^T, \\
s_0^4 &= [-70, 5, -50, -6]^T.
\end{align*}
\]

Since we are assuming a constant platform height, and since the range to our target region may change slightly over the tracking interval, it is assumed that our

³ A version of this section can be found in [51].
depression angle, \( \theta_d \), and squint angle, \( \psi \), will change with time in order to keep the region of interest illuminated.

We assume in this example that the most uncertain target is regarded as being the highest priority. Based on the previous state estimates, the mean square error values are, \( \text{MSE}(\hat{s}_{0}^{1}) = 710.87 \), \( \text{MSE}(\hat{s}_{0}^{2}) = 222.16 \), \( \text{MSE}(\hat{s}_{0}^{3}) = 187.37 \), and \( \text{MSE}(\hat{s}_{0}^{4}) = 140.15 \). Thus, target \( l = 1 \) is the most uncertain, and we denote \( \bar{l} = 1 \).

The simulation parameters are as follows: number of targets \( L = 4 \); sampling time \( T = 0.1 \text{s} \); probability of detection \( p_d = 0.75 \); track standard deviations of target model \( \sigma_x = \sigma_y = 0.5 \text{m} \); measurement noise parameters \( \sigma_r = 20 \text{m}, \sigma_a = 0.5^\circ, \sigma_{\dot{r}} = 5 \text{m/s} \); and platform states \( [p_x, \dot{p}_x, p_y, \dot{p}_y] = [10 \text{km}, 53 \text{m/s}, -30 \text{km}, 85 \text{m/s}] \). We assume a target priority vector of \( \nu = [\nu_1, \nu_2, \nu_3, \nu_4] = [0.6, 0.39, 0.008, 0.002] \).

Recall from Section 2.2 that the target priority determines the covariance of the observation noise. We chose an operating cost of \( c_\nu = 0.8 \), and a stopping cost as in Equation (4.29) (with cumulative function \( F() \) as discussed in Section 4.2.4) with \( \alpha^{1,...,4} = \beta^{2,...,4} = 0.05, \beta^1 = 5 \). The parameterized policy chosen for this example was that of Equation (4.45).

Figure 5.4 illustrates the performance of the micro-manager when tracking the four targets using the optimal parameterized policy. Notice that the state estimates of the current target, \( l = 1 \), settle near the true track near the end of the tracking cycle. Meanwhile, the estimates for the other three targets start to drift off their
Figure 5.4: Figure showing one cycle of micro-manager with accuracy of state estimates for an arbitrary priority vector. True tracks (solid line) with track estimates (dashed line) and error ellipses. An operating cost of $c_v = 0.8$ and stopping cost as in Equation (4.29) were used. Dots represent the start of the track.
respective true tracks near the end of the tracking cycle. We found that the average stopping time for given cost depends largely on the initial uncertainties in the states of each target. In general, the higher the uncertainty for the high priority target, the longer it will take before control is returned to the macro-manager.

We now explore the sensitivity of the sample-path cost (achieved with the parameterized policy) on various system parameters. The two parameters which are varied are the probability of detection, \( p_d \), and the operating cost, \( c_\nu \); the effects of which can be seen in Figure 5.5. The sample-path cost increases with the operating cost and decreases with the probability of detection. Notice also that the plot is annotated with the corresponding average stopping times achieved with the parameterized policy. Higher values of the operating cost, \( c_\nu \), cause the micro-manager to specify the “stop” action sooner than for lower values of \( c_\nu \). As can be seen in the figure, neither the sample-path cost or the average stopping time is very sensitive to changes in the probability of detection. However, varying the operating cost has a large affect on both the sample-path cost and the associated average stopping time.

The performance of the optimal parameterized policy is measured using multiple initial conditions. As seen in Figure 5.6, the optimal parameterized policy achieves the best periodic policy, for any initial condition. That is, the optimal parameterized policy is the lower envelope of all possible periodic stopping times, for each initial condition.

The optimal period for the periodic policy is highly dependent upon the initial condition. This is illustrated in Figure 5.6(b). Notice that for various initial conditions, the optimal stopping time changes frequently. The main performance advantage of the optimal parameterized policy is that it achieves virtually the same cost as the optimal periodic policy, but for any initial condition. Note that the slight vertical shift of the cost associated with the parameterized policy is likely due to the loss in performance associated with approximating the monotone threshold curve by an affine function.
Figure 5.5: Surface plot representing dependence of the sample-path cost (achieved with by the parameterized policy) on the probability of detection, $p_d$, and the operating cost, $c_\nu$. The sample-path cost increases in the operating cost, but decreases in the probability of detection. Note the stopping times associated with the labelled vertices above.

5.2.2 Persistent Surveillance

Persistent surveillance involves exhaustive surveillance of a region over long time intervals, typically over the period of several hours or weeks [77]. This has recently received much attention in the defense literature since it has the potential to provide critical, long-term battlefield information. For example, by tracking targets for long periods of time using aerial based radars, such as DRDC-Ottawa’s XWEAR radar [7] or the U.S. Air Force’s new Gorgon Stare Wide Area Airborne Surveillance System, operators can “rewind the tapes” in order to determine the origin of any target of interest [98]. This paper is concerned with achieving the best estimates
Figure 5.6: Plot of sample-path cost of variance periodic policies and the parameterized policy (thick-dashed line) versus ordered initial conditions. Initial conditions are ordered with respect to the cost achieved using the parameterized policy for that particular initial condition. Notice that the parameterized policy performs just as well as the best deterministic stopping time for any particular initial condition, illustrating the versatility of the parameterized policy.

during the entire tracking period.

Consider Fig. 5.7 illustrating the persistent surveillance setup. Here $\tilde{r}$ is the nominal range from the target region to the GMTI platform, assumed for our simulations to be approximately 30km. The points on the GMTI track labeled (1) – (72) correspond to locations\(^4\) where we evaluate the Jacobian (3.23). Assume a constant platform orbit speed of 250m/s (or approximately 900km/h [91]) and a constant altitude of approximately 5000m. Assuming 72 divisions along the 30km radius orbit, the platform sensor takes 10.4 seconds to travel between the track segments. Using a similar analysis as in the Appendix, the measurement model changes less than 5% in $l_2$-norm in 10.4s, thus the optimal parameter vector is approximately constant on each track segment.

Simulation parameters for this example are as follows: number of targets $L = 4$; sampling time $T = 0.1s$; probability of detection $p_d = 0.9$; track standard deviations of target model $\sigma_x = \sigma_y = 0.5m$; and measurement noise parameters $\sigma_r = 20m$, $\sigma_a = 0.5^\circ$, $\sigma_t = 5m/s$. The platform velocity is now changing (assume a con-

\(^4\)The platform state at location $n \in \{1, 2, \ldots, 72\}$ is defined as $p = [\tilde{r}\cos(n \cdot 5^\circ), -\tilde{v}\sin(n \cdot 5^\circ), \tilde{r}\sin(n \cdot 5^\circ), \tilde{v}\cos(n \cdot 5^\circ)]$
Figure 5.7: Representation of the persistent surveillance scenario in GMTI systems. The GMTI platform (aircraft) orbits the target region in order to obtain persistent measurements as long as targets remain withing the target region. The nominal range from the platform to the target region is assumed to be 30km.

stant speed of $\tilde{v} = 250\text{m/s}$), unlike in the previous example, where only the position of the platform was changing. Since the linearized model will be quite different at each of the pre-specified points, (1)–(72), along the GMTI track, we need to pre-compute the optimal threshold policy at each of the respective locations. We then switch between the optimal strategies when we reach that location on the track.

