Informative Path Planning and Sensor Scheduling for Persistent Monitoring Tasks

by

Syed Talha Jawaid

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Applied Science
in
Electrical and Computer Engineering

Waterloo, Ontario, Canada, 2013

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

In this thesis we consider two combinatorial optimization problems that relate to the field of persistent monitoring.

In the first part, we extend the classic problem of finding the maximum weight Hamiltonian cycle in a graph to the case where the objective is a submodular function of the edges. We consider a greedy algorithm and a 2-matching based algorithm, and we show that they have approximation factors of $\frac{1}{2+\kappa}$ and $\max\{\frac{2}{3(2+\kappa)}, \frac{2}{3}(1-\kappa)\}$ respectively, where κ is the curvature of the submodular function. Both algorithms require a number of calls to the submodular function that is cubic to the number of vertices in the graph. We then present a method to solve a multi-objective optimization consisting of both additive edge costs and submodular edge rewards. We provide simulation results to empirically evaluate the performance of the algorithms. Finally, we demonstrate an application in monitoring an environment using an autonomous mobile sensor, where the sensing reward is related to the entropy reduction of a given a set of measurements.

In the second part, we study the problem of selecting sensors to obtain the most accurate state estimate of a linear system. The estimator is taken to be a Kalman filter and we attempt to optimize the a posteriori error covariance. For a finite time horizon, we show that, under certain restrictive conditions, the problem can be phrased as a submodular function optimization and that a greedy approach yields a $1 - \frac{1}{e^{1-1/e}}$-approximation. Next, for an infinite time horizon, we characterize the exact conditions for the existence of a schedule with bounded estimation error covariance. We then present a scheduling algorithm that guarantees that the error covariance will be bounded and that the error will die out exponentially for any detectable LTI system. Simulations are provided to compare the performance of the algorithm against other known techniques.

Acknowledgements

I would like to thank my supervisor Professor Stephen Smith for giving me the opportunity to work with him at the University of Waterloo. I am grateful for his patience, invaluable guidance and support as well as the numerous paper edits and informative conversations during meetings.

I should also acknowledge the Professors of the courses that I have taken and mention that I appreciate them for taking out time to share their knowledge.

I would also like to thank the the readers of this thesis – Professor Shreyas Sundaram and Professor Christopher Nielsen – for their time and valuable feedback.

Also, it would be unmindful of me to not show gratitude to my family for their patience, support and love.

Finally, I would like to thank the Natural Science and Engineering Research Council of Canada and the University of Waterloo for providing funding without which this research work would not have been possible.

Table of Contents

List of Figures

Chapter 1

Introduction

This thesis covers two topics that are both relevant in the field of persistent monitoring. Most problems in this field relate to monitoring some dynamic process over long periods of time. This can be done by using either mobile sensor(s) (on a robot(s)) that continuously traverse the environment or alternatively by placing static sensors at key locations that work together to give a complete picture of the process.

The first topic we look at relates to using a mobile robot to observe an environment; specifically, the submodular maximum traveling salesman problem (TSP). The second problem relates to state estimation and figuring out which measurements to make given a static sensor network.

1.1 The Submodular Maximum Traveling Salesman Problem

The maximum weight Hamiltonian cycle is a classic problem in combinatorial optimization. It consists of finding a path in a graph that starts and ends at the same vertex and visits all other vertices exactly once while maximizing the sum of the weights (i.e., the reward) on the edges traversed. Also referred to as the max-TSP, the problem is NP-hard; however, a number of approximation algorithms have been developed. In [16], four simple approximation algorithms are analyzed. The authors show that greedy, best-neighbor, and 2-interchange heuristics all give a $\frac{1}{2}$ approximation to the optimal tour. They also show that a 2-matching based heuristic, which first finds a perfect 2-matching and then converts that to a tour, gives a $\frac{2}{3}$ approximation. The simple and elegant Serdyukov's algorithm

[47] — which combines a perfect 2-matching and a 1-matching to compute a tour — gives a $\frac{3}{4}$ approximation. The best known deterministic algorithm is given in [45] and it achieves a $\frac{7}{9}$ approximation in $O(n^3)$ time. A number of randomized algorithms also exist such as the one given in [23] that achieves a $\frac{25}{33}$ approximation ratio. In [54] it was shown that this algorithm can be derandomized while maintaining its approximation factor. In this paper we look at extending the max-TSP problem to the case of submodular rewards.

This extension is motivated, in part, by the application of mobile sensing robots to persistently monitor or patrol a large environment [53, 11, 52]. It is desirable to have a closed walk or a tour over which the sensing robot travels. Applications include tasks such as monitoring oil spills [8], forest fires [7] or underwater ocean monitoring [52].

Informative path planning involves using pre-existing information about the environment (such as a probability distribution) to plan a path that maximizes the information gained. It is a topic that has been researched with various different approaches. For example, in [37] the authors look into creating trajectories to best estimate a Gaussian random field by intelligently generating rapidly-exploring random cycles. One common way to measure information content is using mutual information. In [19], this metric is used to place static sensors in a Gaussian field. Other papers investigate maximizing the knowledge at specific points by planning a path for a sensing robot while also taking into account budget constraints [50, 4]. The metrics used to determine the quality of the sensing, such as mutual information, are usually submodular in nature. The defining property of a submodular function is that of decreasing marginal value of each element, i.e., adding an element to a set will result in a larger increase in value than adding that element to a superset of that set. For example, if a sensor is placed close to another, then the benefit gained by the second sensor will be less than if the first sensor had not already been placed. Other areas where submodular functions arise include viral marketing, active learning [17] and AdWords assignment [18].

Our problem can be stated as maximizing a submodular function over the edges of a graph subject to the constraint that the selected edges form a tour. Generally, one way to represent constraints in a combinatorial optimization problem is through the concept of independence systems or its many specializations, including p-systems and matroids. Although unconstrained minimization of a submodular function can be achieved in polynomial time [46, 30], maximizing a non-decreasing submodular function over an independence system constraint is known to be NP-hard. For a monotone submodular function, a number of approximation algorithms exist for optimizing over multiple matroid constraints [15]. Some bounds that include the dependence on curvature are presented in [9]. Local search methods have been found to be particularly useful [57, 14] for both monotone and non-monotone functions. Various results exist for non-monotone submodular function

maximization for both the unconstrained case [13] and for the case where constraints exist, such as multiple matroid and knapsack constraints [38, 39] or p-system constraints [20]. The use of continuous relaxations of the submodular function have also lead to optimal approximation bounds [5] for the case of a single matroid as well as improved bounds for a combination of various constraints [56].

1.2 Sensor Scheduling

When monitoring a process, another technique of obtaining data from the environment is by deploying a sensor network. Each sensor can be equipped with the ability to make a range of possible measurements depending on what the application is. This includes determining a robot's state [26], tracking the position of a target [29], determining troop movements on a battlefield, selecting the frequency in radar and sonar applications, or monitoring tasks such as chemical processes [33], seismic activity or toxin levels at a factory. Sensor selection techniques can also be applied to problems such as adaptive compressed sensing [40].

The actual processing of the data collected could be done in a distributed manner or centralized. In either case, the sensors themselves usually are resource constrained in terms of energy and processing power. The network may be required to have a long life span; therefore, operating every sensor continuously may not be viable. Network constraints also bring about communication limitations between sensors. To overcome these restrictions, sensors can alternate between being awake and asleep. Unless there is a lot of redundancy in the system, this method could result in an incomplete picture of the phenomenon of interest. Therefore, the schedule has to be constructed in an intelligent way in order to get as much information as possible while meeting the energy constraints. This is, in essence, what the sensor selection (or scheduling) problem is.

The sensor selection problem comes up in numerous areas and can be formulated in various ways. For a related problem of coverage, [6] considers how to schedule the operational time of randomly but densely deployed wireless sensors to achieve constant coverage of certain targets while maximizing life span. A slightly different problem of sensor placement in a Gaussian field is considered in [19] using mutual information as a metric. This is extended to take into account communication costs in [35]. In another instance, a near optimal solution to the sensor placement problem is given through the use of genetic algorithms [61].

In the context of linear Gaussian systems, a Kalman filter is the optimal estimator. Using its equations as a basis for formulating the sensor selection problem is therefore a

reasonable approach. In this context, the infinite horizon sensor scheduling problem is studied in [62]. Under some mild conditions, it is shown that the optimal infinite horizon average-per-stage cost as well as the corresponding schedule are independent of the initial covariance. Also, it is shown that the optimal cost can be estimated arbitrarily closely by a periodic schedule (that has a finite period) and also that the error covariance approaches a unique limit cycle under a periodic schedule. An optimal and semi-optimal algorithm that use tree pruning techniques are provided in [55].

In [51], the authors look at the problem of using a Kalman filter if observations are available at each time step with a certain probability. It is shown that the error covariance will be bounded only if the probability of making an observation is above a certain critical value. In [21], the authors provide a method for stochastically selecting measurements for a Kalman filter, based on an intelligently constructed probability distribution, to minimize the expected steady state error covariance.

An application of sensor selection to CO_2 monitoring using a wireless sensor network is demonstrated in [59] by using a convex relaxation in an attempt to approximate the optimal a posteriori covariance for a Kalman filter. The sensor selection problem is combined with the controller problem in [58] to demonstrate an application for controlling the water level in multiple tanks. Here, the LQG problem is modified to include controller cost and network energy cost from the selected sensors, and is framed as a receding horizon mixed integer program.

A convex relaxation based approach to pick measurements for parameter estimation along with solution dependent bounds is given in [32]. This approach is, however, empirically shown to be worse than a greedy algorithm when optimizing the maximum *a posteriori* estimate in [48]. In this paper the authors also show that using the maximum a posteriori covariance gives a constant factor approximation using the greedy algorithm since the objective is submodular. This is applied to an example that uses a Kalman filter. An extension that provides for randomly dropped measurements is done in [49].

A generalized framework for sensor selection in state estimation for Kalman Filtering in presented in [43]. A number of problems can be addressed using this framework such as minimizing the final covariance over a time horizon, the average covariance, just the variance of a single state, or even the cost of a finite horizon LQG regulator. A number of network constraints can also be included. The problem is framed as a relaxed quadratic program. A greedy approach is given though the error bound is not necessarily constant for unstable systems.

1.3 Contributions

We present and analyze two simple algorithms for constructing a maximum reward tour on a graph. The metric used in maximizing the "reward" of a particular tour is a positive monotone submodular function of the edges. We frame this problem as an optimization over an independence system constraint and present two approximation algorithms. The first method is greedy and gives a $\frac{1}{2+\kappa}$ approximation. The second method creates a 2-matching and then turns it into a tour. This gives a $\max\left\{\frac{2}{3(2+\kappa)}, \frac{2}{3}(1-\kappa)\right\}$ approximation where κ is the curvature of the submodular function. Both techniques require $O(|V|^3)$ value oracle calls to the submodular function, where $|V|$ is the number of vertices in the graph. The algorithms are also extended to directed graphs. To obtain these results, we generalize a known bound for the greedy algorithm to be a function of curvature. We also present an approach for the case of a multi-objective optimization consisting of submodular (sensing) rewards on the edges along with modular (travel) costs. We incorporate these two objectives into a single function which is no longer monotone nor positive. We provide bounds on the performance of our algorithms in this case, which depend on the relative weight of the rewards.

For the finite horizon sensor selection problem, we investigate the submodularity properties of the Kalman update equations. Under certain restrictive assumptions, a metric that relates to the volume of the confidence ellipsoid for the a posteriori estimate is shown to be submodular and monotone non-decreasing. This in turn is used to derive a bound for using a greedy algorithm to pick sensors at each time step. Next, we give necessary and sufficient conditions for the existence of an infinite horizon sensor schedule with a bounded error covariance, which makes a novel connection to detectability. We then provide a modification to a simple greedy algorithm for constructing a sensor schedule that guarantees a bounded infinite horizon cost.

1.4 Organization

The structure of this thesis is as follows. In Chapter 2 some preliminary material in given to review some concepts with the objective of introducing notation and some results that will be used throughout the rest of this thesis. In Chapter 3 the submodular maximum traveling salesman problem is defined and analyzed. In Chapter 4 the sensor scheduling problem is studied in the context of submodularity as well as uniform detectability. We end with conclusions and possible future directions in Chapter 6

Chapter 2

Preliminaries

Here we present preliminary concepts. Independence systems and submodular functions are defined and we give a brief summary of results on combinatorial optimization over independence systems. An extension of a known result on optimization over p-systems to incorporate curvature is also given. Next we review some concepts in linear algebra and study the Kalman filter and its boundedness.

2.1 Independence Systems

Combinatorial optimization problems can often be formulated as the maximization or minimization of an objective function $f : \mathcal{F} \to \mathbb{R}$ over a set system $(E, \mathcal{F})$, where E is the base set of all elements and $\mathcal{F} \subseteq 2^E$. An **independence system** is a set system that is closed under subsets (i.e., if $A \in \mathcal{F}$ then $B \subseteq A \implies B \in \mathcal{F}$). Sets in $\mathcal{F}$ are referred to as **independent sets**. For some subset of the base set, $A \subseteq E$, maximal independent sets of A (i.e., all $B \in \mathcal{F}$ such that $B \subseteq A$ and $B \cup \{x\} \notin \mathcal{F}, \forall x \in A \setminus B$) are the **bases** of A.

Definition 2.1.1 (p-system)**.** Given an independence system $S = (E, \mathcal{F})$, for any $A \subseteq E$, let $U(A)$ and $L(A)$ be the sizes of the maximum and minimum cardinality bases of A, respectively. Then, S is a p-system, for some $p \in \mathbb{R}_+$, if $U(A) \leq pL(A)$ for all $A \subseteq E$. ■

Definition 2.1.2 (p-extendible system)**.** An independence system $(E, \mathcal{F})$ is p-extendible if given any independent set $B \in \mathcal{F}$, for every subset A of B and for every $x \notin A$ such that $A \cup \{x\} \in \mathcal{F}$, there exists $C \subseteq B \setminus A$ such that $|C| \leq p$ and for which $(B \setminus C) \cup \{x\} \in \mathcal{F}$. ■

Definition 2.1.3 (Matroid)**.** An independence system $(E, \mathcal{F})$ is a matroid if it satisfies the additional property that if $X, Y \in \mathcal{F}$ such that $|X| > |Y|$, then $\exists x \in X \backslash Y$ with $Y \cup \{x\} \in \mathcal{F}$. ■

Remark 2.1.4 (Relationship between systems). These specific types of independence systems are intricately related. A matroid is a 1-extendible system and any p-extendible system is a p-system. In addition, any independence system can be represented as the intersection of a finite number of matroids [34]. •

An example of a matroid is the *partition matroid*. The base set is composed of n disjoint sets, $\{E_i\}_{i=1}^n$. Given $k \in \mathbb{Z}_+^n$, the matroid is defined by the collection $\mathcal{F} := \{A \subseteq E : |A \cap E_i| \leq k_i, \forall i = 1 \dots n\}$. Another example is the *uniform matroid* which is defined by the collection of all sets of size less then or equal to $m \in \mathbb{Z}_+$, i.e., $\mathcal{F} := \{A \subseteq E : |A| \leq m\}$.

2.2 Sequences

A **sequence** $A = (a_1, \dots, a_k)$, $k \in \mathbb{Z}_{\geq 0}$ can be constructed by choosing elements from a base set of elements E, i.e., $a_i \in E$. A number of operations can be done on sequences.

Length: The length of a sequence, $|A|$, is the number of elements in the sequence. So for $A = (a_1, \dots, a_k)$, $|A| = k$.

Concatenation: Two sequences defined over the same base set can be concatenated into a larger sequence. Given two sequences $A = (a_1, \dots, a_k)$ and $B = (b_1, \dots, b_l)$, $A \parallel B = (a_1, \dots, a_k, b_1, \dots, b_l)$.

Subsequence: A **subsequence** of A is a sequence derived from it by deleting some elements but not changing the order of the remaining elements, e.g., $B = (a_3, a_5)$ is a subsequence of A, and is denoted $B \subseteq A$.

2.3 Set Functions

Let E be a finite set. A set function, f, defined over E assigns a value to every subset of E, i.e., $f : 2^E \to \mathbb{R}$. The following properties can be defined for a set function.