We consider the special case where the radar devotes all its resources to one target, and none to the other targets. That is, we assume a target priority vector of $\nu = [1, 0, 0, 0]$. In this case, the first target is allocated a Kalman filter, with all the other targets allocated Kalman predictors. Since the threshold parametrization vectors depend on the target’s state and measurement models, the first target $l = \bar{l}$ has a unique parameter vector, where targets $l \neq \bar{l}$ all have the same parameter vectors. Also, $\theta^l = \theta$, for all $l \in \{1, 2, \ldots, L\}$.

The choice of the operating cost $c_\nu$ should be lower in the persistent surveillance case than in the fly-by example. The intuition behind this is due to the
fact that in the fly-by example, where we defined a target priority vector of \( \nu = [0.6, 0.39, 0.008, 0.002] \), the radar is switching between the targets to satisfy the tracking time ratio (on the fast time scale, \( t \)) defined in \( \nu \). Thus the operating cost per iteration \( k \) should be lower in the persistent surveillance case where we assumed to be looking at only one target (\( \nu = [1, 0, 0, 0] \)). A value of \( c_\nu = 0.1 < 0.8 \) satisfies this requirement. The values of \( \alpha \) and \( \beta \) can be chosen to modify the general behaviour of the scheduler. The values chosen in this example were \( \alpha^1 = 0.25, \alpha^2 \ldots ^4 = 0, \beta^1 = 0.25, \beta^2 \ldots ^4 = 1 \) with Equation (4.45) as the parameterized policy and Equation (4.29) (with cumulative function \( F(\)\)) as the stopping cost.

The optimal parameterized policy was computed using Algorithm 4.4.2 at each of the 72 locations on the GMTI sensor track. As the GMTI platform orbits the target region, we switch between these parameterized policy vectors, thus continually changing the adopted tracking policy. For the purposes of these simulations (illustrating the effects of the micro-manager) we chose a simple macro-manager as follows,

\[
\bar{l} = \arg \max_{l=1, \ldots, L} \left\{ \log |P^l| \right\}
\]  

(5.1)

We then assign the priority vector as \( \nu^{\bar{l}} = 1 \) and \( \nu^l = 0 \) for all \( l \neq \bar{l} \). Figure 5.8 shows track estimates over multiple macro-management scheduling intervals while the scheduler is using the policy associated with the second of the 72 locations in Figure 5.7. The corresponding values of the log-determinants of each of the targets’ error covariance matrices is shown in Figure 5.9.
Figure 5.8: Track estimates in persistent surveillance case over multiple micro-manager scheduling intervals, with true tracks (solid line) and track estimates (starred line). The red portion of each target's track estimate corresponds to when the radar is filtering that target, where the cyan portion is where the state is being predicted. Dots represent the start of the track.
Figure 5.9: Plot of log-determinants for each target over multiple micro-manager scheduling intervals. Bold sections of each line correspond to where the radar is filtering the target (radar initially filtering target $l = 1$), whereas normal-weighted lines correspond to the target states being predicted. Data points marked in red indicate that the observation was missed.
Chapter 6

Conclusions and Future Work

This section summarizes the main points of the document and also suggests logical extensions and generalizations of the ideas discussed in the dissertation.

6.1 Summary of Work Accomplished

This dissertation analyzed the problem of tracking $L$ targets with a single multifunction radar with the goal of optimally allocating the radar’s resources. The problem was divided into two main components, the radar macro-manager, responsible for assigning priorities to tasks, and the radar micro-manager, responsible for scheduling each of the specified tasks in time. The main result of the macro-manager was the application of structural results from the field of optimal issuing. The structural results were used to determine an optimal observation queue to maximize a reward proportional to the incentive for observing each target. In order to compute the optimal queue, the threat of each target was computed based on multiple factors: 1) The target’s proximity and heading relative to fixed assets; 2) The error in the state estimate of each target; and 3) The computed vulnerability of each asset. A constraint on the maximum tracking time for each target is also computed based on the relative priority ranks. The specified observation queue and maximum tracking time vector are then passed to the micro-manager. The micro-manager determines the optimal time to continue tracking each target, subject to the maximum tracking time constraint. Sufficient conditions were presented such
that the optimal stopping time policy takes a monotone threshold form. Due to the monotone structure, an efficient numerical algorithm, a hybrid random search/gradient search method, was constructed to compute the approximation to the policy. Detailed numerical results were presented to evaluate the performance of both the macro-manager and micro-manager. We considered two realistic GMTI surveillance problems – the target fly-by problem and the persistent surveillance problem.

6.2 Future Work

1. Region-based Scheduler: The work presented in this dissertation provided the framework for a dual time-scale radar manager consisting of both task prioritization and task scheduling components for individual targets. An interesting extension to the work presented here would be to formulate the task prioritization and scheduling problems on a region basis. This would involve reformulating the macro-manager such that the priority ranks represent the priority of a particular region.

2. Minimization of Tracking Makespan: One interesting extension to the radar resource management problem would be to optimize the tracking makespan. The tracking makespan is defined as the total time taken to observe a queue of $L$ targets, where the tracking time for each target is specified by the radar micro-manager. However, given that the goal is to minimize the error in the level of threat of each target, minimizing tracking makespan would only minimize threat error if the state estimate error was proportional to the target’s threat estimation error. That is, the higher the uncertainty in the target, the higher the level of threat in that target. However, this is not the case in general. Consider Figure 3.5 for a visual reference to the following argument. Consider a specific class of targets shown in the figure, namely weakly maneuvering targets. In the case where a target is moving away from an asset (as is the case for target $l = 3$), even though the state estimate is diverging from the true track, the threat error approaches zero. This makes intuitive sense as both the estimated target state and the true target state do not represent threats given their respective trajectories, albeit being different. However, if the target is moving towards an asset or if the
targets are considered to be highly maneuvering, the relationship between a target’s state estimate and its threat estimate may be complex. It is possible to determine this relationship by coming up with a *predicted* threat (and corresponding predicted threat error) based on the target’s current behaviour. It would be interesting to take advantage of the predicted threat error and use this information to improve the performance of the macro-manager scheduling algorithm. This would require a generalization of the theory presented in the literature (regarding scheduling of jobs with wait-dependent processing times) to allow for nonlinear target-dependent degradation functions, since each target’s predicted threat error is dependent upon not only the target, but also its location to various assets in the environment. As of now, there does not exist any theory that can handle this scenario.

3. **Precedence Constraints:** In some tracking scenarios, it may be useful to perform tasks of a certain type before certain other tasks. A useful example of this in the context of radar tracking would be the desire to observe a particular region using a low resolution tracking mode (to detect a target), before using a higher-resolution, more expensive mode to track any targets in that particular region. The work of Garey [24] considers task scheduling problems with so-called *precedence constraints*. It would be interesting to apply the results of this topic to the radar tracking problem.
Bibliography


Template-based real-time dwell scheduling with energy constraint. In 


[86] B. Sinopoli, L. Schenato, M. Franceschetti, K. Poolla, M. Jordan, and 
S. Sastry. Kalman filtering with intermittent observations. *Automatic 


*Phased Array Systems and Technology, IEEE International Symposium on*, 


*Perspectives on Radio Astronomy: Technologies for Large Antenna Arrays*, 

[93] G. van Keuk and S. Blackman. On phased-array radar tracking and 
parameter control. *Aerospace and Electronic Systems, IEEE Transactions 

*Proceedings of the 33rd Midwest Symposium on Circuits and Systems*, 
1990.


Appendix A

Appendix: Tables

Table A.1: Rate of change of Jacobian matrix for various running times.

<table>
<thead>
<tr>
<th></th>
<th>$D(\bar{s}<em>{10}, \bar{\xi}</em>{10})$</th>
<th>$D(\bar{s}<em>{50}, \bar{\xi}</em>{50})$</th>
<th>$D(\bar{s}<em>{100}, \bar{\xi}</em>{100})$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s_0^a$</td>
<td>0.0010</td>
<td>0.0052</td>
<td>0.0104</td>
</tr>
<tr>
<td>$s_0^b$</td>
<td>0.0009</td>
<td>0.0049</td>
<td>0.0104</td>
</tr>
<tr>
<td>$s_0^c$</td>
<td>0.0010</td>
<td>0.0059</td>
<td>0.0119</td>
</tr>
<tr>
<td>$s_0^d$</td>
<td>0.0007</td>
<td>0.0049</td>
<td>0.0080</td>
</tr>
<tr>
<td>$s_0^e$</td>
<td>0.0010</td>
<td>0.0053</td>
<td>0.0122</td>
</tr>
</tbody>
</table>

Table A.2: Ratio of second-order term to first-order term of Taylor series expansion, $\zeta = 0.1$.

<table>
<thead>
<tr>
<th></th>
<th>$E(s_{10}, \bar{s}<em>{10}, \bar{\xi}</em>{10}, 0.1)$</th>
<th>$E(s_{50}, \bar{s}<em>{50}, \bar{\xi}</em>{50}, 0.1)$</th>
<th>$E(s_{100}, \bar{s}<em>{100}, \bar{\xi}</em>{100}, 0.1)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s_0^a$</td>
<td>0.00019091</td>
<td>0.0010597</td>
<td>0.01395</td>
</tr>
<tr>
<td>$s_0^b$</td>
<td>0.00020866</td>
<td>0.0011699</td>
<td>0.014375</td>
</tr>
<tr>
<td>$s_0^c$</td>
<td>0.00019165</td>
<td>0.0010813</td>
<td>0.01453</td>
</tr>
<tr>
<td>$s_0^d$</td>
<td>0.0002008</td>
<td>0.0011065</td>
<td>0.011844</td>
</tr>
<tr>
<td>$s_0^e$</td>
<td>0.0002294</td>
<td>0.0012735</td>
<td>0.015946</td>
</tr>
</tbody>
</table>
Table A.3: Ratio of second-order term to first-order term of Taylor series expansion, $\zeta = 0.8$.

<table>
<thead>
<tr>
<th></th>
<th>$E(s_{10}, \bar{s}<em>{10}, \xi</em>{10}, 0.8)$</th>
<th>$E(s_{50}, \bar{s}<em>{50}, \xi</em>{50}, 0.8)$</th>
<th>$E(s_{100}, \bar{s}<em>{100}, \xi</em>{100}, 0.8)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s_0^a$</td>
<td>0.00019267</td>
<td>0.0011104</td>
<td>0.014211</td>
</tr>
<tr>
<td>$s_0^b$</td>
<td>0.00021104</td>
<td>0.0012164</td>
<td>0.014633</td>
</tr>
<tr>
<td>$s_0^c$</td>
<td>0.00019447</td>
<td>0.0011228</td>
<td>0.014838</td>
</tr>
<tr>
<td>$s_0^d$</td>
<td>0.00020148</td>
<td>0.0011386</td>
<td>0.012178</td>
</tr>
<tr>
<td>$s_0^e$</td>
<td>0.00023083</td>
<td>0.0013596</td>
<td>0.016603</td>
</tr>
</tbody>
</table>
Appendix B

Appendix: Derivations

B.1 Target Model Derivation

This section derives the discrete white noise acceleration target model, referring to Blackman and Popoli [11] and Bar-Shalom [8]. There are multiple ways to derive the following model; however, the following direct derivation in the discrete-time domain is simplest.

It is common to use a constant velocity model to represent the behaviour of targets. In this case, it would be logical to set the acceleration equal to zero, then apply integration to obtain expressions for the velocity and position of the target. However, in practice, targets do not have a constant velocity, but rather undergo small deviations. Thus we equate the acceleration of a target to white noise instead, accounting for the small changes in velocity,

\[
a_k = w_k
\]  

(B.1)

where the white noise process is characterized by the following,

\[
E[w_k] = 0
\]  

(B.2)

\[
E[w_kw_l] = \sigma_w^2 \delta_{kl}
\]  

(B.3)

where \(\delta_{kl}\) is the Kronecker delta function.
For the purposes of this derivation, we assume motion only along one dimension. Since we have assumed that the $x$ and $y$ dynamics of the target are independent of each other, the generalization to two dimensions is straightforward. Recall the form of the state equation,

$$s_{k+1}^l = F_1 s_k^l + G_1 w_k^l \quad \text{(B.4)}$$

where $F_1$ is the state transition matrix (for one dimension), and $G_1$ is the gain vector. From elementary physics, we know that a particle moving at a velocity $\dot{x}_k$ for a time $T$ will travel a distance of $\int_0^T \dot{x}_k \, dt = T x_k$. This justifies the form of the state transition matrix defined below. Recall from Equation (B.1), the acceleration in the $k^{th}$ iteration is constant (see Figure B.1).

**Figure B.1:** Acceleration during the $k^{th}$ iteration of length $T$.

Thus, the change in displacement for the target is $w_k T^2/2$ with a gain in velocity of $w_k T$. Thus the state matrix, $F_1$, and the gain matrix, $G_1$, are defined as follows,

$$F_1 = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix}, \quad G_1 = \begin{bmatrix} T^2/2 \\ T \end{bmatrix} \quad \text{(B.5)}$$

Similar to before, the covariance matrix of the process noise is defined as,

$$Q = E\{G_1 w_k w_k^T G_1^T\} = G_1 E\{w_k w_k^T\} G_1^T = G_1 \sigma_w^2 G_1^T = \begin{bmatrix} \frac{1}{4} T^4 \sigma_w^2 & \frac{1}{2} T^3 \sigma_w^2 \\ \frac{1}{2} T^3 \sigma_w^2 & T^2 \sigma_w^2 \end{bmatrix} \quad \text{(B.6)}$$

Extending this to two dimensions is straightforward. As mentioned earlier, we assume the filtering across coordinate is decoupled [8]. Also, since each component of the target’s motion is assumed to be governed by the same model, the ma-
trices describing the multi-dimensional system are just block-diagonal repetitions of the matrices in the one dimensional case. That is,

\[ F = \begin{bmatrix}
1 & T & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & T \\
0 & 0 & 0 & 1
\end{bmatrix}, \]  

(B.7)

with process noise covariance matrix,

\[ Q = \begin{bmatrix}
\frac{1}{4}T^4\sigma_x^2 & \frac{1}{2}T^3\sigma_x^2 & 0 & 0 \\
\frac{1}{2}T^3\sigma_x^2 & T^2\sigma_x^2 & 0 & 0 \\
0 & 0 & \frac{1}{4}T^4\sigma_y^2 & \frac{1}{2}T^3\sigma_y^2 \\
0 & 0 & \frac{1}{2}T^3\sigma_y^2 & T^2\sigma_y^2
\end{bmatrix}

\]

B.2 Approximation and Justification of Linear Gaussian State Space Model

In order to apply Kalman filtering techniques, we require our state and observation equations to follow a LGSSM. A LGSSM requires that the state and measurement matrix be linear and time-invariant and the noise terms to be zero-mean, Gaussian random variables. Since the model discussed in Section 2.2 is clearly not linear or time-invariant (due to the measurement process), it is necessary to approximate the nonlinear, time-varying model by a linear, time-invariant model. We then analyze how accurate this approximation is for typical tracking scenarios.