Definition 2.3.1 (Normalized)**.** The function, f, is normalized if $f(\emptyset) = 0$. ■

Definition 2.3.2 (Monotonicity)**.** The function, f, is monotone non-decreasing if for all $A \subseteq B \subseteq N$, $f(A) \leq f(B)$. Similarly, it is monotone non-increasing if $f(A) \geq f(B)$. If neither of these conditions hold, then the function is non-monotone. ■

Definition 2.3.3 (Submodularity)**.** The function f is submodular if

$$f(S) + f(T) \geq f(S \cup T) + f(S \cap T),$$

for all $S, T \subseteq N$. ■

Submodular functions satisfy the property of *diminishing marginal returns*. That is, the contribution of any element x to the total value of a set decreases as the set gets bigger. More formally, let $\Delta_f(B|A) := f(A \cup B) - f(A)$. Then,

$$\Delta_f(x|A) \geq \Delta_f(x|B), \quad \forall A \subseteq B \subseteq N.$$

Henceforth, the subscript f will be omitted unless there is ambiguity.

Since the domain of f is 2^N, there are an exponential number of possible values for the set function. We will assume that $f(S)$, for any $S \subseteq N$, is determined by a black box function. This *value oracle* is assumed to run in polynomial time in the size of the input set.

The class of submodular functions is fairly broad and includes linear functions. One way to measure the degree of submodularity of a function is the *curvature*. A submodular function has a curvature of $\kappa \in [0, 1]$ if for any $A \subset N$ and $x \in N \setminus A$,

$$\Delta(x|A) \geq (1 - \kappa) f(x). \tag{2.1}$$

In other words, the minimum possible marginal benefit of any element x is within a factor of $(1 - \kappa)$ of its maximum possible benefit.

2.4 Sequence Functions

A sequence function defined over a base set E is one that takes in a sequence of any length, composed of elements of E, and outputs a number in $\mathbb{R}$. Note that this is different from a set function in that the actual order of the elements in the sequence matters, i.e., $f((a, b)) \neq f((b, a))$.

The definitions in the previous section can also be applied to sequence functions. Let E be a finite set and f a function defined on sequences derived from E.

Definition 2.4.1 (Monotonicity)**.** The function, f, is monotone non-decreasing if for all subsequences A of a sequence B, i.e., $A \subseteq B$, $f(A) \leq f(B)$. Similarly, it is monotone non-increasing if $f(A) \geq f(B)$. If neither of these conditions hold, then the function is non-monotone. ■

The concept of submodularity can also be applied to sequence functions. Denote the marginal value of adding a sequence C to a sequence A as $\Delta_f(C|A) := f(A \parallel C) - f(A)$.

Definition 2.4.2 (Submodularity)**.** The function f is submodular if

$$\Delta_f(C|A) \geq \Delta(C|B),$$

for all $A \subseteq B$. ■

2.5 Submodular Function Optimization

Without any additional structure on a set function f, optimizing f subject to any constraints is generally intractable and inapproximable. However, a fairly general class of objective functions for which approximation algorithms exist is the class of normalized and monotone non-decreasing submodular set functions.

Here we present some results for optimization over independence systems. Combining results from [24], [15] and [5], we have the following.

Lemma 2.5.1. *Consider the problem of approximating the maximum valued basis of a p-system.*

1. *If the objective function is linear and non-negative, then the greedy algorithm gives a $\frac{1}{p}$ approximation.*
2. *If the objective function is submodular, non-negative and monotone non-decreasing, then the greedy algorithm gives a $\frac{1}{p+1}$ approximation.*

In [44] the authors look at maximizing a submodular function over a uniform matroid (i.e., selecting k elements from a set). They show that the greedy algorithm gives an approximation of $1 - \frac{1}{e}$. This is the best factor that can be achieved, as in [12] it is shown that to obtain a $(1 - \frac{1}{e} + \epsilon)$-approximation for any $\epsilon > 0$ is NP-hard for the maximum k-cover problem (which is the special case of a uniform matroid constraint).

In [15], the optimization problem is generalized to an independence system represented as the intersection of p matroids. The authors state that the result can be extended to p-systems. A complete proof for this generalization is given in [5]. For a single matroid constraint, an algorithm to obtain a $(1 - 1/e)$ approximation is also given in [5].

Linear functions are a special case of submodular functions (curvature is 0) so it is reasonable to expect the bound for the greedy algorithm to be a continuous function of the curvature. In [9], for a system that is the intersection of p matroids the greedy bound is shown to be $\frac{1}{p+\kappa}$. We now extend this result to p-systems and generalize to an α-approximate greedy algorithm.

Definition 2.5.2 (α-approximate greedy)**.** Given an objective function f defined over an independence system $(E, \mathcal{F})$. For $\alpha \in (0, 1]$, an α-approximate greedy algorithm greedily constructs an approximation to the maximum value basis by selecting at each iteration i an element g_i such that

$$\Delta(g_i|G_{i-1}) \geq \alpha \max_{\substack{e \in E \setminus G_{i-1} \\ G_{i-1} \cup \{e\} \in \mathcal{I}}} \Delta(e|G_{i-1}),$$

where $G_i = \bigcup_{j=0}^{i} g_j$ and $g_0 = \emptyset$. ■

Theorem 2.5.3. *Consider the problem of maximizing a monotone submodular function f with curvature κ, over a p-system. Then, the α-approximate greedy algorithm gives an approximation factor of $\frac{\alpha}{p+\alpha\kappa}$.*

Proof. The proof is inspired from the proof where the system is the intersection of p integral polymatroids [9, Theorem 6.1]. Let W be the optimal set. Let s_i be the element chosen by the greedy algorithm at iteration i. For $t = 1, \ldots, k$, let $S_t := \{s_1, \ldots, s_t\}$, so S_k is the final greedy solution. Also, let $\rho_t := \Delta(s_t|S_{t-1}) = f(S_t) - f(S_{t-1})$, so $f(S_k) = \sum_{t=1}^{k} \rho_t$. Therefore,

$$f(W \cup S_k) \geq f(W) + \sum_{e \in S_k} \Delta(e|(W \cup S_k) \setminus e) \geq f(W) + (1 - \kappa) \sum_{t=1}^{k} \rho_t, \tag{2.2}$$

since $\Delta(e|(W \cup S_k) \setminus e) \geq (1 - \kappa) f(e)$ by the definition of curvature, and the rest of the inequalities hold due to submodularity. Also,

$$f(W \cup S_k) \leq f(S_k) + \sum_{e \in W \setminus S_k} \Delta(e|S_k). \tag{2.3}$$

Following from the analysis of the greedy algorithm for p-systems in [5, Appendix B], a k-partition $W_1, \ldots, W_k$ of W, can be constructed such that $|W_t| \leq p$ and $\rho_t \geq \alpha\Delta(e|S_{t-1})$ for all $e \in W_t$. Therefore,

$$\sum_{e \in W \setminus S_k} \Delta(e|S_k) = \sum_{i=1}^{k} \sum_{e \in W_i \setminus S_k} \Delta(e|S_k) \leq \sum_{i=1}^{k} |W_i \setminus S_k| \frac{\rho_t}{\alpha} \leq \frac{p}{\alpha} \sum_{i=1}^{k} \rho_t, \tag{2.4}$$

since $\Delta(e|S_k) \leq \Delta(e|S_{t-1})$ for all t by submodularity.

Combining (2.3) and (2.4) with (2.2), the desired result can be derived as follows,

$$f(W) + (1-\kappa)f(S_k) \leq f(S_k) + \frac{p}{\alpha} f(S_k) \implies f(W) \leq \frac{(p + \alpha\kappa)}{\alpha} f(S_k).$$

□

For the case of a sequence submodular functions, the following lemma quantifies the greedy algorithms performance for a uniform matroid like constraint.

Lemma 2.5.4 ([1])**.** *Given a normalized monotone non-decreasing submodular sequence function f defined over the base set of elements E. The problem of selecting the maximum value sequence of size T can be approximated to within $1 - \frac{1}{e^{\alpha}}$ using an α-approximate greedy algorithm.*

2.6 Set Systems on Graphs

In this section we introduce some graph constructs and relate them to p-systems.

Given a graph $G = (V, E)$ where V is the set of vertices and E is the set of edges. We follow the standard definitions for *simple path*, *simple cycle* and *Hamiltonian cycle* (cf. [34]). We refer to a Hamiltonian cycle as a **tour**, and a simple cycle that is not a tour as a **subtour**. Let $\delta(v)$ denote the set of edges that are incident to v.

Definition 2.6.1 (Simple b-matching)**.** Given vertex capacities $b : V \to \mathbb{N}$, a **simple b-matching** is an assignment $f \in \{0,1\}^E$ to the edges such that $\sum_{e \in \delta(v)} f(e) \leq b(v)$, $\forall v \in V$. If equality holds for all v, then the b-matching is **perfect**. ■

For the rest of this thesis, any reference to a b-matching will always refer to a simple b-matching. Taking the base set of elements to be the edges of the graph, a b-matching system is defined by the collection of all subsets of edges, such that assigning a 1 only to the edges in the subset satisfies the definition of a b-matching.

Lemma 2.6.2 ([41]). *A b-matching system is a 2-extendible system.*

Given a complete graph, each edge in the graph can be assigned a cost given by $c : E \to \mathbb{R}_+$. The classic Maximum Traveling Salesman Problem (Max-TSP) is to find a maximum cost tour. On a directed graph, the TSP can be divided into two variants: the Asymmetric TSP (ATSP) – where for two vertices u and v, $c(u, v) \neq c(v, u)$ – and the Symmetric TSP (STSP) – where $c(u, v) = c(v, u)$, which is the case if the graph is undirected.

The set of feasible solutions for the directed TSP can be defined using a directed (Hamiltonian) tour independence system. A set of edges is independent if they form a collection of vertex disjoint simple paths, or a complete tour. The directed tour system can be formulated as the intersection of three matroids. These are:

1. Partition matroid: Edge sets such that the in-degree of each vertex ≤ 1,
2. Partition matroid: Edge sets such that the out-degree of each vertex ≤ 1,
3. The 1-graphic matroid: the set of edges that form a forest with at most one simple cycle.

Lemma 2.6.3 ([41]). *The directed tour independence system is 3-extendible.*

For an undirected graph, one option is to double the edges and formulate the problem as a STSP. Since the STSP is just a special case of the ATSP, the above formulation applies. Instead, directly defining an undirected tour independence system for an undirected TSP can lead to a stronger classification. In this case, a set of edges is independent if the induced graph satisfies the two conditions:

1. each vertex has degree at most 2,
2. there are no subtours.

Theorem 2.6.4. *The undirected tour system is 3-extendible.*

Proof. To show this, we can consider all the cases to show that the system satisfies the definition of a 3-extendible system. Specifically, assume some given set $A \subset B \in \mathcal{F}$ and determine the number of edges that will need to be removed from $B \setminus A$ so that adding any $x = \{u, v\} \notin B$ (such that $A \cup x \in \mathcal{F}$) to B will maintain independence.

Adding an edge can violate the degree constraint on at most two vertices (specifically u and v) and/or the subtour constraint. To satisfy the degree constraint, at most one edge

will need to be removed from B for each vertex (since any vertex in B would have a degree at most 2 before adding x). To satisfy the subtour requirement, at most one edge will need to be removed from the subtour in order to break it into a simple path. Therefore, up to three edges will have to be removed in total which means that the system is 3-extendible.

One case where exactly three edges will have to be removed comes about if A contains an edge, e_1, incident to u and another, e_2, to v. If adding x to B violates both conditions of independence then we know there exists a simple path $P \subseteq B$, where the endpoints of the path are u and v. Assume that both $e_1, e_2 \in P$. Then one edge (that is not e_1 or e_2) will have to be removed from P to break the cycle (produced by adding x) and two more will need to be removed to satisfy the degree requirements at u and v. □

As a result, the undirected tour is a 3-system. This result is, however, somewhat redundant for our purposes since a stronger result exists.

Lemma 2.6.5 (Jenkyns, [31])**.** *On a graph with n vertices, the undirected tour is a p-system with $p = 2 - \left\lfloor \frac{n+1}{2} \right\rfloor^{-1} < 2$.*

Note that although by definition of a p-system the value of p does not have to be an integer, we will assume it is. Since a p-system is also a $(p + k)$-system (for $k \geq 0$), this assumption does not eliminate any p-systems since the value of p can just be rounded up to the nearest integer (as in Lemma 2.6.5).

2.7 Review of Linear Algebra Concepts

Here we give a quick overview of concepts and notations that will be used throughout this thesis.

Given a square matrix $A \in \mathbb{R}^{n \times n}$. This can also be viewed as a transform that maps vectors in $\mathbb{R}^n$ to $\mathbb{R}^n$, i.e., $A : \mathbb{R}^n \to \mathbb{R}^n$. The identity matrix will be referred to as I.

The **range space** or **image**, $\mathcal{R}(A)$, is all the vectors $y \in \mathbb{R}^n$ such that there exists an x for which $y = Ax$.

The **null space**, $\mathcal{N}(A)$, is the set of all the vectors x such that $Ax = 0$.

The **rank** of A, $\text{rank}(A)$, is the dimension of the range space, or the number of linearly independent (LI) columns of A.

Let (λ_i, v_i) be the eigenvalue-eigenvector pairs of A. If the geometric multiplicity of every eigenvalue is equal to its algebraic multiplicity, then A can be diagonalized. In

this case, the eigenvectors of A will span $\mathbb{R}^n$. If the geometric multiplicity is less than the algebraic multiplicity for any eigenvalue, then A is **defective**. In this case, the **generalized eigenvectors** of A have to be constructed. The chain of generalized eigenvectors coming from $v_i =: v_i^{(1)}$ satisfies

$$(A - \lambda_i)v_i^{(j)} = v_i^{(j-1)} \implies Av_i^{(j)} = \lambda_i v_i^{(j)} + v_i^{(j-1)}.$$

Using this, we can derive a representation of $f(A)v_i^{(j)}$ for some function f of A,

$$f(A)v_i^{(j)} = f(\lambda_i)v_i^{(j)} + f^{(1)}(\lambda_i)v_i^{(j-1)} + \cdots + f^{(j-1)}(\lambda_i)v_i^{(1)},$$

where $f^{(s)}(\lambda_i)$ is the s^{th} derivative of $f(\lambda_i)$. This can be applied to calculate, for example, $A^k v_i^{(j)}$ by taking $f(\lambda_i) = \lambda_i^k$.

(P1) **Cayley-Hamilton Theorem:** [25, Thm. 2.4.2] If $p_A(t)$ is the characteristic polynomial of $A \in \mathbb{C}^{n\times n}$, then $p_A(A) = 0$. As a consequence of this theorem, $A^n = \sum_{i=0}^{n-1} \alpha_i A^i$.
This can further be applied recursively to get that $(\forall l > n)$, A^l is a linear combination of $A^0, \ldots, A^{n-1}$.

(P2) Given a full column rank matrix $A \in \mathbb{R}^{m\times n}$ and $k \leq n$ LI vectors $\{x_i\}_{i=1}^k$. Then $\{Ax_i\}_{i=1}^k$ are also LI.
This can be shown by contradiction. If the transformed vectors were not LI, then $\sum \alpha_i Ax_i = 0$ for some α_i. So $A(\sum \alpha_i x_i) = Ay = 0$ which can happen only if $y = \sum \alpha_i x_i = 0$. This contradicts the LI of x_i.

The following are identities that can be used to calculate the determinant or inverse of a matrix subject to a "perturbation".

(P3) **Matrix Determinant Lemma:** Given $A_{n\times n}$, $U_{n\times k}$ and $V_{n\times k}$. Assuming A^{-1} exists,

$$\det(A + UV^\top) = \det(I + V^\top A^{-1}U)\det(A).$$

For the special case of $A = I$, this is just *Sylvester's Theorem of Determinants*.

(P4) **Woodbury matrix identity (matrix inversion lemma)**: Given $A_{n\times n}$, $U_{n\times k}$, $C_{k\times k}$ and $V_{k\times n}$. Then

$$(A + UCV)^{-1} = A^{-1} - A^{-1}U\left(C^{-1} + VA^{-1}U\right)^{-1}VA^{-1},$$

assuming all inverses exist.