The first step is to linearize the model. We follow Jazwinski [40] and linearize around a deterministic target trajectory, \( \tilde{s}_{k+1} = F\tilde{s}_k \), and nominal measurement, \( \tilde{z}_k = h(\tilde{s}_k, \xi_k) \). Let \( \tilde{s}_k = s_k - \bar{s}_k \) and \( \tilde{z}_k = z_k - \bar{z}_k \), the first-order Taylor series expansion around the nominal trajectory results in the following model,

\[ \tilde{s}_{k+1} = F\tilde{s}_k + Gw_k \]  

(B.8)
\[ \dot{z}_k^j = \begin{cases} \nabla h(s_k^j, \xi_k) \dot{s}_k^j + R_1(s_k^j, \xi_k, \gamma) + \frac{1}{\sqrt{\nu \Delta}} \dot{v}_k^j & \text{with probability } p_d^j \\ 0 & \text{with probability } 1 - p_d^j \end{cases} \] 

with probability \( p_d^j \). 

(B.9)

where \( R_1(s_k^j, \dot{s}_k^j, \xi_k, \gamma) \) represents the linearization error and is bounded as \( \| R_1(s_k^j, \dot{s}_k^j, \xi_k, \gamma) \| \leq \frac{1}{2} \| (s_k^j - \dot{s}_k^j)^T \nabla^2 h(\xi, \gamma) (s_k^j - \dot{s}_k^j) \| \), with \( \xi = \gamma s_k^j + (1 - \gamma) \dot{s}_k^j \) for some \( \gamma \in [0, 1] \). This expression is dependent upon the Hessian tensor, \( \nabla^2 h(\xi_k, \zeta_k) \in \mathbb{R}^{3 \times 4 \times 4} \). The first term in Equation (2.2) corresponds to the Jacobian, \( \nabla_s h(s_k^j, \xi_k) \) defined as (with indices \( k \) and \( l \) omitted for simplicity),

\[ \nabla_s h(s, \xi) = \frac{\partial h_i(s, \xi)}{\partial s_l} = \begin{bmatrix} \frac{\delta_i}{r} & 0 & \frac{\delta_i}{r} & 0 \\ \frac{\delta_i}{r} & 0 & \frac{\delta_i}{r} & 0 \\ \frac{\delta_i}{r} & \frac{\delta_i \delta_k + \delta_i \delta_x}{r^2} & \frac{\delta_i}{r} & \frac{\delta_i \delta_k + \delta_i \delta_x}{r^2} \\ \frac{\delta_i}{r} & \frac{\delta_i \delta_k + \delta_i \delta_x}{r^2} & \frac{\delta_i}{r} & \frac{\delta_i \delta_k + \delta_i \delta_x}{r^2} \end{bmatrix}, \quad r = \sqrt{\delta_x^2 + \delta_y^2 + \xi_z^2} \]

(B.10)

with relative coordinates defined as \( \delta_x = x - \xi_x, \delta_x = y - \xi_y \) and similarly for the relative velocities \( \dot{\delta}_x \) and \( \dot{\delta}_y \).

We seek to evaluate the accuracy of the linear and time-invariant approximation. First, consider the following quantities,

\[ E(s_k, \tilde{s}_k, \xi_k, \zeta) = \frac{1}{2} \left\| (s_k - \tilde{s}_k)^T \nabla^2 h(\xi, \gamma) (s_k - \tilde{s}_k) \right\| \]

(B.11)

\[ D(\tilde{s}_k, \xi_k) = \frac{\left\| \nabla_s h(\tilde{s}_k, \xi_k) - \nabla_s h(\tilde{s}_0, \xi) \right\|}{\left\| \nabla_s h(\tilde{s}_k, \xi_k) \right\|} \]

(B.12)

The first ratio, \( E(s_k, \tilde{s}_k, \xi_k, \zeta) \), computes the error in linearizing the model by comparing the magnitude of the second-order term to the first-order term of the Taylor series expansion of \( h(s, \xi) \). Recall that from the definition of \( R_1(s_k, \tilde{s}_k, \xi_k, \zeta) \),

\[ \frac{\left\| R_1(s_k, \tilde{s}_k, \xi_k, \zeta) \right\|}{\left\| \nabla_s h(\tilde{s}_k, \xi_k) \right\|} \leq E(s_k, \tilde{s}_k, \xi_k, \zeta) \]

(B.13)

Thus by evaluating \( E(s_k, \tilde{s}_k, \xi_k, \zeta) \) for typical tracking scenarios, we are bounding
the linearization error. The second ratio, $D(\bar{s}_k, \xi_k)$, evaluates how much the first order approximation of $h(s, \xi)$ varies over time, and thus indicates the accuracy of the time-invariant property of the new model.

Consider the following significantly different initial target states $(a - e)$,

\[
\begin{align*}
  s^a_0 &= [100\ 3\ 40\ 7], \\
  s^b_0 &= [-20\ -4\ 200\ 1], \\
  s^c_0 &= [-70\ 5\ -50\ -6], \\
  s^d_0 &= [50\ 2\ 95\ 10], \\
  s^e_0 &= [150\ -15\ 10\ 0]
\end{align*}
\]  

(B.14)

The platform states are defined as follows,

\[
\xi_k = \left[\xi_{x_k}, \dot{\xi}_{x_k}, \xi_{y_k}, \dot{\xi}_{y_k}\right]^T = [-35\text{km}, 100\text{m/s}, -15\text{km}, 20\text{m/s}]^T
\]  

(B.15)

Note that we have assumed that the platform is moving along a linear track with a constant speed. Recall the equation relating the platform altitude to the $x$ and $y$ displacement of the aircraft, $\xi = \sqrt{\xi_x^2 + \xi_y^2} \theta_d$. Assuming a depression angle of $\theta_d = 15^\circ$, the corresponding platform altitude is $\bar{\xi}_c = 10203.2\text{m}$.

We now allow the system to propagate according to the following model: $T = 0.1\text{s}, \sigma_x = \sigma_y = 0.5, \sigma_r = 20\text{m}, \sigma_r = 5\text{m/s}, \sigma_a = 0.5^\circ$ with a true track variability parameter of $\sigma_p = 1.5$. The value $\sigma_p$ is used as the value for $\sigma_x$ and $\sigma_y$ to simulate the target tracks (treated as true data for simulation purposes). The tables seen in the Appendix, Section A, present the values of the quantities in Equations (B.11) and (B.12) at various running times, $k$. Notice that the analysis was performed for a maximum iteration count of $k = 100$. At a sampling time of $T = 0.1\text{s}$, this corresponds to a maximum usage length of 10s for a specific tracking allocation. As will be seen later in the simulations section, the typical stopping time was around $k = 10$ to 15 iterations (1 to 1.5 seconds), thus the assumption of a maximum time of 10 seconds is quite conservative.