A symmetric matrix A is denoted to be positive definite (p.d.) as $A \succ 0$ and positive semi-definite (p.s.d.) as $A \succeq 0$.

(P5) [25, Cor. 7.7.4] If $M \succeq N \succ 0$ then $\lambda_i^M \geq \lambda_i^N$ when the eigenvalues of both matrices are sorted in descending order. This in turn implies that $\det(M) \geq \det(N)$.

(P6) Given $A \succeq 0$. Using Cholesky Decomposition, $A = LL^\top$ where L is a lower triangular matrix. If $A \succ 0$, then L is unique and has strictly positive diagonal entries. Also, any p.s.d. matrix A has a unique p.s.d. "square root" $A^{1/2}$ that is invertible if and only if A is invertible.

(P7) [25, Obs. 7.1.6] Given that $A \succeq 0$. For any $C \in \mathbb{C}^{n\times m}$, $C^*AC \succeq 0$. As a generalization, $A \succeq B \implies C^*AC \succeq C^*BC$.

2.8 Observability and Detectability

2.8.1 Time-Invariant Systems

Consider the discrete-time linear time invariant (LTI) system

$$\begin{aligned} x_{k+1} &= Ax_k, \quad x_k \in \mathbb{R}^n \\ y_k &= Cx_k, \quad y_k \in \mathbb{R}^m. \end{aligned}$$

A known problem is that of determining the value of the initial state given a sequence outputs or measurements, y_k. [1] Recognizing that the solution to the recursive equation is just $x_k = A^k x_0$ gives an exact expression for what the measurement vector at each time step is. Specifically,

$$\begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_k \end{bmatrix} = \begin{bmatrix} C \\ CA \\ \vdots \\ CA^k \end{bmatrix} x_0 =: M_k x_0.$$

This is a system of linear equations that need to be solved in order to obtain a value for x_0. Assuming all the measurements are consistent with one another, the value of x_0 can

[1] The more well known representation of the problem includes input(s) that affect the state at each time step. This has been omitted as we are not interested in such a system.

be uniquely determined if and only if M_k is full rank. In this case, the system is said to be **observable**.

The minimum value of k required for observability will vary depending on the exact system under consideration. Since x_0 has n unknown entries, it is reasonable to expect that at most n measurements should be needed for observability. Indeed, by the Cayley-Hamilton theorem (P1) it is easy to show that $\text{rank}(M_j) = \text{rank}(M_{n-1})$ for all $j \geq n-1$. Thus, we have the following test for observability.

Lemma 2.8.1 (Observability)**.** *An LTI system (A, C) is observable if and only if its **observability matrix**, Θ, has rank n, where*

$$\Theta := \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{n-1} \end{bmatrix}.$$

For many practical purposes, it is not necessary for the system to be completely observable. In this case, the observability matrix will not be full rank, i.e., $\mathcal{N}(\Theta) \neq \emptyset$. Such a system can be decomposed into observable and unobservable components. This is known as the **standard form for unobservable systems**.

$$\begin{gathered} z = T^{-1}x = \begin{bmatrix} z^1 \\ z^2 \end{bmatrix}, \qquad \bar{A} = T^{-1}AT = \begin{bmatrix} A_{\bar{o}} & A_{12} \\ 0 & A_o \end{bmatrix}, \qquad \bar{C} = CT = \begin{bmatrix} 0 & C^o \end{bmatrix}, \\ \therefore z_{k+1} = \bar{A}z_k, \qquad y_k = \bar{C}z_k, \end{gathered} \tag{2.5}$$

where $T = \begin{bmatrix} T_{\bar{o}} & T' \end{bmatrix}$ such that $T_{\bar{o}}$ is a matrix whose columns form a basis for $\mathcal{N}(\Theta)$ and T' is a matrix whose columns are linearly independent to those in $T_{\bar{o}}$, so $\text{rank}(T) = n$. The system (2.5) is equivalent to the original system. Some interesting points to note:

- Since (2.5) is just a similarity transform, the eigenvalues of A and $\bar{A}$ coincide.
- The modes (eigenvalues) of A_o are observable (i.e., (A_o, C^o) is observable) whereas the modes of $A_{\bar{o}}$ are not.
- If Θ, $\bar{\Theta}$ and Θ^o are the observability matrices of (A, C), $(\bar{A}, \bar{C})$ and (A_o, C^o) respectively, then $\bar{\Theta} = \Theta T = \begin{bmatrix} 0 & \Theta^o \end{bmatrix}$.
- The unobservable subspace is A-invariant (and is in fact the largest A-invariant subspace contained in the null space of C).

Since the stable modes will die out exponentially, for most purposes it suffices to be able to predict the value of the unstable modes. This generalization of observability is known as detectability.

Definition 2.8.2 (Detectability)**.** The following are equivalent for a LTI system:

1. (A, C) is detectable.
2. The null space of the observability matrix is a subset of the stable spectral subspace of A.
3. $A_{\bar{o}}$ is stable, i.e., no unstable mode is unobservable.
4. $\text{rank}(\begin{bmatrix} A - \lambda I \\ C \end{bmatrix}) = n$ for every eigenvalue λ of A with $|\text{Re}(\lambda)| \geq 1$.
5. For every eigenvector v of A associated with an unstable eigenvalue, $Cv \neq 0$.

■

2.8.2 Time-Varying Systems

Now consider the linear time-varying (LTV) system

$$\begin{aligned} x_{k+1} &= A_k x_k, \quad x_k \in \mathbb{R}^n \\ y_k &= C_k x_k, \quad y_k \in \mathbb{R}^m. \end{aligned}$$

The State Transition Matrix (STM) for $t_2 \geq t_1$ is $\Phi_{t_2,t_1} = \Phi_{t_2,t_2-1}\Phi_{t_2-1,t_1}$ where $\Phi_{t+1,t} = A_t$. We can define the **sequence observability matrix**,

$$B(t, t+k) = \begin{bmatrix} C_t \\ C_{t+1}\Phi_{t+1,t} \\ \vdots \\ C_{t+k}\Phi_{t+k,t} \end{bmatrix},$$

as well as the observability **Gramian**,

$$X(t, t+k) = \sum_{i=0}^{k} \Phi_{t+i,t}^\top C_{t+i}^\top C_{t+i} \Phi_{t+i,t} = B(t, t+k)^\top B(t, t+k).$$

Although a generalization of observability can be made, we are interested in a slightly stricter notion.

Definition 2.8.3 (Uniform Detectability and Observability)**.** Given a LTV system with STM Φ_{t,t_0} and measurement matrices C_t. The system is uniformly detectable if there exists non-negative integers s, r and constants $\alpha \in [0, 1)$ and $\beta > 0$, such that for all $x \in \mathbb{R}^n$ and all times t,

$$||\Phi_{t+r,t}x|| \geq \alpha||x|| \implies x^\top X(t, t+s)x \geq \beta||x||^2. \tag{2.6}$$

Additionally, the system is uniformly observable if there exists integer s and positive constants β_1, β_2 such that

$$0 \prec \beta_1 I \preceq X(t, t+s) \preceq \beta_2 I,$$

holds in the positive semidefinite sense. ■

Remark 2.8.4 (Alternative representation)*.* Note that the condition in (2.6) can be written as

$$\begin{aligned} x^\top X(t, t+s)x \geq \beta||x||^2 &\iff x^\top B(t, t+s)^\top B(t, t+s)x \geq \beta||x||^2 \\ &\iff \|B(t, t+s)x\|^2 \geq \beta||x||^2. \end{aligned}$$

Using this fact, the condition of detectability can be phrased for all $\{x|\, \|x\| = 1\}$ without loss of generality (since for $x = 0$ the condition is trivially satisfied) as

$$||\Phi_{t+r,t}x|| \geq \alpha \implies \|B(t, t+s)x\| \geq \beta > 0, \tag{2.7}$$

where Φ, α and β are as defined above.

Similarly, the condition for uniform observability can be written as

$$0 < \beta_1 \leq \|B(t, t+s)x\| \leq \beta_2 \tag{2.8}$$

where β_1, β_2 and s are as defined above. •

Remark 2.8.5 (Stabilizability)*.* Uniform stabilizability can be similarly defined. The definition is omitted here and can be found in [3]. •

In the special case that the system is time-invariant, detectability and uniform detectability are actually equivalent concepts.

Lemma 2.8.6. *A linear time-invariant system is detectable if and only if it is uniformly detectable.*

Proof. This is shown in [22, Appendix B]. Combining some ideas, we give a slightly shorter proof here. For a time-invariant system, $B(t, t+s) =: B_s$ for all t.

$\underline{\Longleftarrow}$: Given that uniform detectability holds. Assume that the system is not detectable, i.e., for some unstable eigenvector v_i of A, $Cv_i = 0$. Then $B_s v_i = 0 \implies \|B_s v_i\| = 0$.

Now, $\|A^r v_i\| = \|\lambda_i^r v_i\| = |\lambda_i|^r \|v_i\|$. Since $|\lambda_i| \geq 1$, $|\lambda_i|^r \|v_i\| \geq \alpha \|v_i\|$ for all possible (r, α). So from (2.6), $\|B_s v_i\| \geq \beta > 0$ which is a contradiction.

$\underline{\Longrightarrow}$: Given that (A, C) is detectable, which means that $(\bar{A}, \bar{C})$ is also detectable (cf. (2.5)). We need to show the existence of appropriate (s, r, α, β). For notation refer to (2.5). Also, assume without loss of generality that $\|z\| = 1$.

There are 2 cases. If $z^2 \neq 0$, then it is possible that $(\exists(r, \alpha))$ such that $\left\|\bar{A}^r z\right\| \geq \alpha \|z\|$. So it is required to show that there exist (s, β) such that $\|B_s z\|^2 \geq \beta$. Let $\bar{\Theta}$ and Θ^o be the n-step observability matrices of $(\bar{A}, \bar{C})$ and (A_o, C^o) respectively (so $B_n = \bar{\Theta}$). Since $\bar{\Theta} = \begin{bmatrix} 0 & \Theta^o \end{bmatrix}$, $\bar{\Theta} z = \Theta^o z^2$. However, since (A_o, C^o) is observable, Θ^o is full rank, which means that $\Theta^o z^2 \neq 0$. As a result, $\|B_n z\|^2 = \left\|\bar{\Theta} z\right\| > 0$, or in other words, there has to exist a $\beta > 0$ such that the desired condition is satisfied. Specifically,

$$s \geq n, \qquad 0 < \beta \leq \min_{\|z^2\|=1} \left\|\Theta^o z^2\right\|,$$

where the minimization is that of a continuous function (vector norm) over a compact set and therefore the minimum will be attained in that set by the extreme value theorem.

If $z^2 = 0$, then $\|B_s z\|^2 \geq \beta$ can not be satisfied since the $\|B_s z\|^2 = 0$. Note that the system dynamics are affected only by $A_{\bar{o}}$ which, by the definition of detectability, is stable. So $\left\|\bar{A}^r z\right\| = \|A_{\bar{o}}^r z^1\| \to 0$ as r increases. In other words, for any $\alpha \in (0, 1)$, there exists a valid r such that $\left\|\bar{A}^r z\right\| < \alpha$. More rigorously, take $P = \begin{bmatrix} v_1, \ldots, v_n \end{bmatrix}$ where v_i are the normalized generalized eigenvectors of $\bar{A}$. Let $a = P^{-1} z$, so $\|a\|_1 = \sum |a_i| \leq \|P^{-1}\|_1 =: b$. Now pick r such that $|\lambda_i|^r < |\lambda_i|^{r-n} < \frac{\alpha}{bn}$ for all stable eigenvalues of $\bar{A}$, i.e., all i such that $|\lambda_i| < 1$. Note that z can be written as a linear combination of only eigenvectors of $\bar{A}$ that correspond to stable eigenvalues. Therefore, if $\bar{A}$ is diagonalizable,

$$\left\|\bar{A}^r z\right\| = \left\|\bar{A}^r \sum a_i v_i\right\| = \left\|\sum a_i \lambda_i^r v_i\right\| \leq \sum |a_i| \, |\lambda_i|^r \, \|v_i\| < \sum |a_i| \, \frac{\alpha}{bn} \leq \alpha.$$

If $\bar{A}$ is defective, then the second equality above will not hold since $A^r v \neq \lambda^r v$ if v is a generalized eigenvector. However, since r satisfies $|\lambda_i|^{r-n} < \frac{\alpha}{bn}$, the argument being made still holds. Picking (r, α) in this way means that $B_s z$ will never be 0 if $\left\|\bar{A}^r z\right\| \geq \alpha$. $\square$

2.9 Kalman Filter

A Kalman filter uses noisy measurements to estimate the state in a linear dynamical system. An in depth study can be found in [2]. We give a quick overview in this section as well as some stability results.

Given the following stochastic LTV system:

$$\begin{aligned} x_{t+1} &= A_t x_t + w_t \\ y_{t+1} &= C_t x_t + v_t, \end{aligned}$$

where

- $t \in \mathbb{Z}_{\geq 0}$, $x_t \in \mathbb{R}^n$ and $y_t \in \mathbb{R}^m$,
- $A_t \in \mathbb{R}^{n \times n}$ and $C_t \in \mathbb{R}^{m \times n}$ are bounded,
- w_t (state noise) and v_t (process noise) are zero mean Gaussian noise vectors with covariance matrices $W_t \in \mathbb{R}^{n \times n} \succeq 0$ and $V_t \in \mathbb{R}^{m \times m} \succ 0$ respectively. Assume that the noise vectors are independent.

The Kalman filter works in two steps. The *Measurement Update* finds an estimate of the current state using the previous state and all measurements including the current one. This gives the *a posteriori* estimate.

$$\hat{x}_{t|t} = \hat{x}_{t|t-1} + \Sigma_{t|t-1} C_t^\top (C_t \Sigma_{t|t-1} C_t^\top + V_t)^{-1} (y_t - C_t \hat{x}_{t|t-1}) \tag{2.9}$$

$$\Sigma_{t|t} = \Sigma_{t|t-1} - \Sigma_{t|t-1} C_t^\top (C_t \Sigma_{t|t-1} C_t^\top + V_t)^{-1} C_t \Sigma_{t|t-1} \tag{2.10}$$

Applying the Woodbury matrix identity (P4), the covariance update can be alternatively written as

$$\Sigma_{t|t} = \left(\Sigma_{t|t-1}^{-1} + C_t^\top V_t^{-1} C_t \right)^{-1} \tag{2.11}$$

The *Time Update* finds an estimate of the state using the previous state and previous measurements. This gives the *a priori* estimate.

$$\begin{aligned} \hat{x}_{t+1|t} &= A_t \hat{x}_{t|t} \\ \Sigma_{t+1|t} &= A_t \Sigma_{t|t} A_t^\top + W_t \end{aligned}$$

The two steps can be combined into one to get

$$\Sigma_{t+1|t} = A_t \Sigma_{t|t-1} A_t^\top + W_t - A_t \Sigma_{t|t-1} C_t^\top (C_t \Sigma_{t|t-1} C_t^\top + V_t)^{-1} C_t \Sigma_{t|t-1} A_t^\top, \tag{2.12}$$

which is a Riccati recursion with initial condition $\Sigma_{0|-1} = \Sigma_0$. Since this does not use the actual measurement, the next step predicted covariance can be calculated before actually making any observations.

An interesting question is under what conditions the filter is stable, i.e., the expected error of the state estimate goes to zero. This is answered in the following lemmas derived from [3].

Lemma 2.9.1. *Given that A_t and C_t are bounded and that the system (A_t, C_t) is uniformly detectable, the Kalman filter error covariance, $\Sigma_{t|t}$, and predictor covariance, $\Sigma_{t+1|t}$, are bounded.*

Lemma 2.9.2. *Given that A_t and C_t are bounded and that the system (A_t, C_t) is uniformly detectable and $(A_t, W_t^{\frac{1}{2}})$ is uniformly stabilizable, the Kalman filter is exponentially stable.*

Therefore, uniform detectability is sufficient for the Kalman filter error covariance to be bounded and for the estimation error to die out. The next result establishes necessity.

Lemma 2.9.3. *Given that A_t and C_t are bounded and that the system $(A_t, W_t^{\frac{1}{2}})$ is uniformly stabilizable. If the optimal filter error covariance is bounded then (A_t, C_t) is uniformly detectable.*

Chapter 3

Informative Path Planning as a Maximum Traveling Salesman Problem with Submodular Rewards

The organization of this chapter is as follows. In Section 3.1 we formalize the submodular max TSP problem. In Section 3.2 we analyze two different strategies for approximating a solution; namely, a greedy approach and a 2-matching based tour. We also discuss extending Serdyukov's algorithm. In Section 3.3 we extend our algorithms to the case where the graph is directed. In Section 3.4 we discuss a method to incorporate costs into the optimization. Finally, we provide some simulation results in Section 3.5 and demonstrate an example application in monitoring.

3.1 Problem Formulation

Given a complete graph $G = (V, E, w)$, where a set of edges has a reward or utility given by the normalized monotone non-decreasing submodular rewards function $w : 2^E \to \mathbb{R}_{\geq 0}$ that has a curvature of κ. We are interested in analyzing simple algorithms to find a Hamiltonian tour that has the maximum reward:

$$\max_{S \in \mathcal{H}} w(S), \tag{3.1}$$

where $\mathcal{H}$ is the set of all Hamiltonian tours on the graph G. In Section 3.4, we will briefly discuss the problem where costs are incorporated into the optimization problem; that is,

where the graph contains both edge weights, representing travel costs, and edge rewards, representing information gain.

3.2 Algorithms for the Submodular Max-TSP

In this section we present some algorithms for approximating the submodular Max-TSP, i.e., approximating (3.1). We first describe a technique to generalize any known approximation algorithm, that uses a modular objective function, by performing a linear relaxation of the submodular function. Next, we analyze a greedy algorithm and a 2-matching based algorithm and provide complexity and approximation bounds for each. Finally, we discuss extending Serdyukov's algorithm to the submodular setting.

3.2.1 Linear Relaxation

One possibility for constructing an algorithm for a submodular objective function is to just use any algorithm that is for a modular objective. In order use this approach, the submodular function needs to be approximated by a modular function.

We define a linear relaxation $\tilde{w}$ of the submodular function w as follows,

$$\tilde{w}(S) = \sum_{e \in S} w(e) = \sum_{e \in S} \Delta(e|\emptyset), \quad \forall S \subseteq E. \tag{3.2}$$

In other words, each edge is assigned its maximum possible marginal benefit. Using this relaxation, any known algorithm to approximate the maximum tour can be applied. The question arises as to what the bound is for the value of the final tour.

Theorem 3.2.1. *Consider an independence system $(E, \mathcal{I})$ over which a submodular rewards function, w, is defined, with a curvature of κ. Denote the linear relaxation of w as $\tilde{w}$. Let M_O^s and M_O^r be the maximum value bases with respect to w and $\tilde{w}$ respectively. For some set M_1, if $\tilde{w}(M_1) \geq \alpha \tilde{w}(M_O^r)$ for $\alpha \in [0, 1]$, then*

$$w(M_O^s) \geq w(M_1) \geq (1 - \kappa)\alpha w(M_O^s).$$

Proof. The definition of curvature states that $\Delta(e|S) \geq (1-\kappa)w(e)$ for $S \subset E$ and $e \in E \backslash S$.

Using this and the definition of submodularity,

$$\begin{aligned} w(M_1) &= \Delta(e_1|\emptyset) + \Delta(e_2|\{e_1\}) + \Delta(e_3|\{e_1, e_2\}) + \cdots \\ &\geq (1-\kappa) \sum_{e \in M_1} w(e) = (1-\kappa)\tilde{w}(M_1) \\ &\geq (1-\kappa)\alpha\tilde{w}(M_O^r) \end{aligned}$$

Since $\tilde{w}(M_O^r)$ is maximum, $\tilde{w}(M_O^r) \geq \tilde{w}(M_O^s)$. Therefore,

$$w(M_1) \geq (1-\kappa)\alpha\tilde{w}(M_O^s) \geq (1-\kappa)\alpha w(M_O^s),$$

where the last inequality holds due to decreasing marginal benefits. □

Although the maximum value of the element is easy to evaluate, it could be a gross overestimate of the actual contribution of the element to a set. As a result, this method works better for lower values of curvature.

3.2.2 A Simple Greedy Strategy

A greedy algorithm to construct the TSP tour is given in Algorithm 1. The idea is to pick the edge that will give the largest marginal benefit at each iteration. The selected edge cannot cause the degree of any vertex to be more than 2 nor create any subtours.

Theorem 3.2.2. GREEDYTOUR *(Algorithm 1) gives a* $\frac{1}{2+\kappa}$ *approximation of the optimal tour and has a complexity of* $O(|V|^3(f + \log|V|))$*, where* f *is the runtime of the function oracle.*

Proof. By Lemma 2.6.5 and Theorem 2.5.3, Algorithm 1 is a $\frac{1}{2+\kappa}$-approximation of (3.1).

At each iteration, the marginal benefit of edges not yet selected are recalculated and sorted (line 3). Since recalculation need only be done when the set M changes, and only one edge is added to the partial tour M at each iteration, recalculation only needs to take place a total of $|V|$ times. This dominates the runtime and has a complexity of $O(|V|(|E|f + |E|\log|E|))$. □

Remark 3.2.3. The detection of subtours can be done using disjoint-sets for the vertices where each set represents a group of vertices that are in the same subtour. The overall runtime for detecting subtours adds up to $|V|^2 \log|V|$ following the analysis in [10, Ch. 21,23]. The "recalculation" part dominates the total runtime and so the exact method used to check for validity of edges does not have a significant effect on the runtime. •

Algorithm 1: GREEDYTOUR

Input: Graph $G = (V, E)$. Function oracle $w : 2^E \to \mathbb{R}_{\geq 0}$
Output: Edge set M corresponding to a tour.

```
1 M ← ∅
2 while E ≠ ∅ and |M| < |V| do
3     if M was updated then recalculate Δ(e|M), ∀e ∈ E
4     e_m ← argmax_{e∈E} Δ(e|M)
5     Determine if e_m is valid by checking vertex degrees and checking for subtours
6     if e_m is valid then M ← M ∪ {e_m}
7     E ← E \ {e_m}
8 return M
```

Motivated by the reliance of the bound on the curvature, in the next section we will consider a method to obtain improved bounds for objective functions with a lower curvature.

3.2.3 2-Matching Based Tour

Another approach to finding the optimal basis of an undirected tour set system is to first relax the "no subtours" condition. The set system defined by the independence condition that each vertex can have a degree at most 2 is in fact just a 2-matching system. As before, finding the optimal 2-matching for a submodular function is a NP-hard problem. We discuss two methods to approximate a solution. The first is a greedy approach and the second is by using a linear relaxation of the submodular function. We will see that the bounds with linear relaxation will be better than the greedy approach for certain values of curvature.

Greedy 2-Matching

One way to find an approximate maximum 2-matching is to use a greedy approach similar to GREEDYTOUR, except there is no need to check for subtours. We refer to this as the GREEDYMATCHING algorithm. The algorithm is exactly the same as Algorithm 1 except for a single change:

5 Determine if e_m is valid by checking vertex degrees.

Theorem 3.2.4. *The* GREEDYMATCHING *algorithm gives a* $\frac{1}{2+\kappa}$*-approximation of the optimal 2-matching and has a complexity of* $O(|V|^3(f + \log|V|))$*, where* f *is the runtime of the function oracle.*

Proof. A simple 2-matching is a 2-extendible system (Lemma 2.6.2), so the greedy solution will be within $\frac{1}{2+\kappa}$ of the optimal (Theorem 2.5.3). The runtime analysis is similar to that of GREEDYTOUR. □

Maximum 2-Matching Linear Relaxation

For a linear objective function, the problem of finding a maximum weight 2-matching can be formulated as a binary integer program. Let $x = \{x_{ij}\}$ where $1 \leq i < j \leq |V|$ and let each edge be assigned a real positive weight given by $\tilde{w}_{ij}$. Define $E(x)$ as the set of edges for which $x_{ij} = 1$. Then the maximum weight 2-matching, $(V, E(x))$, can be obtained by solving

$$\begin{aligned} \max & \sum_{i=1}^{|V|-1} \sum_{j>i} \tilde{w}_{ij} x_{ij} \\ \text{s.t.} & \sum_{j>i} x_{ij} + \sum_{j<i} x_{ji} = 2, \quad \forall i \in \{1, \ldots, |V|\} \\ & x_{ij} \in \{0, 1\}, \quad 1 \leq i < j \leq |V|. \end{aligned}$$

Alternatively, for a weighted graph the maximum weight 2-matching can be found in $O(|V|^3)$ time [34] via an extension of Edmonds' Maximum Weighted Matching algorithm.

For our original problem with (3.1) as the objective function for the maximization, the two methods for constructing an optimal 2-matching described here can not be applied directly. Therefore, we use the linear relaxation $\tilde{w}$ of the submodular function w. The optimal 2-matching based on the weights $\tilde{w}$ can be calculated and this 2-matching will be within $(1 - \kappa)$ of the optimal 2-matching based on w by Theorem 3.2.1.

Reduced 2-Matching

The output of either of the two algorithms described will be a basis of the 2-matching system. Once a maximal 2-matching has been obtained, it needs to be converted into a tour. The edge set corresponding to the 2-matching can be divided into a collection of

vertex-disjoint sets of edges. At most one of these will consist of a simple path, which will contain at most two vertices (i.e., one edge); the rest will be subtours. If any simple path consisted of more than one edge, its endpoints could be joined together contradicting the maximality of the 2-matching.

In order to convert the maximal 2-matching to a tour, the subtours are broken by removing an edge from each one. The remaining set of simple paths are then connected up.

Theorem 3.2.5. *Given a submodular function f defined over a set E, and k disjoint subsets $\{E_i\}_{i=1}^k$ of E, at least one of these subsets satisfies $f(E \setminus E_i) \geq \left(1 - \frac{1}{k}\right) f(E)$.*

Proof. Consider the marginal value of each set E_i. At least one of these sets will have a marginal value that is less than the average of the marginal values of all the sets. Without loss of generality, let E_1 be such that

$$\Delta(E_1 | E \setminus E_1) \leq \frac{1}{k} \sum_{i=1}^{k} \Delta(E_i | E \setminus E_i).$$

Also, letting $E_0 = \emptyset$,

$$\sum_{i=1}^{k} \Delta(E_i, E \setminus E_i) \leq \sum_{i=1}^{k} \Delta(E_i | \bigcup_{j=1}^{i-1} E_j) = f\left(\bigcup_{i=1}^{k} E_i\right) \leq f(E).$$

Combining these two inequalities,

$$f(E) - f(E \setminus E_1) = \Delta(E_1 | E \setminus E_1) \leq \frac{1}{k} f(E),$$

and the desired result follows. □

Using this theorem, an algorithm to reduce a 2-matching can be constructed. The REDUCEMATCHING algorithm is outlined in Algorithm 2. The basic idea is to arbitrarily index the edges of each subtour T^i. Then similarly indexed edges are grouped into the same set, i.e., edge 1 of each subtour form a set, edge 2 from each subtour form another set, and so on. The algorithm then just cycles through these disjoint sets to find one that can be removed while retaining at least $\frac{2}{3}$ of the original value of the 2-matching.

Theorem 3.2.6. *Algorithm 2 reduces the 2-matching while maintaining at least $\frac{2}{3}$ of the original value of the 2-matching. The complexity of the algorithm is $O(|V|f)$.*

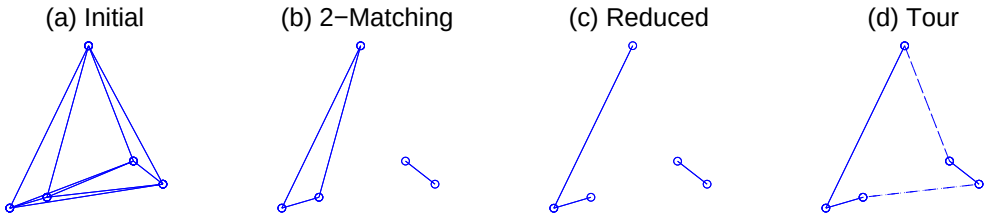


Figure 3.1: The steps in the 2-matching based tour algorithm.

Proof. Note that each subtour will consist of at least three edges and so $k \geq 3$. Assume the edges within each subtour are arbitrarily indexed. Construct a collection of sets $\{E_i\}_{i=1}^{k}$ such that E_i contains edge i from each of the subtours. These sets will be disjoint. Therefore, by Theorem 3.2.5, at least one of these sets can be removed from the 2-matching while maintaining $(1 - \frac{1}{k}) \geq \frac{2}{3}$ of the value of the 2-matching.

As the existence of such a set is guaranteed, the algorithm merely cycles through each of the sets E_i until an appropriate one is found. The number of possible sets is bounded by the size of the smallest subtour, which is at most $|V|$ (consider the case where the output of the 2-matching is a tour). □

Algorithm 2: REDUCEMATCHING

Input: A 2-matching $G_M = (V, M)$ where $M = \bigcup_{i=1}^{m} T^i$ and the sets T^i are the subtours.

Output: A set of edges to remove from the 2-matching.

Ignore all sets T^i such that $|T^i| \leq 1$. Label the remaining $n \leq m$ sets $A^1, \ldots, A^n$.
`// Let` $\{a_j^i\}_j$ `be the edges in subtour` A^i
$j \leftarrow 1$; $k \leftarrow \min_i |A^i|$
$R := \bigcup_{i=1}^{n} a_j^i$
while $w(M \setminus R) < \frac{k-1}{k} w(M)$ **do**
    $j \leftarrow j + 1$
    $R := \bigcup_{i=1}^{n} a_j^i$
return R

Tour using matching algorithm

We now present an outline of the complete 2-matching tour algorithm. The steps are illustrated in Figure 3.1.

(a) Run GREEDYMATCHING to get a simple 2-matching, M_1. Using the linear relaxation $\tilde{w}$ of w, solve for the maximum weight 2-matching, M_2. From M_1 and M_2, choose the 2-matching that has a higher value.

(b) Determine all sets of subtours.

(c) Run Algorithm 2 to select edges to remove. This results in a set of simple paths.

(d) Add edges to connect the paths together into a tour.

Theorem 3.2.7. *The 2-matching tour algorithm will output a tour that is within* $\frac{2}{3}\max\left\{\frac{1}{(2+\kappa)}, (1-\kappa)\right\}$ *of the optimal in* $O((|V|^3+|V|)f + |V|^3\log|V|)$ *time.*

Proof. Note that the optimal tour has a value less than or equal to the optimal 2-matching. The 2-matching is a $\max\{\frac{1}{2+\kappa}, 1-\kappa\}$-approximation from Theorems 3.2.4 and 3.2.1. Removing edges from the 2-matching retains atleast $\frac{2}{3}$ of the original value of the 2-matching (Theorem 3.2.6) and adding edges can only increase the value (i.e., no loss).

The greedy 2-matching takes $O(|V|^3(f+\log|V|))$ time and the linear relaxation approximation takes $O(|V|^3)$ time. Finding the edges in all the subtours is $O(|V|)$ since the number of edges in a 2-matching is at most $|V|$. Reducing the 2-matching is $O(|V|f)$. Connecting up the final graph is $O(m) = O(|V|)$ where m is the number of subtours and ranges from 1 to $\left\lfloor \frac{|V|}{3} \right\rfloor$. Therefore, the total runtime is $O(|V|^3(f+\log|V|)+|V|(2+f))$. □

A similar method of using a 2-matching is used in [16] for a linear objective function. In that case, the loss in value at the reduction step is shown to be at most $\frac{1}{3}$ of the value of the optimal 2-matching. We showed that a similar bound limiting the loss can be obtained for the submodular case; however, we see a further loss in value since the 2-matching was constructed greedily, resulting in the final tour being within $\frac{2}{3}\frac{1}{2} = \frac{1}{3}$ of the optimal. By also using the relaxation method of finding the 2-matching, our resulting bound for the final tour in the case of a linear function improves to $\frac{2}{3}$.