Consider the tables in the Appendix, Section A. Table A.1 represents the rate of change of the Jacobian matrix with respect to the micro-manager tracking time, $k$, while tables A.2 and A.3 compare the ratio of the second-order term of the Taylor series expansion to the first-order term, for $\zeta = 0.1$ and $\zeta = 0.8$, respectively.
can see that the values in the above tables are small, with approximate bounds of,

\[
\|R_1(s_k, \bar{s}_k, \xi_k, \zeta)\| \leq E(s_k, \bar{s}_k, \xi_k, \zeta) \leq 0.016 \tag{B.16}
\]

\[
D(\bar{s}_k, \xi_k) = \frac{\|\nabla s h(\bar{s}_k, \xi_k) - \nabla s h(\bar{s}_0, \xi_0)\|}{\|\nabla s h(\bar{s}_k, \xi_k)\|} \leq 0.012 \tag{B.17}
\]

for any of the listed initial states and for all \( k \leq 100 \). Thus, on the micro-manager time scale, we can safely state that the system is well-approximated by the following linear, time-invariant state space model, \(^1\)

\[
s_{k+1}^l = F s_k^l + G w_k^l
\]

\[
z_k^l = \begin{cases} H s_k^l + \frac{1}{\sqrt{\nu_l}} v_k^l & \text{w.p. } p_d^l \\ \emptyset & \text{w.p. } 1 - p_d^l \end{cases} \tag{B.19}
\]

where the linearized measurement model is defined as, \( H_l = \nabla s h(\bar{s}_k^l, \xi_k) \) evaluated using the initial target and platform states.

### B.3 Kalman Filtering with Intermittent Observations

Due to the fact that we cannot assume that the radar successfully receives all observations, we must re-consider the classical Kalman filtering problem, but this time noting the presence of these missed detections. The presence of intermittent observations causes the normal \((p_d = 1)\) Riccati recursion, Equation (2.17), to no longer hold. Note that it is assumed here that the observations are dropped completely, rather than only partially received as in [57]. The general recursion, stated below, takes into account the probability of detection, and is derived in [86]. We will provide an outline of the derivation here to give the reader some intuition into the filtering equations.

The derivation begins by treating the arrival of an observation as a Bernoulli random variable, \( \kappa \), with PMF \( f(\kappa; p_d) = p_d^\kappa (1 - p_d)^{1-\kappa} \) where \( p_d \) is the probability of detection. The output noise for a given time step \( k \) and target \( l \), \( v_k = \frac{1}{\sqrt{\nu}} v_k^l \)

\(^1\)The Gaussian property and covariance calculations follow similarly under the assumption that the SNR is moderate, allowing the higher-order-terms to be omitted.
is defined as (index \(l\) omitted),

\[
Pr(\nu_k|\kappa_k) = \begin{cases} 
N(0, R) & \text{if } \kappa_k = 1 \\
N(0, \sigma^2 I) & \text{if } \kappa_k = 0 
\end{cases}
\] (B.20)

for a given \(\sigma^2\). The idea of the derivation is to use a “dummy” observation with a fixed variance when the observation is missed, then take the limit as \(\sigma\) approaches infinity. Incorporating this idea into the equations for the state and covariance updates, we obtain the following equations,

\[
s_k^{d | k} = s_k^{d | k - 1} + P_k^{d | k - 1}H^T \left[ HP_k^{d | k - 1}H^T + 1_{(1)}(\kappa_k)R \right]^{-1} \left( z_k - Hs_k^{d | k - 1} \right) + 1_{(0)}(\kappa_k)\sigma^2 I \] (B.21)

\[
P_k^{d | k} = P_k^{d | k - 1} - P_k^{d | k - 1}H^T \left[ HP_k^{d | k - 1}H^T + 1_{(1)}(\kappa_k)R \right]^{-1} HP_k^{d | k - 1} + 1_{(0)}(\kappa_k)\sigma^2 I \] (B.22)

where the indicator function is defined as,

\[
1_A(x) = \begin{cases} 
1 & \text{if } x \in A \\
0 & \text{if } x \notin A 
\end{cases}
\] (B.23)

Taking the limit as \(\sigma^2 \to \infty\), it can be shown [86] that the above equations (Equations (B.21) and (B.22)) when combined with Equations (2.14) and (2.16) form the modified state update and the modified Riccati update, shown below,

\[
s_{k+1}^{d} = F_s s_k + \kappa_k Fp_k^l H \left[ H^T P_k^l H + R \right]^{-1} \left( z_k - H^T s_k \right) \] (B.24)

\[
P_k^{d + 1} = Fp_k^l F^T + Q - \kappa_k Fp_k^l H^T \left( H^T P_k^l H + R \right)^{-1} HP_k^l F^T \] (B.25)

where we use the simplifying notation \(s_k^d = s_k^{d | k - 1}\) and \(P_k^d = P_k^{d | k - 1}\). Note that in the deterministic update, the Riccati recursion is guaranteed to converge to some stationary matrix; however, due to the random process \(\kappa_1, \kappa_2, \ldots, \kappa_k\) introduced, the update is now stochastic and convergence is no longer guaranteed. A numerical lower bound for the value of \(p_d\) is discussed in [86] in order to ensure convergence.
For the purposes of this paper, we consider a probability of detection high enough that we do not need to worry about the possibility of the modified Riccati update diverging from the stationary matrix.²

Note that $\kappa_k = 1 - \mathbf{1}_\theta(\tilde{z}_k^l) = \mathbf{1}_{\text{wce}}(\tilde{z}_k^l)$, thus the modified Riccati update is in fact measurement-dependent (and model dependent) unlike the classical Riccati update, in Equation (2.17), which is only dependent upon the model. We define the following error covariance update functions (the measurement-dependent Riccati update and the measurement-free Lyapunov update) for later use,

$$R(P_k, \tilde{z}_k^l) \overset{\text{def}}{=} FP_k F^T + Q - \mathbf{1}_{\text{wce}}(\tilde{z}_k^l) F P_k H^T (H P_k H^T + R)^{-1} H P_k F^T$$  \hspace{1cm} (B.26)

$$L(P_k) \overset{\text{def}}{=} FP_k F^T + Q$$  \hspace{1cm} (B.27)

Note that the Lyapunov update, Equation (B.27), is equivalent to the Riccati update, Equation (B.26), for a given time $k$ if $\tilde{z}_k^l = \emptyset$. That is, when an observation is missed, the covariance update is equivalent to just predicting the error covariance.

### B.4 Threat Error Covariance Derivation

The details of the second-order Taylor approximation of the threat error is presented. Keeping two terms of the expansion in Equation (3.22), we obtain the following estimate, with $s = E \{ \text{vec}(ee^T) \} = \text{vec}(P) \in \mathbb{R}^{4 \times 1}$,

$$P_T \approx E \{ (r - E\{r\})(r - E\{r\})^T \}$$  \hspace{1cm} (B.28)

$$= E \left\{ \left( \Gamma(\hat{s}) + \nabla_s \Gamma e + \frac{1}{2} \nabla^2_s \Gamma e \otimes e - \left( \Gamma(\hat{s}) + \frac{1}{2} \nabla^2_s \Gamma s \right) \right) \right\}$$

$$= E \left\{ \left( \nabla_s \Gamma e + \frac{1}{2} \nabla^2_s \Gamma e \otimes e - \frac{1}{2} \nabla^2_s \Gamma s \right) \right\}$$  \hspace{1cm} (B.29)

$$= E \left\{ \left( \nabla_s \Gamma e + \frac{1}{2} \nabla^2_s \Gamma e \otimes e - \frac{1}{2} \nabla^2_s \Gamma s \right) \right\}$$  \hspace{1cm} (B.30)