Remark 3.2.8 (Comparison of bounds). For any value of $\kappa < \frac{1}{2}(\sqrt{3}-1) \approx 0.366$, constructing a 2-matching and then converting it into a tour, gives a better bound with respect to the optimal tour than by using the greedy tour approach (cf. Figure 3.2). •

Remarks 3.2.9 (Heuristic improvements). i) The $\frac{2}{3}$ loss is actually a worst case bound where the smallest subtour is composed of three edges. For a given problem instance, an improved bound of $\frac{k-1}{k}$ can be obtained, where k is the size of the smallest subtour in the 2-matching.

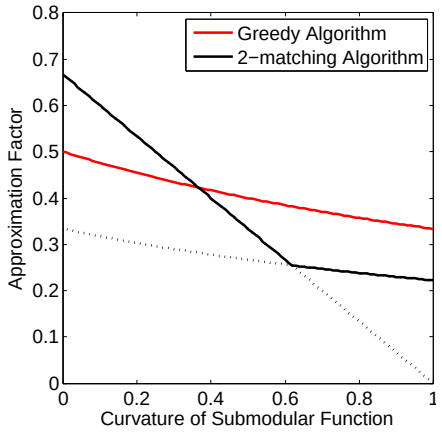


Figure 3.2: Comparison of bounds for the two algorithms.

ii) In removing the edges we used an algorithm to quickly find a "good" set of edges to remove but made no effort to look for the "best" set. Although the $\frac{2}{3}$ bound is tight, using a different heuristic we could get a better result on some problem instances (of course at the cost of a longer runtime). For instance, instead of just constructing one group of disjoint sets, a few other groups can also be randomly constructed (by labeling the edges differently) and then the best result of all the iterations can be selected. If each subtour contains three edges, then there are $3^{O(|V|)}$ possible groups. As a result, there are numerous possibilities to choose from.

iii) In the last step, instead of arbitrarily connecting up the components, completing the tour can be achieved using various different techniques. A greedy approach could be used (so running Algorithm 1 except with an initial state); this would not change the worst case runtime given in Theorem 3.2.7. Alternatively, if the number of subtours is small, an exhaustive search could be performed. •

3.2.4 Discussion on Serdyukov's Algorithm

In the case that f is a modular objective function, the 2-matching algorithm presented in [16] constructs a perfect 2-matching then reduces it by removing an edge from each subtour. In doing so, the entire value of the removed edges is lost. Serdyukov's algorithm [47] improves upon this by trying to use the edges that are removed. A brief description of the algorithm is now given for the case in which the number of vertices is even. The algorithm can be extended to the case where the graph has an odd number of vertices but that is slightly more involved.

Given a graph, an optimal perfect 2-matching, F, as well as an optimal perfect 1-matching, M, are created. Denote the optimal tour as H. The edges to be removed from the subtours of F, denoted E, are chosen in a way that they can be added to M without creating any subtours. Since $f(F) \geq f(H)$ and $f(M) \geq \frac{1}{2}f(H)$, one of the two edge sets constructed is a partial tour that is at least $\frac{3}{4}$ the value of the optimal tour.

Can this be extended to the case with submodular functions? Using a linear relaxation of the objective function results in a bound of $\frac{3}{4}(1-\kappa)$ by Theorem 3.2.1. The downside of this method is that for high curvature the bound is close to 0. As a result, we now attempt to extend the algorithm to objective functions with a high degree of submodularity.

The main idea in Serdyukov's algorithm is that the edges removed from the 2-matching do not lose their value since they are transfered over to the 1-matching. In the case that the objective function is submodular, the marginal increase in the value of the 1-matching due to the addition of these edges is unknown. Finding the best set of edges to transfer can be phrased as the constrained maximization of a non-monotone submodular function. Specifically,

$$\max_{E \subset F} f(F \setminus E) + f(M \cup E),$$

such that E contains exactly one edge from each subtour of F. Note that the constraint is not an independence system; only sets E that satisfy $|E| =$ (number of subtours) are valid, which violates the subset property of independence systems. As a result any existing approximation results do not apply. Additionally, it appears quite challenging to relate the value of resulting partial tours back to the value of the optimum tour. Therefore, an extension for high values of curvature is left as a question for future research.

3.3 Extension to Directed Graphs

The algorithms described can also be applied to directed graphs yielding approximations for the Max-ATSP.

3.3.1 Greedy Tour

For the greedy tour algorithm, a slight modification needs to be made to check that the in-degree and out-degree of the vertices are less than or equal to 1 instead of checking for the degree being less than or equal to 2. Since the directed tour is a 3-extendible system (Lemma 2.6.3), the approximation of the greedy algorithm changes to $\frac{1}{3+\kappa}$ instead of $\frac{1}{2+\kappa}$ for the undirected case.

3.3.2 Tour Using Matching

Instead of working with a 2-matching, the system can be modelled as the intersection of two partition matroids:

- Edge sets such that the indegree of each vertex ≤ 1.
- Edge sets such that the outdegree of each vertex ≤ 1.

This system is still 2-extendible (Remark 2.1.4) and so the approximation for the greedy 2-matching does not change. For the second approximation using the linear relaxation, the problem becomes the Maximum Assignment Problem (max AP). It can be solved optimally by representing the edge weights as a weight matrix $\tilde{W}$, where we set $\tilde{W}_{ii} = -\infty$, then applying the Hungarian algorithm, which has a complexity of $O(|V|^3)$ [34].

The result of the greedy algorithm or the solution to the assignment problem will be a set of edges that together form a set of subtours, with the possibility of a lone vertex. Note that a subtour could potentially consist of just two vertices. Therefore, removing one edge from each subtour will result in a loss of at most $\frac{1}{2}$ instead of $\frac{1}{3}$. This follows from Theorem 3.2.5, setting $k = 2$. The final bound for the algorithm is therefore $\max\left\{\frac{1}{2(2+\kappa)}, \frac{1}{2}(1-\kappa)\right\}$.

3.4 Incorporating Costs

Given a graph $G = (V, E, w, c)$ with edge rewards w defined as previously. Each edge has a cost given by $c : E \to \mathbb{R}_{\geq 0}$ and the cost of a set of edges is the sum of the cost of each edge in the set. We can consider the tradeoff between the reward of a set and its associated cost. A number of algorithms presented in the literature seek to maximize the reward given a "budget", k, on the cost, i.e. find a tour T such that

$$T \in \operatorname*{argmax}_{S \in \mathcal{H}} w(S) \text{ s.t. } c(S) \leq k.$$

This involves maximizing a monotone non-decreasing submodular function over a knapsack constraint as well as an independence system constraint. However, the result of these "budgeted" solutions may not be a tour.

We are interested in finding tours that balance the reward obtained with travel cost. Therefore, we will work with a different form of the objective function defined by a weighted

combination of the reward and cost. For a given value of $\beta \in [0, 1]$, solve

$$T \in \operatorname*{argmax}_{S \in \mathcal{H}} f(S, \beta) \tag{3.3}$$

$$f(S, \beta) = (1 - \beta)w(S) - \beta c(S). \tag{3.4}$$

An advantage of having this form for the objective is that the "cost trade-off" is being incorporated directly into the value being optimized. Since the cost function is modular, maximizing the negative of the cost is equivalent to minimizing the cost. So the combined objective seeks to simultaneously maximise the reward and minimize the cost.

The parameter β is used as a weighting mechanism. The case of $\beta = 0$ corresponds to ignoring costs and that of $\beta = 1$ corresponds to ignoring rewards and just minimizing the cost (i.e., the classic TSP).

Finally, we scale w and c so they share a common range of values. Therefore, the final objective function is:

$$f(S, \beta) = \frac{1 - \beta}{M_w} w(S) - \frac{\beta}{M_c} c(S) \tag{3.5}$$

where M_w and M_c are estimates of the maximum reward and cost, respectively. Henceforth, we will assume that w and c are appropriately scaled and so the constants M_w and M_c will be omitted.

The objective (3.5) is non-monotone and may be negative. To address this, consider the alternative modified cost function

$$c'(S) = |S|M - c(S), \quad M = \max_{e \in E} c(e).$$

This gives the following form for the objective function,

$$f'(S, \beta) = (1 - \beta)w(S) + \beta c'(S) = f(S, \beta) + \beta |S| M, \tag{3.6}$$

which is a monotone non-decreasing non-negative submodular function; hence, has the advantage of offering known approximation bounds. The costs have in a sense been "inverted" and so maximizing c' still corresponds to minimizing the cost c.

Remark 3.4.1. Instead of using $|S|M$ as the offset, the sum of the $|S|$ largest costs in E could be used. This would not improve the worst case bound but may help to improve results in practice. •

Lemma 3.4.2. *For any two sets S_1 and S_2, if $f'(S_1, \beta) \geq \alpha f'(S_2, \beta)$ for some $\alpha > 0$, then $f(S_1, \beta) \geq \alpha f(S_2, \beta) + \beta M(\alpha |S_2| - |S_1|)$.*

Proof. If $f'(S_1, \beta) \geq \alpha f'(S_2, \beta)$, then by definition we have $f(S_1, \beta) + \beta|S_1|M \geq \alpha(f(S_2, \beta) + \beta|S_2|M)$. This implies that $f(S_1, \beta) \geq \alpha f(S_2, \beta) + \beta M(\alpha|S_2| - |S_1|)$, completing the proof. □

Remark 3.4.3. As a special case, if $|S_1| = |S_2|$, then $f(S_1, \beta) > f(S_2, \beta)$ if and only if $f'(S_1, \beta) > f'(S_2, \beta)$. Therefore, it can be deduced that over all sets of the same size, the one that maximizes (3.4) is the same one that maximizes (3.6). •

Using this result, any α-approximate algorithm maximizing f' can be related back to a bound on maximizing f.

3.4.1 Optimization Bounds Incorporating Costs

We first look at how the result of maximizing (3.6) relates to the optimal value of (3.4). This result is then used to derive an approximation factor for the greedy algorithm.

Theorem 3.4.4. *Consider a p-system and the submodular functions f and f' as defined in* (3.4) *and* (3.6) *respectively. An α-approximation to the problem $\max_{S\in\mathcal{F}} f'(S, \beta)$, corresponds to an approximation of $\alpha OPT - \left(1 + \frac{p-2}{p}\alpha\right)\beta Mn$, for the problem $\max_{S\in\mathcal{F}} f(S, \beta)$, where* OPT *is the value of the optimal solution and n is the size of the maximum cardinality basis.*

Proof. Let S be the solution obtained by a α-approximation algorithm to f'. Let T be the optimal solution using f'. Let Z be the optimal solution using f. Note the following inequality:

$$|A| \leq |B|p \implies \alpha|B| - |A| \geq |B|(\alpha - p) \geq |A|(\frac{\alpha}{p} - 1).$$

By using the property of p-systems that for any two bases A and B, $|A| \leq p|B|$, and by applying Lemma 3.4.2 to the fact that $f'(S, \beta) \geq \alpha f'(T, \beta)$,

$$\begin{aligned} f(S, \beta) &\geq \alpha f(T, \beta) + \beta M\left(\alpha|T| - |S|\right) \\ &\geq \alpha f(T, \beta) - \beta M|S|\left(1 - \frac{\alpha}{p}\right) \geq \alpha f(T, \beta) - \beta Mn\left(1 - \frac{\alpha}{p}\right). \end{aligned}$$

Deriving a similar expression using the fact that $f'(T, \beta) \geq f'(Z, \beta)$ and substituting, we obtain

$$f(S, \beta) \geq \alpha f(Z, \beta) - \beta Mn\left(1 + \frac{p-2}{p}\alpha\right).$$

□

Remark 3.4.5. In the special case of a 1-system (this includes matroids), or more generally any problem where the output to the algorithm will always be the same size, we have $|S| = |T| = |Z|$ and also $T = Z$ following from Remark 3.4.3. This means that an algorithm that gives a relative error of α when using f' as the objective will give a normalized relative error of α when using f as the objective (i.e. $f(S, \beta) \geq \alpha \text{OPT} + (1 - \alpha)\text{WORST}$). •

Greedy with Costs

Corollary 3.4.6. *The problem* $\max_{S \in \mathcal{F}} f(S, \beta)$ *can be approximated to* $(p+1)^{-1}OPT + \beta(1+\epsilon)WORST$ *(where* $\epsilon \in [-\frac{1}{2}, 1)$ *is a function of* p*) using a greedy algorithm.*

Proof. Run the greedy algorithm using $f'(S, \beta)$ to obtain the set T_G. Let T and Z be defined as in Theorem 3.4.4. Since f' is a non-negative monotone function, $f'(T_G, \beta) \geq \frac{1}{p+1} f'(T, \beta)$. So applying Theorem 3.4.4,

$$f(T_G, \beta) \geq \frac{1}{p+1} f(Z, \beta) - \beta M n \left(1 + \frac{p-2}{p(p+1)}\right) \tag{3.7}$$

□

From Remark 3.4.3, the following can be deduced about the greedy algorithm.

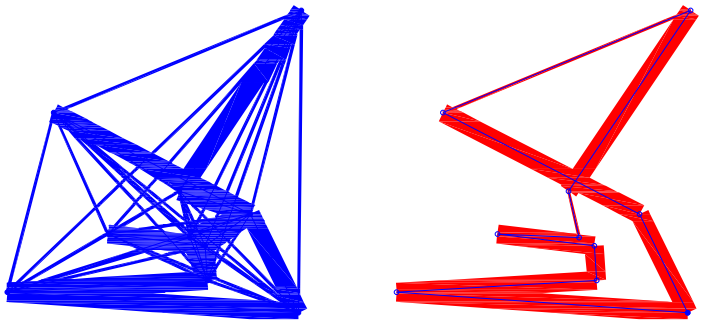
Theorem 3.4.7. *Let* G_1 *be the greedy solution obtained maximizing* (3.4) *and* G_2 *be the greedy solution maximizing* (3.6)*. Then* $G_1 = G_2$*.*

Proof. At each iteration i of the greedy algorithm, we are finding the element that will give the maximum value for a set of size $i + 1$. Since comparison is being done between sets of the same size, the same element will be chosen at each iteration. □

Summarizing, for any p-system, applying the greedy algorithm using the non-monotone objective (3.4) leads to the same set as using (3.6) and the resulting bound is given by (3.7).

3.4.2 Performance Bounds for Submodular Max-TSP

Using the proposed modification, new bounds can be derived for the algorithms discussed in this thesis. One thing to note is that in the case of the tour, all tours will have length $|V|$, even though the tour is not a 1-system. Therefore, we can apply Lemma 3.4.2 directly.



(a) Example graph. (b) Greedy solution.

Figure 3.3: An example ten vertex graph with the edges representing area cover, and the corresponding greedy approximation for the max value tour.

Theorem 3.4.8. *Using* (3.4) *as the objective function,*

- *Algorithm 1 outputs a tour that has a value of at least* $\frac{1}{3}OPT - \frac{2}{3}\beta M|V|$,
- *the tour based on a matching algorithm outputs a tour that has a value at least* $\frac{2}{9}OPT - \frac{7}{9}\beta M|V|$,

where M *is the maximum cost of any edge.*

Note that in [31] an approximation of $\frac{1}{2}(OPT + WORST)$ was given for finding a minimum TSP using a greedy approach which is better than the $\frac{1}{3}OPT + \frac{2}{3}WORST$ that we calculate for the greedy tour with costs.

3.5 Simulations and Applications

In order to empirically compare our algorithms, we have run simulations for a function that represents coverage of an environment. A complete graph is generated by randomly placing vertices over a rectangular region. Each edge in the graph is associated with a rectangle and each rectangle is assigned a different width to represent different amounts of coverage. The objective is to construct a tour that maximizes the area covered.

An example of a ten vertex graph is given in Figure 3.3a. Here we see the complete graph as well as a representation of the value of each edge given by the area of the rectangle

the edge corresponds to. The majority of edges have a low weight with a few having a much larger value. The greedy tour algorithm outputs the tour given in Figure 3.3b.

We used a slightly different implementation for the greedy algorithm in our tests. For a submodular objective function, the marginal value of each element in the base set changes every iteration and so has to be recalculated. In [42], the author presents an accelerated greedy algorithm that uses the decreasing marginal benefits property of submodularity to reduce the number of recalculations and thus improve the efficiency of the simple greedy approach. The idea is fairly simple. Given $A \subset B$ and two elements $e_1, e_2 \notin B$. If $\Delta(e_1|A) \leq \Delta(e_2|B)$, then $\Delta(e_1|B) \leq \Delta(e_2|B)$. Therefore, $\Delta(e_1|B)$ does not need to be calculated. The accelerated greedy algorithm has the same worst case bound as the naive version; however, empirical results have shown that it can achieve significant speed-up factors [42],[17].

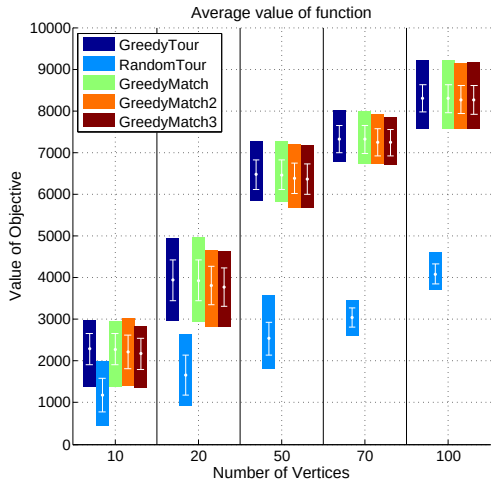
The simulations were performed on a quad-core machine with a 3.10 GHz CPU and 6GB RAM.

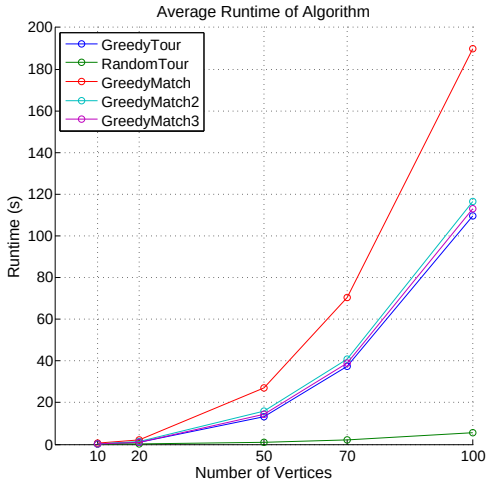
3.5.1 Comparison of Algorithms

All algorithms were run on 30 randomly generated graphs for five different graph sizes. The resulting value of the objective function was recorded and averaged over all 30 instances. The vertices were distributed uniformly randomly over a 100×100 region. The edge thickness was assigned a value of 7 with probability $2/\sqrt{|V|}$, or 1 otherwise (so $O(|V|)$ of the edges had a high reward). The results are shown in Figure 3.4. For a second simulation (Figure 3.5), the set up was the same except the edges thickness were distributed uniformly over $[0, 7]$ and a total of 40 instances were averaged.

The algorithms compared are:

- GreedyTour (GT): The greedy algorithm for constructing a tour.
- RandomTour (RT): Randomly select edges to construct a tour.
- For the 2-matching based algorithm, three possibilities are considered. All three start off by greedily constructing a 2-matching.
  - GreedyMatching (GM): Remove from each subtour the element that will result in the least loss to the total value, then greedily connect up the tour.
  - GreedyMatching2 (GM2): Use Algorithm 2 to reduce the matching, then greedily connect up the tour.





	GT	RT	GM	GM2	GM3
10	27 (3)	0 (0)	25 (1)	16 (1)	2 (0)
20	23 (8)	0 (0)	21 (5)	8 (0)	1 (0)
50	26 (16)	0 (0)	14 (4)	5 (0)	0 (0)
70	27 (18)	0 (0)	12 (3)	3 (0)	1 (0)
100	27 (16)	0 (0)	14 (3)	2 (0)	1 (0)

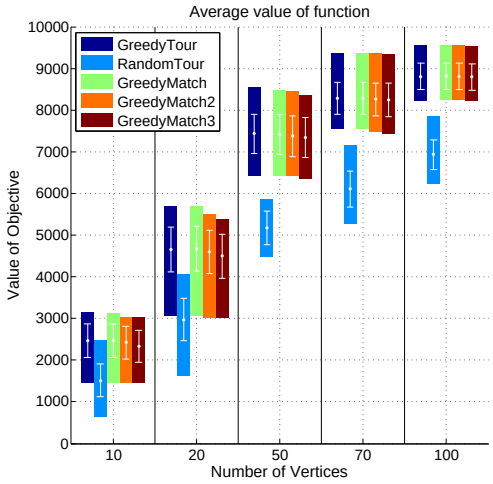
	GT	RT	GM	GM2	GM3
10	33.4	1	54.3	45.7	36.4
20	103.1	1	141.2	122.1	107.3
50	803.4	1	882.1	832.3	807
70	1903.6	1	2012.1	1942	1912.7
100	4818.9	1	4987.2	4886.8	4851.4

Figure 3.4: (Top left) The bars give the range of results. The white markers inside the bars show the mean and standard deviation. (Bottom left) Number of wins for each algorithm, i.e., number of times the algorithm performed better then the others. Wins include ties (unique wins specified in parens). (Top right) Average runtime (seconds) of algorithms. (Bottom right) Average number of function calls for each algorithm.

– GreedyMatching3 (GM3): Use Algorithm 2 to reduce the matching, then arbitrarily connect up the tour.

Since we do not have a complete extension of Serdyukov's to the submodular case, we have omitted it from this comparison

While the performance of GT and GM are similar (note that the number of ties is high especially for smaller graph sizes), GM takes significantly longer to run, due to the oracle calls needed to determine which edge to remove from each subtour. Although GM2 and GM3 had similar runtimes, note that in this particular set up there were only a small number of subtours (compared to the number of vertices), so few calculations were needed to construct the final tour from the reduced subtours in GM2. It is however possible for the number of subtours to be $\Theta(|V|)$ and in those cases GM2 would be considerably slower



	GT	RT	GM	GM2	GM3
10	35 (10)	0 (0)	28 (3)	22 (0)	4 (0)
20	31 (8)	0 (0)	32 (9)	11 (0)	0 (0)
50	33 (13)	0 (0)	25 (5)	12 (2)	1 (0)
70	29 (10)	0 (0)	29 (10)	8 (1)	4 (0)
100	35 (5)	0 (0)	35 (5)	17 (0)	12 (0)

Figure 3.5: (Top) The bars give the range of results. The white markers inside the bars show the mean and standard deviation. (Bottom) Number of wins for each algorithm, i.e., number of times the algorithm performed better then the others. Wins include ties (unique wins specified in parens).

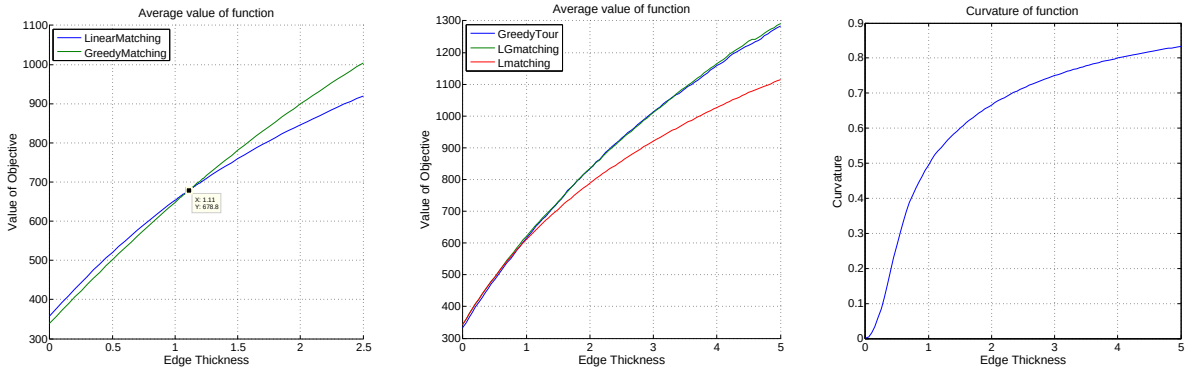
than GM3 as the problem size would not be significantly reduced by first coming up with a 2-matching.

3.5.2 Dependence on Curvature

To illustrate how the results of the 2-matching based algorithm change with curvature, the values of the greedy 2-matching and the linear approximation are now compared. The objective function is modified to be

$$w_{\text{new}}(S) = w(S) + \sum_{e \in S} \text{length}(e).$$

Since $w(S)$ is the total area, its value depends on the thickness of the edges. As the edge thickness is increased, the curvature of the new objective function will (generally) also



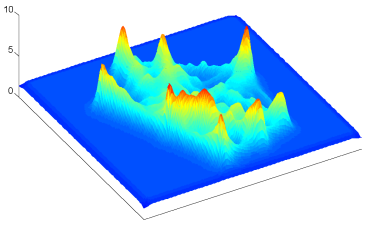
(a) (b) (c)

Figure 3.6: Comparison of value of 2-matching (Left) and final tour (Middle) as a function of edge thickness. (Right) Curvature of objective as a function of edge thickness.

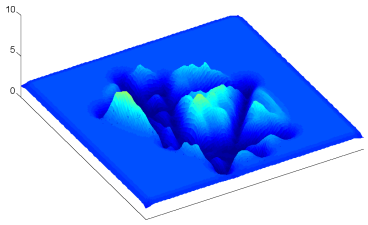
increase.

The experiment is performed on a ten vertex graph. The two 2-matching approximations are compared and the results are shown in Figure 3.6a. For a second test, the GreedyTour and the 2-matching algorithm are compared and the average for 20 different ten vertex graphs is shown in Figure 3.6b. In the figure, the "LGmatching" algorithm creates a greedy 2-matching as well as the linear approximation and takes the best of the two. The best edge from each subtour is removed and the tour is then constructed greedily. The "Lmatching" algorithm runs only the linear approximation to find the 2-matching. Figure 3.6c shows how the curvature changes with edge thickness.

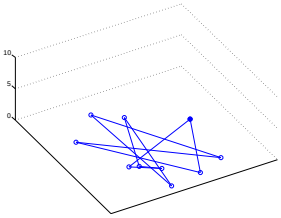
From Figure 3.6a, we can see that for the case where the function is linear (thickness is 0), the linear approximation does better (since it is actually finding the optimal). The greedy 2-matching starts to perform better at a curvature of around 0.53. Looking at the results for the value of the actual tour (Figure 3.6b), we can see that at low values of curvature the linear approximation is being used to create the tour. Eventually, greedily constructing the 2-matching becomes more rewarding and so the linear approximation is disregarded. Generally, over all the values of curvature tested, the 2-matching algorithm performs close to the greedy tour algorithm.



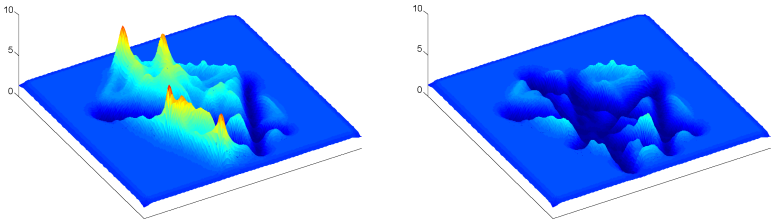
(a) A Priori Variance.



(b) Random Tour A Posteriori Variance (t=10).



(c) Greedy Tour.



(d) Greedy Tour A Posteriori Variance (t=3 and t=10).

Figure 3.7: Monitoring example: The graphs represent uncertainly at each point in the environment. The a priori variance is shown along with the a posteriori variance after the robot has traversed 10 edges of a random tour, as well as 3 and 10 edges of a greedy tour.

3.5.3 Application: Environment monitoring

An example where such a set up is useful is that of monitoring an environment with a mobile robot. Consider the task of monitoring a spatial phenomenon. This phenomenon can be approximated by a Gaussian Process (GP) which, although not a perfect model, can be used to understand the effects of placing sensors at particular locations [19]. Given a robot traveling over the environment, the measurements made by the on-board sensors (at the visited locations) combined with the GP, aid in predicting the environmental process over the entire region of interest.

Assume that the environment is discretized into a finite set of points V. Each $v \in V$ is associated with a random variable X_v that describes the process at that point. For a multi-variate Gaussian distribution, any subset of the random variables also has a multi-variate Gaussian distribution. A subset, A, of the locations is chosen to take measurements. A method of quantifying the uncertainty in a set of measurements (given by the Gaussian

random variable Y_A) is to use the entropy,

$$H(X|Y_A) = \frac{1}{2}\log(2\pi\sigma^2_{X|Y_A}).$$

The problem then becomes to choose the locations that minimize the differential entropy. An alternative measure is mutual information

$$I(X;Y_A) := H(X) - H(X|Y_A),$$

which is submodular and monotone as long as the observations Y_A are conditionally independent given X [36], [17].

For our set up, we can, in addition to discretizing the environment, discretize the edges of the graph. For any edge that is chosen as part of the path of the robot, the points on that edge are taken to be measurement locations.

As an example, a 20×20 environment is created with grid cell size of 0.5×0.5. Any two points p_1 and p_2 have a covariance given by $e^{-k\|p_1 - p_2\|}$. The graph is generated by uniformly randomly placing 10 vertices over the region. The a priori and a posteriori variances for the greedy and random tours are shown in Figure 3.7. In the example shown, only the variance of the region over which the graph is defined is affected. Placing the vertices to have better coverage is a separate problem that is not addressed here.

Chapter 4

Sensor Scheduling for Kalman Filtering

In this chapter we consider the problem of monitoring using a static sensor network. Due to energy and communication constraints, using all the sensors is inefficient. Therefore, we seek to determine the best subset of the sensors that should be used at each time step. The goal is to create a schedule that will optimize the covariance of the error of the state estimate over a finite time horizon. The estimator is taken to be a Kalman filter since that is the optimal choice to obtain the best state estimate for a given schedule. In Section 4.1 we formally define the problem. We discuss some possible objective functions for the optimization in Section 4.2 and define the one that we will use. We then study the submodularity properties of the chosen metric in Section 4.3.

4.1 Problem Statement

Consider the stochastic LTI system

$$\begin{aligned} x_{t+1} &= Ax_t + w_t \\ y_{t+1} &= S_t C x_t + S_t v_t, \end{aligned} \tag{4.1}$$

where

- $t \in \mathbb{Z}_{\geq 0}$, $x_t \in \mathbb{R}^n$ and $y_t \in \mathbb{R}^k$.

- $A \in \mathbb{R}^{n\times n}$ and $C \in \mathbb{R}^{m\times n}$.
- w_t (process noise) and v_t (measurement noise) are zero mean Gaussian noise vectors with covariance matrices $W \in \mathbb{R}^{n\times n} \succeq 0$ and $V \in \mathbb{R}^{m\times m} \succ 0$ respectively. Assume that the noises are independent over time.
- $S_t \in \{0,1\}^{k\times m}$ is a binary sensor selection matrix. Assume that every row of S contains only a single element that is equal to 1.

The covariance update steps for the Kalman Filter can be written as

$$\Sigma_{t|t-1} = \rho^T(\Sigma_{t-1|t-1}) := A\Sigma_{t-1|t-1}A^\top + W \tag{4.2}$$

$$\begin{aligned}\Sigma_{t|t} = \rho^M(\Sigma_{t|t-1}) &:= \Sigma_{t|t-1} - \Sigma_{t|t-1}C^\top S_t^\top (S_t C\Sigma_{t|t-1}C^\top S_t^\top + S_t V S_t^\top)^{-1} S_t C\Sigma_{t|t-1} \\ &= \left(\Sigma_{t|t-1}^{-1} + C^\top S_t^\top (S_t V S_t^\top)^{-1} S_t C\right)^{-1}.\end{aligned} \tag{4.3}$$

It is known that the Kalman filter gives the best mean squared error of the state estimate among all linear estimators. Since we can only choose k sensors at each time step, the problem that we consider here is: from all the possible T-horizon sensor schedules $\sigma = (\sigma_1, \ldots, \sigma_T)$, which one optimizes the error covariance?