²This stationary matrix is no longer unique (unlike the standard algebraic Riccati equation) due to the stochastic nature of the modified Riccati update.
\begin{align*}
\mathbf{P}_\Gamma & \approx \nabla_s \Gamma \mathbf{P} (\nabla_s \Gamma)^T + \frac{1}{2} \nabla_s \Gamma E \left\{ \mathbf{e} (\mathbf{e} \otimes \mathbf{e})^T \right\} (\nabla_s \Gamma)^T \\
& \quad - \frac{1}{2} \nabla_s \Gamma E \left\{ \mathbf{e} \mathbf{s}^T \right\} (\nabla_s \Gamma)^T + \frac{1}{2} \nabla_s \Gamma E \left\{ (\mathbf{e} \otimes \mathbf{e}) \mathbf{e}^T \right\} (\nabla_s \Gamma)^T \\
& \quad + \frac{1}{4} \nabla^2_s \Gamma E \left\{ (\mathbf{e} \otimes \mathbf{e}) (\mathbf{e} \otimes \mathbf{e})^T \right\} (\nabla^2_s \Gamma)^T - \frac{1}{4} \nabla^2_s \Gamma E \left\{ (\mathbf{e} \otimes \mathbf{e}) \mathbf{s}^T \right\} (\nabla^2_s \Gamma)^T \\
& \quad - \frac{1}{2} \nabla^3_s \Gamma E \left\{ \mathbf{s} \mathbf{e}^T \right\} (\nabla^3_s \Gamma)^T - \frac{1}{4} \nabla^3_s \Gamma E \left\{ \mathbf{s} (\mathbf{e} \otimes \mathbf{e})^T \right\} (\nabla^3_s \Gamma)^T \\
& \quad + \frac{1}{4} \nabla^2_s \Gamma E \left\{ \mathbf{s} \mathbf{s}^T \right\} (\nabla^2_s \Gamma)^T
\end{align*}

\text{(B.31)}

Now, using the fact that \( \mathbf{e} \) is zero-mean, the above simplifies to,

\begin{align*}
\mathbf{P}_\Gamma & \approx \nabla_s \Gamma \mathbf{P} (\nabla_s \Gamma)^T + \frac{1}{2} \nabla_s \Gamma C (\nabla^2_s \Gamma)^T \\
& \quad + \frac{1}{2} \nabla^2_s \Gamma C^T (\nabla_s \Gamma)^T + \frac{1}{4} \nabla^2_s \Gamma \left( D - \mathbf{s} \mathbf{s}^T \right) (\nabla^2_s \Gamma)^T
\end{align*}

\text{(B.32)}

where \( C = E \left\{ \mathbf{e} (\mathbf{e} \otimes \mathbf{e})^T \right\} \), and \( D = E \left\{ (\mathbf{e} \otimes \mathbf{e}) (\mathbf{e} \otimes \mathbf{e})^T \right\} = E \left\{ (\mathbf{e}^T) \otimes (\mathbf{e}^T) \right\} \).

Knowing that the PDF of the noise term \( \mathbf{e} \) is symmetric about the mean (since \( \mathbf{e} \) is a Gaussian-distributed random variable), then the third order moments contained in the matrix \( C \) are zero [60]. The expression for the error covariance of the threat then reduces to the following,

\begin{align*}
\mathbf{P}_\Gamma & \approx \nabla_s \Gamma \mathbf{P} (\nabla_s \Gamma)^T + \frac{1}{4} \nabla^2_s \Gamma \left( D - \mathbf{s} \mathbf{s}^T \right) (\nabla^2_s \Gamma)^T
\end{align*}

\text{(B.33)}

The matrix \( D \) contains fourth-order moments, and along with the matrix \( \mathbf{s} \mathbf{s}^T \), are of dimension \( 4^2 \times 4^2 \). The matrix \( D \) can equivalently be regarded as a \( 4 \times 4 \) block matrix with \( 4 \times 4 \) matrix elements,

\begin{equation}
D = E \left[ \begin{array}{cccc}
\mathbf{c}_1 \mathbf{e}^T \\
\mathbf{e}_1 \mathbf{e}^T \\
\mathbf{e}_2 \mathbf{e}^T \\
\mathbf{e}_3 \mathbf{e}^T \\
\mathbf{e}_4 \mathbf{e}^T
\end{array} \right]
\end{equation}

\text{(B.34)}

\Rightarrow D_{mn,ij} = E \left\{ \mathbf{e}_m \mathbf{e}_n \mathbf{e}_i \mathbf{e}_j \right\}

\text{(B.35)}

117
where \( e_i \in \mathbb{R}^{4 \times 1} \) represents the \( i^{th} \) element of the error vector \( e \) and \( m, n \) and \( i, j \) refers to the block’s and element’s row and column, respectively. For Gaussian PDFs, \( D \) can be computed from the error covariance matrix, \( \mathbf{P} \), as the following Gaussian approximation [26] (p.192),

\[
D_{mn,ij} = \mathbf{P}_{mi} \mathbf{P}_{nj} + \mathbf{P}_{mj} \mathbf{P}_{nj} + \mathbf{P}_{nj} \mathbf{P}_{nj}
\]  

(B.36)

The elements of the matrix \( \mathbf{s s}^T \) can be written in terms of elements of the covariance matrix \( \mathbf{P} \) as,

\[
\mathbf{s s}_{mn,ij}^T = \mathbf{P}_{mi} \mathbf{P}_{nj}
\]  

(B.37)
Appendix C

Appendix: Proofs

C.1 Proofs: Macro-manager

C.1.1 Proof of Lemma 3.4.1

Proof. We begin by computing the derivatives of $\|p\|_2^2$ with respect to $p$,

\[
\frac{\partial}{\partial p} \|p\|_2^2 = 4n^2 p^3 - 6n^2 p^2 + 4n(n + \sigma^2) p - 2n(\mu + \sigma^2) \tag{C.1}
\]

\[
\frac{\partial^2}{\partial p^2} \|p\|_2^2 = 12n^2 p(p - 1) + 4n(n + \sigma^2) \tag{C.2}
\]

The general solution method to determine the optimal parameter $p$ in the least-squares sense is to compute the gradient and equate it to zero. Since we only have one parameter, we only have one gradient equation, represented by Equation (C.1). In order to ensure that we minimize $\|p\|_2^2$ when equating the gradient to zero, we must first be certain that $\|p\|_2^2$ is convex in $p$. This is equivalent to checking if the second derivative, Equation (C.2), is positive for all $p$.

\[
\frac{\partial^2}{\partial p^2} \|p\|_2^2 > 0 \tag{C.3}
\]

\[
12n^2 p(p - 1) + 4n(n + \sigma^2) > 0 \tag{C.4}
\]

\[
3np(p - 1) + n + \sigma^2 > 0 \tag{C.5}
\]
Referring to Figure 3.13(b), we can see that the maximum value of the variance of a binomial distribution, \( \sigma_b^2 = np(1 - p) \), is \( n/4 \) achieved at \( p = 1/2 \). Thus the first term in Equation (C.5) is bounded below by \(-3n/4\), that is,

\[
3np(p - 1) + n + \sigma^2 > -\frac{3n}{4} + n + \sigma^2 = \frac{n}{4} + \sigma^2 > 0 \quad (C.6)
\]
since \( n, \sigma^2 > 0 \). As a result, the second derivative of \( \|\rho\|_2^2 \) with respect to \( p \) is strictly positive and equivalently \( \|\rho\|_2^2 \) is convex in \( p \). Using the convexity result, we can expect to minimize \( \|\rho\|_2^2 \) by equating the first derivative, Equation (C.1), to zero.

\[
\frac{\partial}{\partial p} \|\rho\|_2^2 = 0 \quad (C.7)
\]

\[
\Rightarrow 4n^2 p^3 - 6n^2 p^2 + 4n(n + \sigma^2)p - 2n(\mu + \sigma^2) = 0 \quad (C.8)
\]

\[
\Rightarrow 2np^3 - 3np^2 + 2(n + \sigma^2)p - \sigma^2 - \mu = 0 \quad (C.9)
\]

Equation (C.9) is a cubic function in \( p \). There are three solutions to this equation, but only one of which is real, as we will now show.

From [38], it follows that the cubic, Equation (C.9), has a unique real root if and only if the following condition holds,

\[
3n^2 \sigma^2 + 12n \sigma^4 + 16 \sigma^6 - 27n^2 \mu + 27n \mu^2 + 7n^3 > 0 \quad . \quad (C.10)
\]

We now show that for any \( n > 0 \) and \( \sigma^2 \geq 0 \), the inequality (C.10) follows. Since \( \sigma^2 \geq 0 \), we have,

\[
3n^2 \sigma^2 + 12n \sigma^4 + 16 \sigma^6 - 27n^2 \mu + 27n \mu^2 + 7n^3 \geq -27n^2 \mu + 27n \mu^2 + 7n^3 . \quad (C.11)
\]

Let \( g(n, \mu) = -27n^2 \mu + 27n \mu^2 + 7n^3 \). Differentiating \( g(n, \mu) \) with respect to \( \mu \) yields,

\[
g(n, \mu)_\mu = -27n^2 + 54n \mu . \quad (C.12)
\]

The second derivative of \( g(n, \mu) \) with respect to \( \mu \) equals \( 54n \) and thus is strictly positive for any \( n > 0 \). Setting the right-hand side of (C.12) to zero and solving for \( \mu \) then yields the unique global minimizer \( \mu^* \) of \( g(n, \mu) \), that is \( \mu^* = n/2 \). Since
\( g(n, \mu^*) = n^3/4 > 0 \), the inequality \((\text{C.11})\) implies \((\text{C.10})\) must hold for any \( n > 0 \). Applying Theorem ?? in cite ??, the unique solution to \((\text{C.9})\) is then given by \((\text{3.51})\). 

\section*{C.1.2 Proof of Lemma 3.4.2}

\textit{Proof.} By Lemma 3.4.1 there is a unique real solution \( p^* \) to \((\text{3.50})\) for any variance \( \sigma^2 \geq 0 \), mean \( \mu \in \mathbb{R} \), and \( n > 0 \). We now show that for any mean \( \mu \in [0, n] \), variance \( \sigma^2 \geq 0 \), and \( n > 0 \), if \( p^* \notin [0, 1] \) then there exists a \( \tilde{p} \in [0, 1] \) such that

\[
\left( \mu - n\tilde{p} \right)^2 + \left( \sigma^2 - n\tilde{p}(1 - \tilde{p}) \right)^2 < \left( \mu - np^* \right)^2 + \left( \sigma^2 - np^*(1 - p^*) \right)^2 \tag{C.13}
\]

and thus any \( p^* \notin [0, 1] \) cannot be the unique real solution to \((\text{3.50})\).

First assume \( p^* > 1 \) and let \( \tilde{p} = 1 \). We then have for any mean \( \mu \in [0, n] \),

\[
\begin{align*}
(\mu - np^*)^2 &= \mu^2 - np^*(2\mu - np^*) \\
&= \mu^2 + n\left( 2\mu(1 - p^*) + n((p^*)^2 - 1) - (2\mu - n) \right) \\
&\geq \mu^2 + n\left( 2n(1 - p^*) + n((p^*)^2 - 1) - (2\mu - n) \right) \\
&= \mu^2 + n(n(p^*-1)^2 - (2\mu - n)) \\
&> (\mu - n\tilde{p})^2,
\end{align*}
\]

where the first inequality holds since \( p^* > 1 \), \( \mu \in [0, n] \), and \( n > 0 \), and the last inequality holds since \( p^* > 1 \) and \( n > 0 \).

Next assume \( p^* < 0 \) and let \( \tilde{p} = 0 \). We then have for any mean \( \mu \geq 0 \),

\[
\begin{align*}
(\mu - np^*)^2 &= \mu^2 + np^*(np^*-2\mu) \\
&> \mu^2 \\
&= (\mu - n\tilde{p})^2,
\end{align*}
\]

where the inequality holds since \( np^* < 0 \) and \( (np^*-2\mu) < 0 \) for any \( \mu \geq 0 \), \( p^* < 0 \) and \( n > 0 \).

Lastly, note that any \( p^* \notin [0, 1] \) implies \( p^*(1 - p^*) < 0 \), thus for any variance
\( \sigma^2 \geq 0 \) and \( n > 0 \) we have,

\[
(\sigma^2 - np^* (1 - p^*))^2 = \sigma^4 + np^* (1 - p^*) (np^* (1 - p^*) - 2\sigma^2) > \sigma^4
\]

\[
= (\sigma^2 - n\bar{p}(1 - \bar{p}))^2,
\]

where \( \bar{p} \in \{0, 1\} \). Combining (C.14)-(C.16) yields (C.13).

\( \square \)

### C.1.3 Proof of Lemma 3.4.3

**Proof.** Before proving Lemma 3.4.3, we state the following lemma from Mattner [45],

**Lemma C.1.1** (Mattner [45]). Consider two random variables, denoted \( X \sim \text{Bin}(n_1, p_1) \) and \( Y \sim \text{Bin}(n_2, p_2) \). Then \( X \geq_{LR} Y \) if and only if,

\[
p_2 = 0 \quad \text{or} \quad n_1 \geq n_2 \quad \text{and} \quad \frac{n_1 p_1}{1 - p_1} \geq \frac{n_2 p_2}{1 - p_2}
\]  

(C.17)

Define \( f_1 = \text{Bin}(N, p_1) \) and \( f_2 = \text{Bin}(N, p_2) \) corresponding to the two binomial random variables, \( K_1 \sim f_1 \) and \( K_2 \sim f_2 \), with a common parameter \( N \). By Lemma C.1.1, we know that for \( X \sim \text{Bin}(n_1, p_1) \) and \( Y \sim \text{Bin}(n_2, p_2) \), we require that either \( p_2 = 0 \) or \( n_1 \geq n_2 \) and \( \frac{n_1 p_1}{1 - p_1} \geq \frac{n_2 p_2}{1 - p_2} \) in order for \( X \geq_{LR} Y \). Ignoring the trivial case \( p_2 = 0 \), clearly \( n_1 \geq n_2 \) is satisfied for \( n_1 = n_2 = N \), then a sufficient condition for \( K_1 \geq_{LR} K_2 \) is,

\[
\frac{N p_1}{1 - p_1} \geq \frac{N p_2}{1 - p_2}
\]

(C.18)

\[
\Rightarrow \frac{p_1}{1 - p_1} \geq \frac{p_2}{1 - p_2}
\]

(C.19)

\[
\Rightarrow p_1 \geq p_2
\]

(C.20)

\( \square \)
C.1.4 Proof of Theorem 3.2.2

This proof is restated from Ross [78] in the context of our problem. It is necessary to state and prove some lemmas before proving the main result.