The initial a posteriori covariance estimate is Σ_0. The a posteriori update can be written as the following recursion,

$$\begin{aligned}\Omega_t^\sigma(\Sigma_0) &:= \rho_{\sigma_t}^M(\rho^T(\Omega_{t-1}^\sigma(\Sigma_0))) = \left((A\Omega_{t-1}^\sigma(\Sigma_0)A^\top + W)^{-1} + M_t\right)^{-1}, \\ \Omega_0^\sigma(\Sigma_0) &= \Sigma_0, \\ M_t &:= C^\top S_t^\top (S_t V S_t^\top)^{-1} S_t C,\end{aligned} \tag{4.4}$$

where S_t is the binary selection matrix that depends on σ_t and M_t is defined for convenience. We will omit the initial covariance parameter for Ω_t from now on unless there is ambiguity. Note that with this definition, the first time update comes before the first measurement update, so in effect, an initial measurement is skipped. This can be avoided by using a "dummy" initial covariance of $A^{-1}(\Sigma_{0|-1} - W)A^{-\top}$ where $\Sigma_{0|-1}$ is the initial a priori estimate.

Formally, we seek to optimize $F(\Omega_T^\sigma(\Sigma_0))$, for some covariance metric F, under the constraint that at most k sensors can be chosen at each time step. For the rest of this chapter, $F(\Omega_t^\sigma(\Sigma_0))$ and $F(\sigma)$ will be used interchangeably to refer to the value of the covariance after running the sequence σ.

Remark 4.1.1 (Sequence function). When looking at multiple time steps, the exact order of the measurements makes a difference. Therefore, the problem is choosing a sequence of measurements and not just set of possible measurements. So the objective function is a sequence function. •

4.2 Possible performance metrics

For a sequence of sensor selections $\sigma = (\sigma_1, \sigma_2, \ldots)$, the problem is to find the sensor sequence that minimizes the covariance over a given finite horizon T. The covariance matrix it self has a physical interpretation with respect to the actual state estimate. The η-confidence ellipsoid for the estimation error is the minimum volume ellipsoid that contains the estimation error $x - \hat{x}$ with probability η [32]. The lengths of the axis of this ellipsoid are proportional to the eigenvalues of the covariance matrix.

One method to compare different sensor selections is to order the a posteriori covariance matrices in the p.s.d. sense. This means that one of the ellipsoids will be smaller than the other (P5). However, having a scalar measure is more convenient. A detailed comparison of various performance measures can be found in [60]. Some of these are summarized below:

- Two possible measures are the **volume** of the confidence ellipsoid or the **mean radius**, defined as the geometric mean of the lengths of the semi-axes. Both are directly proportional the determinant of the covariance matrix.
- Using the **mean squared error** can be represented by the sum of the eigenvalues of the covariance matrix which is equivalent to its trace.
- The **worst case error covariance** is proportional to the maximum eigenvalue of the covariance. A obvious drawback of this metric is that it does not take into account any of the other eigenvalues. So two measurements that do not have an effect on the maximum length axis will be treated as equivalent even though one of them might significantly reduce the overall volume.

Remark 4.2.1 (Entropy). The **entropy** of the covariance matrix of a Gaussian random variable is given by $\frac{1}{2}\log\det(\Sigma) + \frac{N}{2}\log(2\pi e)$ [40] which visibly relates to the ellipsoid volume. Another similar measure is the **mutual information** which is just the change in entropy. The advantage of this is that it takes into account the initial covariance. •

For the purposes of this chapter, we will use the following as the objective function for the optimization,

$$F(\sigma) = F(\Omega_T^\sigma(\Sigma_0)) = \log\det(\Omega_0^\sigma) - \log\det(\Omega_T^\sigma) = \log\left(\det(\Sigma_0)\det(\Omega_T^\sigma)^{-1}\right). \tag{4.5}$$

A similar objective function was used previously [32, 48, 40]; here we attempt to generalize some of these results.

4.3 Submodularity of the Kalman Update

In this section we are interested in characterizing the conditions under which sensor selection can be posed as a submodular optimization problem. This will allow us to apply known results in submodular set function optimization to obtain bounds for various algorithms such as the greedy algorithm. Previously, in [48], the authors show that for a single time step submodularity does hold. They do not, however, make any comments about whether or not the concept of submodularity can be applied over multiple time steps. In [27], the authors show that the function $H(\sigma) = \text{trace}(\Sigma_{t|t-1}^{\emptyset} - \Sigma_{t|t-1})$, where $\Sigma_{t|t-1}^{\emptyset}$ is the final covariance if no sensors are selected at each time step, is submodular for a single time step as well as over multiple time steps (for a single sensor per time step). We found that this claim is false as evident in the following example.

Example 4.3.1 (Counterexample)**.** Consider the system

$$A = \begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix}, \quad C = \begin{bmatrix} 1.0 & 0 \\ 0.5 & 0.5 \\ 0.7 & 0.3 \\ 0.3 & 0.7 \end{bmatrix}, \quad W = \Sigma_0 = I_{2\times 2}, \quad V = I_{4\times 4}. \tag{4.6}$$

Take $X_1 = \{2, 3\}$ and $X_2 = \{2, 3, 4\} \supset X_1$. So $H((X_1 \cup \{1\})) - H((X_1)) = 0.0684$ but $H((X_2 \cup \{1\})) - H((X_2)) = 0.0692$. As we can see, the gain of adding sensor 1 to sensors X_2 is greater than that by adding to its subset X_1, which contradicts the decreasing marginal benefits property of submodularity. ▲

We first consider the case of picking the optimal set of sensors at a particular time step. With certain assumptions, the problem can be stated as a submodular set function optimization. The second step is to determine the sequence of measurements over a time horizon that will optimize the covariance. We will see that the objective function is no longer submodular; however, we can derive a set of conditions under which submodularity can be obtained.

4.3.1 Single Time Step

At a particular time step, k measurements have to be selected out of a set of m possible ones. This means there are $\binom{m}{k}$ possibilities to consider (which is $O(m^k)$). In order to reduce complexity, it makes sense to approximate the optimal choice.

Remark 4.3.2 (Set function). At a particular time step, a set of sensors is chosen to make a measurement. Since all of the measurements will be made together, the exact ordering in unimportant. Therefore, the problem is one in combinatorial optimization where the objective is a set function. •

Studying the objective function (4.5), we can see that, for a given time t and sequence σ that is defined until $t-1$, maximizing $F(\Omega_t^\sigma(\Sigma_0))$ is equivalent to maximizing $F(\Omega_1^{\sigma_t}(\Omega_{t-1}^\sigma))$, where σ_t is the element at index t in σ. Note that since $\Omega_{t-1}^\sigma(\Sigma_0)$ is known, so is $\rho^T(\Omega_{t-1}^\sigma(\Sigma_0))$, i.e., the a priori covariance at time step t. Therefore, we can define a new objective function,

$$G_t(X) := \log\det(\rho^T(\Omega_{t-1}^\sigma))\det(\rho^T(\Omega_{t-1}^\sigma)^{-1} + M_X), \tag{4.7}$$

where X is the set of sensors to be chosen at time t and M_X (through an abuse of notation) is the measurement matrix (similar to the one defined in (4.4)) if the rows of C corresponding to X are chosen. The optimization problem is

$$\max_{\{X\subseteq E|\ |X|=k\}} G_t(X). \tag{4.8}$$

If $G_t(X)$ is a monotone non-decreasing function, then the constraint can be changed into $\{X \subseteq E|\ |X| \leq k\}$ which is a uniform matroid constraint. If the function is additionally submodular, then we can obtain bounds for a greedy approach to selecting the set of measurements at time t.

Assume that V is a diagonal matrix (i.e., the measurement noises are independent over sensors). This assumption is necessary as otherwise (4.7) is not generally submodular as demonstrated in the following example.

Example 4.3.3 (Effect of correlated measurement noise)**.** Consider the simple system

$$A = C = \Sigma_0 = I_2, \qquad W = 0, \qquad V = \begin{bmatrix} 1 & 0.75 \\ 0.75 & 1 \end{bmatrix} \tag{4.9}$$

In this case, $G_1(\{2,1\}) - G_1(\{2\}) = 1.3683 > G_1(\{1\}) = 0.6931$, i.e., adding measurement 1 to measurement 2 gives a larger benefit then adding it to the empty set. This violates the decreasing marginal benefits property of submodularity. ▲

Lemma 4.3.4. *For a single time step, the function $G_t(X)$ is monotone non-decreasing and submodular with an initial value of 0.*

Proof. The function can be written as

$$G_t(X) = \log\det(\rho^T(\Omega^\sigma_{t-1})) + \log\det(\rho^T(\Omega^\sigma_{t-1})^{-1} + M_X)$$

Note that $\log\det(\rho^T(\Omega^\sigma_{t-1}))$ can be treated as a constant at time step t. In [48] it is shown that $\log\det(\rho^T(\Omega^\sigma_{t-1})^{-1} + M_X)$ is monotone non-decreasing and submodular in the selected measurements. Adding a constant does not change this property.

Also, if $X = \emptyset$, then $M_X = 0$ and therefore $G_t(\emptyset) = 0$. □

By this lemma, picking the best set of sensors at a particular time step can be phrased as the optimization of a normalized non-decreasing submodular set function over a uniform matroid constraint. Using a greedy algorithm, this can be solved to within $(1 - \frac{1}{e})$ of the optimal [44].

Remark 4.3.5. In [48] it is shown that maximizing the log det of the a posteriori covariance inverse gives a $(1-\frac{1}{e})$-approximation using a greedy algorithm. This statement is incorrect as stated since the objective function is required to be normalized in order to get the constant factor bound. If it is not, then there will be an error term added that depends on the initial value, i.e., GREEDY $\geq (1 - \frac{1}{e})\text{OPT} + G(\emptyset)\epsilon$, where $\epsilon > 0$ depends on k. In this case, if the initial value is negative then the bound will get worse. •

4.3.2 Multiple Time Steps

In the previous section we saw that a greedy algorithm can be used to obtain a $(1 - 1/e)$-approximation for a particular time step since the problem can be phrased as a submodular set function maximization. We now assume that at each time step the optimal sensor can be chosen in an attempt to study the properties of the objective function over multiple time steps. We assume that A is non-singular and that there is no process noise, i.e., $w_t = 0$.

Consider the following example.

Example 4.3.6 (Effect of process noise)**.** Consider the following system.

$$A = \begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix} \quad C = \Sigma_0 = I \quad W = V = \begin{bmatrix} 0.25 & 0 \\ 0 & 0.5 \end{bmatrix} \tag{4.10}$$

Take $\sigma_1 = (1, 2, 2)$, $\sigma_2 = (1, 2, 2, 2)$ and $s = \{2\}$. We can see that the objective function (4.5) is not submodular since $F(\sigma_1 \parallel s) - F(\sigma_1) = -0.0124$ which is less than $F(\sigma_2 \parallel s) - F(\sigma_2) = -0.0043$. This violates the decreasing marginal benefits of submodularity.

If $W = 0$, then there is no longer a contradiction. In fact, as we will see, this system (with $W = 0$) actually results in the objective function being sequence submodular. ▲

Using the assumptions above (that A is full rank and $W = 0$), we can solve for the solution to the recursive covariance update definition.

$$\begin{aligned}
\Omega_0^\sigma &= \Sigma_0 \\
\Omega_t^\sigma &= \left((A\Omega_{t-1}^\sigma A^\top)^{-1} + M_t\right)^{-1} = \left(A^{-\top}(\Omega_{t-1}^\sigma)^{-1}A^{-1} + M_t\right)^{-1} \\
\therefore \Omega_t^\sigma &= \left((A^{-\top})^t\Sigma_0^{-1}A^{-t} + \sum_{i=1}^{t}(A^{-\top})^{(t-i)}M_iA^{-(t-i)}\right)^{-1}. \qquad (4.11)
\end{aligned}$$

Lemma 4.3.7. *Given that $XPX^\top \preceq P$ for some p.s.d. P, then $X^lP(X^\top)^l \preceq X^jP(X^\top)^j$ holds for $j < l$ where $j, l \in \mathbb{Z}_+$. As a result $X^kP(X^\top)^k \preceq P$ holds for all $k \in \mathbb{Z}_+$.*

Proof. We can show that $X^iP(X^\top)^i \preceq X^{i-1}P(X^\top)^{i-1}$. The statement holds for $i = 1$. Assume it holds for $i = k$. By (P7), $X^kP(X^\top)^k \preceq X^{k-1}P(X^\top)^{k-1} \implies XX^kP(X^\top)^k(X^\top) \preceq XX^{k-1}P(X^\top)^{k-1}X^\top$. Therefore the claim holds for $i = k + 1$. Applying this recursively creates a sequence of orderings and the statement to be proven follows. □

Theorem 4.3.8. *For the function (4.5), the value of the empty set is 0. Also, with the assumptions that A is full rank and $W = 0$, if $A\Sigma_0A^\top \preceq \Sigma_0$ and $AM_iA^\top \preceq M_i$ for all possible measurement matrices, then the function is monotone non-decreasing and submodular.*

Proof. The value of empty set is easy to see,

$$F(\Omega_0^\emptyset(\Sigma_0)) = \log\det(\Sigma_0)\det(\Sigma_0^{-1}) = 0.$$

Take $A \subset B$ where B is a sequence of measurements. Let $B = (1, \ldots, b)$, with the corresponding measurements $\{M_i\}_{i\in B}$, and $\{A(i)\}_{i=1}^a$ are the indices in B that are part of A. Note that by definition of subsequence, $A(i) < A(i+1)$ for all i, i.e., the order in which elements appear in B must be the same as the order in which they appear in A.

Monotonicity: For monotonicity, the requirement is $F(A) \leq F(B)$.

$$\begin{aligned}
&\log\det(\Sigma_0)\det(\Omega_a^A)^{-1} \leq \log\det(\Sigma_0)\det(\Omega_b^B)^{-1} \\
&\iff \det(\Omega_a^A) \geq \det(\Omega_b^B)
\end{aligned}$$

Submodularity: For submodularity, the requirement is $F(A \parallel x) - F(A) \geq F(B \parallel x) - F(B)$. This can be written as,

$$\begin{aligned} &\log\det(\Omega_{a+1}^{Ax})^{-1} - \log\det(\Omega_a^A)^{-1} \geq \log\det(\Omega_{b+1}^{Bx})^{-1} - \log\det(\Omega_b^B)^{-1} \\ \iff &\log\det(\Omega_{a+1}^{Ax})^{-1}\det(\Omega_a^A) \geq \log\det(\Omega_{b+1}^{Bx})^{-1}\det(\Omega_b^B). \end{aligned}$$

Now, applying the covariance update formula (4.4),

$$\begin{aligned} \Omega_{a+1}^{Ax}(\Sigma_0) = \Omega_1^x(\Omega_a^A) &= \left((A\Omega_a^A A^\top)^{-1} + M_x\right)^{-1}, \\ \Omega_{b+1}^{Bx}(\Sigma_0) = \Omega_1^x(\Omega_b^B) &= \left((A\Omega_b^B A^\top)^{-1} + M_x\right)^{-1}. \end{aligned}$$

Substituting back, the condition for submodularity becomes

$$\iff \det(\Omega_a^A)\det((A\Omega_a^A A^\top)^{-1} + M_x) \geq \det(\Omega_b^B)\det((A\Omega_b^B A^\top)^{-1} + M_x).$$

Multiplying both sides by $\det(A)\det(A^\top)$,

$$\begin{aligned} \iff \det(A\Omega_a^A A^\top)\det((A\Omega_a^A A^\top)^{-1} + M_x) &\geq \det(A\Omega_b^B A^\top)\det((A\Omega_b^B A^\top)^{-1} + M_x). \\ \iff \det(I + (A\Omega_a^A A^\top)M_x) &\geq \det(I + (A\Omega_b^B A^\top)M_x) \end{aligned}$$

Taking $M_x := L^\top L$ (since $M_x \succeq 0$) and applying the matrix determinant lemma (P3), the condition becomes

$$\det(I + (LA)\Omega_a^A(LA)^\top) \geq \det(I + (LA)\Omega_b^B(LA)^\top).$$

A sufficient condition for both monotonicity and submodularity to hold then is $\Omega_a^A \succeq \Omega_b^B$. Applying (4.11), this is equivalent to

$$\begin{aligned} &(A^{-\top})^a\Sigma_0^{-1}A^{-a} + \sum_{i=1}^{a}(A^{-\top})^{(a-i)}M_{A(i)}A^{-(a-i)} \\ &\preceq (A^{-\top})^b\Sigma_0^{-1}A^{-b} + \sum_{j=1}^{b}(A^{-\top})^{(b-j)}M_jA^{-(b-j)}. \end{aligned} \tag{4.12}$$

As a consequence of Lemma 4.3.7, since $\Sigma_0 \succ 0$ and $a \leq b$,

$$(A^{-\top})^a\Sigma_0^{-1}A^{-a} \preceq (A^{-\top})^b\Sigma_0^{-1}A^{-b}.$$

For some $A(i)$, $M_{A(i)}$ will appear on both sides of the inequality. We can show that $(a-i) \leq (b-j)$. Since $A \subset B$, the first element of A ($i = 1$) can be at most in position

$b-a+1$ in B. Similarly, the second element in A can be at most in position $b-a+2$ in B. Therefore, the inequality $j \leq b-a+i$ holds. Combining this with the fact that $M_{A(i)} \succeq 0$, applying Lemma 4.3.7 again, we have

$$(A^{-\top})^{a-i} M_{A(i)} A^{-(a-i)} \preceq (A^{-\top})^{b-j} M_{A(i)} A^{-(b-j)}.$$

Therefore, under the assumptions made, the inequality (4.12) holds and the function is monotone non-decreasing and submodular. □

So, the objective function is a normalized non-decreasing submodular sequence function if the conditions of Theorem 4.3.8 are met. As a result, by Lemma 2.5.4, greedily selecting the measurement matrix at each time step will give a $(1 - \frac{1}{e})$-approximation.