Lemma C.1.2. Suppose that $X_1$ and $X_2$ are two, independent random variables with probability distributions $f$ and $g$, respectively and let $X_1 \geq_{LR} X_2$. If $h(x_1, x_2)$ is a real-valued function satisfying,

$$h(x_2, x_1) \geq h(x_1, x_2) \text{ whenever } x_1 \geq x_2 \quad (C.21)$$

then,

$$h(X_2, X_1) \geq_{st} h(X_1, X_2) \quad (C.22)$$

Proof. We first define two new random variables, $U$ and $V$, from the random variables $X_1$ and $X_2$, as $U = \max(X_1, X_2)$ and $V = \min(X_1, X_2)$. Next, we condition on $U$ and $V$ with $u$ and $v$, respectively, $u \geq v$. The conditional distribution of $h(X_1, X_2)$ is concentrated on the two points, $h(u, v)$ and $h(v, u)$. The following probability,

$$\lambda_1 \equiv P(X_1 = \max(X_1, X_2), X_2 = \min(X_1, X_2) | U = u, V = v)$$

$$\frac{f(u)g(v)}{f(u)g(v) + f(v)g(u)} \quad (C.23)$$

is assigned to the smaller point $h(u, v)$. Similarly, $h(X_2, X_1)$ is also concentrated on the two points $h(u, v)$ and $h(v, u)$ with probability,

$$\lambda_2 \equiv P(X_2 = \max(X_1, X_2), X_1 = \min(X_1, X_2) | U = u, V = v)$$

$$\frac{g(u)f(v)}{g(u)f(v) + g(v)f(u)} \quad (C.24)$$

given to the smaller point $h(u, v)$. Recall the hypotheses $u \geq v$ and $X_1 \geq_{LR} X_2$ and notice that the expressions (C.23) and (C.24) have a common denominator, thus,

$$\lambda_1 \geq \lambda_2 \Rightarrow f(u)g(v) \geq g(u)f(v) \quad (C.25)$$
Therefore, conditional on \( U = u \) and \( V = v \), \( h(X_2, X_1) \) is larger than \( h(X_1, X_2) \) with respect to first order stochastic dominance,

\[
P(h(X_2, X_1) \geq a | U = u, V = v) \geq P(h(X_1, X_2) \geq a | U = u, V = v)
\]

(C.26)

Taking expectations of both sides results in the desired result,

\[
h(X_2, X_1) \geq_{st} h(X_1, X_2).
\]

(C.27)

**Lemma C.1.3.** If \( d(x) \) is a non-negative convex function, then for all \( 0 < r_1 < r_2 \) and \( y \geq 0 \),

\[
r_1 d(y) + r_2 d(y + r_1 d(y)) \leq r_2 d(y) + r_1 d(y + r_2 d(y)).
\]

(C.28)

**Proof.** Fix \( y \geq 0 \). If \( d(y) = 0 \) then (C.28) is satisfied since both sides vanish. We thus assume \( d(y) > 0 \). Note that \( y \geq 0 \) and \( 0 < r_1 < r_2 \) implies,

\[
0 \leq y < y + r_1 d(y) < y + r_2 d(y).
\]

(C.29)

Next observe that convexity of \( d(x) \) implies that for any \( t, h > 0 \),

\[
d(y + t) < \frac{t}{r_1 h} d(y + t + h) + \frac{h}{t + h} d(y),
\]

\[
\Rightarrow (t + h)d(y + t) - td(y + t + h) - hd(y) < 0,
\]

\[
\Rightarrow (t + h)(d(y + t) - d(y)) - t(d(y + t + h) - d(y)) < 0,
\]

\[
\Rightarrow \frac{d(y + t) - d(y)}{t} < \frac{d(y + t + h) - d(y)}{t + h}.
\]

Hence \( (d(y + t) - d(y)) / t \) is increasing in \( t \), thus from (C.29) we have,

\[
\frac{d(y + r_1 d(y)) - d(y)}{r_1 d(y)} < \frac{d(y + r_2 d(y)) - d(y)}{r_2 d(y)}.
\]

(C.30)

124
Rearranging (C.30) as follows then yields (C.28),

\[(d(y + r_1 d(y)) - d(y)) r_2 d(y) < (d(y + r_2 d(y)) - d(y)) r_1 d(y)\]
\[\Rightarrow r_2 d(y + r_1 d(y)) - r_2 d(y) < r_1 d(y + r_2 d(y)) - r_1 d(y),\]
\[\Rightarrow r_1 d(y) + r_2 d(y + r_1 d(y)) < r_2 d(y) + r_1 d(y + r_2 d(y)).\]

(C.31)

**Lemma C.1.4.** If \(d(y)\) is a non-negative, non-decreasing, convex function, then \(x + td(x)\) is an increasing function of \(x, x \geq 0, \text{ for all } t \geq 0\).

**Proof.** Given the form of the function \(d\), we know that \(d'(x) \geq 0\) for all \(x\). By differentiating \(x + td(x)\) with respect to \(x\), we obtain \(1 + td'(x)\) which is positive for all \(x,t \geq 0, \text{ thus } x + td(x)\) in increasing in \(x\) for all \(x,t \geq 0\).

We now use the aforementioned results to prove Theorem 3.2.2.

**Proof.** We first introduce some notation; the target sequence \(l_1, l_2, \ldots, l_L\) corresponds to the sequence of ranks \(\hat{X}_1, \hat{X}_2, \ldots, \hat{X}_L\). The proof of the main result is based on induction. The base case, with two targets, can be shown using Lemmas C.1.2 and C.1.3. The total accumulated reward is maximized by observing the target with the larger rank (with respect to likelihood ratio ordering) first, thus the base case, with \(L = 2\) holds.

**Induction Hypothesis:** Assume that for \(L - 1\) targets it is optimal to observe targets in order of decreasing priority ranks (with respect to the likelihood ratio order).

We now analyze the \(L\) target scenario, first considering the class of policies that specify the target observation queue at time zero, termed static policies. There are \(L!\) of these policies, corresponding to the \(L!\) possible orders to observe \(L\) targets. Consider any static policy. Let target \(l_j\) be the target that is observed last in the queue. Using Lemma C.1.4, it follows that among the queues which specify target \(l_j\) as the last observation, the queue resulting in the maximum total accumulated reward is maximized by observing the target with the larger rank (with respect to likelihood ratio ordering) first, thus the base case, with \(L = 2\) holds.
reward is the one which stochastically maximizes the accumulated reward for observing the first $L - 1$ targets. Recalling the induction hypothesis, of all observation queues that specify target $l_j$ as the last observation, the queue which stochastically maximizes the accumulated reward is the one which observes the first $L - 1$ targets in order of decreasing priority ranks (in a likelihood ratio sense).

*Interchange Argument:* If $l_j$ is not the target with the smallest priority rank, $\hat{X}_1$, that is, $l_j \neq 1$, then the best observation queue is the one which specifies the observation of the target with the smallest priority rank immediately prior to the last target observation $l_j$. However, if we let $z$ denote the level of accumulated reward when only targets $l_1$ and $l_j$ remain, then the problem is now one where there are two target observations a a function $d_z(x) = d(z + x)$. Since we have defined the function $d$ to be non-negative and convex, it follows from the base case of the induction argument that if we interchange the observations of targets $l_j$ and $l_1$, we would achieve a stochastically larger accumulated reward. Thus for any static policy, there exists one that observes target $l_1$ last and has a stochastically larger reward. However, out of all the queues that specify to observe target $l_1$ last, the one with the stochastically largest accumulated reward is (by Lemma C.1.4 and the induction hypothesis) the order provided by the theorem.

We now consider the class of observation queues which allow for the option of changing the order of the remaining observations after the first $k$ targets have been observed. The policies associated with these queues are referred to as *dynamic* policies. It is shown that the optimal policy in the static case (above) also stochastically maximizes the accumulated reward in the class of dynamic policies. Suppose $k = 1$, then it follows from the interchange argument that it is optimal to not assign a queue that specifies anything other than to observe targets in order of decreasing priority ranks. If this is true for $k - 1$ completed observations then, by the same argument, it is true when $k$ observations have been completed since, by the induction hypothesis, we would only use this opportunity at most once. Since the result is true for all $k$, then the optimal policy in the static case also stochastically maximizes the reward in the class of dynamic policies. \hfill $\square$