Remark 4.3.9. For an easy check, the condition can be restricted to requiring $Ac_j c_j^\top A^\top \preceq c_j c_j^\top$ instead of the more general $AM_iA^\top \preceq M_i$ (i.e. for each sensor rather than each possible set of sensors). This only applies if V is diagonal. •

4.3.3 Greedy Approximation

Assuming that the conditions of Theorem 4.3.8 are met, sequentially selecting the best measurement matrix at each time step will give a $(1 - \frac{1}{e})$-approximation (≈ 0.6321). One drawback is that there are $\binom{m}{k} = O(m^k)$ possible measurement matrices at each step. Combining this approach with the previous result of greedily constructing the measurement matrix at each time step can lead to a faster runtime but a less tight bound, as shown in the following theorem.

Theorem 4.3.10. *Consider the sensor scheduling problem such that $W = 0$, V is diagonal, A is full rank, $A\Sigma_0 A^\top \preceq \Sigma_0$ and $AM_iA^\top \preceq M_i$ for all possible measurement matrices M_i. Using (4.5) as the objective function leads to a $1 - \frac{1}{e^{1-1/e}}$-factor approximation in $O(Tkmn^2 + Tn^3)$ time if the k sensors are chosen greedily at each time step.*

Proof. Sequentially selecting the best measurement matrix at each time step corresponds to solving the optimization problem

$$\begin{aligned}
&\max_{\sigma_t} \Delta(\sigma_t | \sigma_{[1,t-1]}) \\
\equiv &\max_{\sigma_t} \log\det(\Sigma_0)\det(\Omega_t^\sigma)^{-1} - \log\det(\Sigma_0)\det(\Omega_{t-1}^\sigma)^{-1} \\
\equiv &\max_{\sigma_t} \log\det(\Omega_{t-1}^\sigma)\det(\Omega_t^\sigma)^{-1},
\end{aligned}$$

at each time step, where $\sigma_{[1,t-1]}$ is the sensor sequence from 1 to $t-1$.

By Lemma 4.3.4, since the measurement noise matrix V is diagonal, building the measurement matrix greedily at a particular time step can be solved to within $(1-\frac{1}{e})$ of the optimal using (4.7) as the objective function. Let Σ_t^g and Σ_t^o be the resulting covariance after applying the sequence of measurements $\sigma_{[1,t-1]}$ and then selecting the greedy and optimal measurements respectively at time step t. Therefore, taking $\alpha = (1-\frac{1}{e})$,

$$\begin{aligned}
&\log\det(\rho^T(\Omega_{t-1}^\sigma))\det(\Sigma_t^g)^{-1} \geq \alpha\log\det(\rho^T(\Omega_{t-1}^\sigma))\det(\Sigma_t^o)^{-1} \\
\implies &\log\det(\Omega_{t-1}^\sigma)\det(\Sigma_t^g)^{-1} \geq \alpha\log\det(\Omega_{t-1}^\sigma)\det(\Sigma_t^o)^{-1} + (\alpha-1)\log\det(A)^2,
\end{aligned}$$

since $W=0$ so $\rho^T(\Omega_{t-1}^\sigma) = A\Omega_{t-1}^\sigma A^\top$.

Let $\sigma^g = (\sigma_1^g,\ldots,\sigma_T^g)$ be the sequence of measurements by greedily selecting the k sensors at each time step and $\sigma^o = (\sigma_1^o,\ldots,\sigma_T^o)$ be the optimal schedule. Therefore,

$$\Delta(\sigma_t^g|\sigma_{[1,t-1]}^g) \geq \alpha\max_{\sigma_i}\Delta(\sigma_i|\sigma_{[1,t-1]}^g) + \epsilon,$$

where $\epsilon = (\alpha-1)\log\det(A)^2$. Note that if $\epsilon=0$, then we can apply Lemma 2.5.4 to deduce that $F(\sigma^g) \geq (1-\frac{1}{e^\alpha})F(\sigma^o)$. However, since $\epsilon\neq 0$ we cannot apply this directly. Instead, we can solve for a bound of the greedy schedule by imitating the proof of Lemma 2.5.4 (given in [1]).

$$\begin{aligned}
\Delta(\sigma_t^g|\sigma_{[1,t-1]}^g) &\geq \alpha\max_{\sigma_i}\Delta(\sigma_i|\sigma_{[1,t-1]}^g) + \epsilon \geq \alpha\max_{\sigma_i\in\sigma^o}\Delta(\sigma_i|\sigma_{[1,t-1]}^g) + \epsilon \\
&\geq \frac{\alpha}{T}\Delta(\sigma^o|\sigma_{[1,t-1]}^g) + \epsilon \\
&\geq \frac{\alpha}{T}\left(F(\sigma_{[1,t-1]}^g \parallel \sigma^o) - F(\sigma_{[1,t-1]}^g)\right) + \epsilon \\
&\geq \frac{\alpha}{T}\left(F(\sigma^o) - F(\sigma_{[1,t-1]}^g)\right) + \epsilon \\
\implies F(\sigma_{[1,t]}^g) &\geq \frac{\alpha}{T}F(\sigma^o) + (1-\frac{\alpha}{T})F(\sigma_{[1,t-1]}^g) + \epsilon.
\end{aligned}$$

Solving this recurrence relation,

$$\begin{aligned}
F(\sigma^g) = F(\sigma_{[1,T]}^g) &\geq \left(\frac{\alpha}{T}F(\sigma^o)+\epsilon\right)\sum_{i=0}^{T-1}(1-\frac{\alpha}{T})^i \\
&= F(\sigma^o)\left(1-(1-\frac{\alpha}{T})^T\right) + \frac{T}{\alpha}\left(1-(1-\frac{\alpha}{T})^T\right)\epsilon \\
&\geq F(\sigma^o)(1-e^{-\alpha}) + \frac{T}{\alpha}(1-e^{-\alpha})\epsilon
\end{aligned}$$

Therefore, the greedy schedule is within a factor of $(1 - e^{-\alpha})$ of the optimal but there is an error term of $T(1-\alpha^{-1})(1-e^{-\alpha}) \log\det(A)^2 \approx (-0.2727)T \log\det(A)^2$. Note, however, that since $\Sigma_0 \succ 0$,

$$A\Sigma_0 A^\top \preceq \Sigma_0 \implies \det(A\Sigma_0 A^\top) \leq \det(\Sigma_0) \implies \det(A)^2 \leq 1 \implies \log\det(A)^2 \leq 0.$$

As a result, $F(\sigma^g) \geq (1 - e^{-\alpha})F(\sigma^o)$ and substituting the value of α gives the constant factor bound of ≈ 0.4685.

As for the complexity, the pure greedy approach means that there will be kmT iterations; for each of the T time steps, k measurements need to be selected from a set of m. The optimization for each time step can be performed intelligently to avoid having to repeatedly calculate inverses and determinants. This results in a runtime of $O(n^2mk)$ per time step [48] such that the output is the a posteriori covariance matrix. In addition to this, the time update requires two matrix multiplications which naively require $O(n^3)$ time. Therefore, the total runtime is $O(Tkmn^2 + Tn^3)$. □

Note that for a finite time horizon, T, the constraint of selecting k sensors out of m possibilities per time step can be phrased as a partition matroid. The number of "partitions" here would be T. An alternate method of approximating the solution would be to consider all time steps at once. So instead of sequentially constructing the schedule, we start off with an empty schedule of length T. A measurement is added greedily to some time until all the kT slots have been filled. In this case, the optimization is over a partition matroid. However, we would need to determine if the objective function is non-decreasing and submodular. If it is, then the greedy algorithm would be within $\frac{1}{2}$ of the optimal by Lemma 2.5.1.

Chapter 5

Infinite Horizon Sensor Scheduling

In the previous chapter we determined that the finite horizon sensor selection problem can be framed as a submodular optimization under some restrictive assumptions. Motivated by the requirement of persistent monitoring, we now consider the infinite horizon case. The notation used in the previous chapter applies here. However, the objective function (4.5) is defined for a finite time horizon T and the limit as $T \to \infty$ may not exist since the error covariance may never settle to a single value. If the error covariance is bounded then some possible metrics are:

- the limiting upper bound to the volume of the covariance ellipsoid $F(\sigma) = \limsup_{T\to\infty} \log\det(\Omega_T^\sigma)$,
- the limiting upper bound to the mean squared error $F(\sigma) = \limsup_{T\to\infty} \operatorname{trace}(\Omega_T^\sigma)$,
- the average mean squared error $F(\sigma) = \lim_{T\to\infty} \frac{1}{T}\sum_{t=0}^{T-1} \operatorname{trace}(\Omega_t^\sigma)$.

Regardless of what the actual metric is, it is important that the error covariance be bounded since otherwise the state estimate will never be accurate.

The system (4.1) can be viewed as a LTV system where the only time-varying quantity is the measurement matrix. As is deducible from the structure of (4.1), the measurement matrix C_t will just be a submatrix of C.

We know that the error covariance is bounded if the system is uniformly detectable (Lemma 2.9.1). Now, the question to ask is, given an LTI system, does there even exist a sequence of measurements that is uniformly detectable? In this section we restrict the

problem to the case where $k = 1$, i.e., only a single measurement is taken at each time step.

Considering the system that we are looking at is only partially time varying, we can redefine some of the definitions given in Section 2.8. Given an LTI system (A, C) and a sequence of measurements $\sigma = (\sigma_0, \sigma_1, \ldots)$ where σ_i is a row of C. For a given time t and time window k, the sequence observability matrix is

$$B_\sigma(t, t+k) = \begin{bmatrix} \sigma_t \\ \sigma_{t+1}A \\ \vdots \\ \sigma_{t+k}A^k \end{bmatrix}.$$

The sequence of measurements σ is uniformly detectable if there exists non-negative integers s, r and constants $\alpha \in [0, 1)$ and $\beta > 0$, such that for all $\{x \in \mathbb{R}^n | \, \|x\| = 1\}$ and all times t,

$$||A^r x|| \geq \alpha \implies \|B(t, t+s)x\| \geq \beta > 0. \tag{5.1}$$

Additionally, the sequence is uniformly observable if there exists integer s and positive constants β_1, β_2 such that

$$0 < \beta_1 \leq \|B(t, t+s)x\| \leq \beta_2 \iff \operatorname{rank}(B(t, t+s)) = n. \tag{5.2}$$

Naturally, a uniformly observable sequence is also uniformly detectable. We now look at when a schedule can be uniformly detectable and hence have a bounded error covariance estimate.

Remark 5.0.11 (Comment on observability condition). In the general time varying case the "if and only if" in (5.2) holds in the forward direction (assume that $\operatorname{rank}(B) < n$, then there exists x such that $Bx = 0$). It may not, however, necessarily hold in the reverse; since the condition has to hold over all t, an upper bound may not exist if C_t or A_t is unbounded. In our case though this is not a problem since A_t is time invariant and C_t can only take one of m possible values (and is hence bounded). •

5.1 Existence of Uniformly Detectable Sequence

It is reasonable to expect that if (A, C) is observable, a sequence of measurements could exist such that the system is uniformly observable through that sequence. This, however, is not the case.

Example 5.1.1 (Non-existence of observable sequence)**.** A uniformly observable sequence may not exist if $\text{rank}(A) < n$. As an example consider the system

$$A = \mathbf{1}_{3\times 3} \qquad C = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix},$$

so (A, C) is full rank. However, every vector $c_i A^k, k > 0$ and c_i is the ith row of C, consists of just the same three values — the sum of the elements in c_i (i.e., $2^{k-1}(c_i\mathbf{1})\begin{bmatrix}1 & 1 & 1\end{bmatrix}$). Therefore, any sequence will result in $\text{rank}(B(t, t+k))$ either 1 or 2 for all t and k, where $B(t, t+k)$ is the sequence observability matrix.

A more trivial example is if $A = \text{zeros}(3)$. Here, it is obvious that the second row onwards is 0 so $\text{rank}(B(t, t+s)) = 1$.

Note that, in both these cases, although the initial state cannot be predicted, the actual progression after a certain time period can, in fact, be determined. ▲

In this section we will show that if (A, C) is detectable, then there will exist a sequence of measurements that is uniformly detectable.

Proposition 5.1.2. *Given that (A, C) is observable and that A is full rank, then there exists a sequence of measurements $\sigma = (d_0, \ldots, d_s)$, where d_i is a row of C, such that the system is uniformly observable through the periodic sequence σ.*

Proof. By construction, take the periodic sequence where each measurement is repeated n times, i.e., let $\sigma = (1, \ldots, 1, 2, \ldots, 2, \ldots, m)$. Let c_i be the i^{th} row of C. Define,

$$\Theta = \begin{bmatrix} C \\ CA \\ CA^2 \\ \vdots \\ CA^{n-1} \end{bmatrix} = \begin{bmatrix} c_1 \\ \vdots \\ c_m \\ c_1A \\ \vdots \\ c_mA \\ \vdots \\ c_1A^{n-1} \\ \vdots \\ c_mA^{n-1} \end{bmatrix}, \qquad B = \begin{bmatrix} c_1 \\ \vdots \\ c_1A^{n-1} \\ c_2A^n \\ \vdots \\ c_2A^{2n-1} \\ \vdots \\ c_mA^{mn-1} \end{bmatrix}. \tag{5.3}$$

In order for the sequence to be observable, we require the existence of (s, β) such that $\text{rank}(B(t, t+s)) = n$ for all t. Take $s = 2mn$ so that $B(t, t+s)$ always contains the full sequence σ. As a result, the sequence observability matrix for s time steps will always contain the rows of BA^p for some p. So, showing $\text{rank}(BA^p) = n$ is sufficient for observability. Note that since A is full rank, $\text{rank}(BA^p) = n \iff \text{rank}(B) = n$. Since it is given that $\text{rank}(\Theta) = n$, it suffices to show that each of the rows of Θ can be written as a linear combination of the rows in B.

First, define sets to represent the rows of Θ and B. Let $X_i = \{c_i, c_i A, \ldots, c_i A^{n-1}\}$ for $i = 1, \ldots, m$. Note that the rows of Θ comprise of the vectors in the multiset $\bigcup_{i=1}^{m} X_i$. Also, note that $x \in X_i \implies xA^{(i-1)n}$ is a row of B. Let $X_i^b = X_i A^{(i-1)n} = \{c_i A^{(i-1)n}, c_i A^{(i-1)n+1}, \ldots, c_i A^{in-1}\}$. So the rows of B comprise of elements of the multiset $\bigcup_{i=1}^{m} X_i^b$.

For any particular $1 \leq i \leq m$, there are $k_i \leq n$ linearly independent vectors in X_i. Since A is full rank, the set X_i^b contains k LI vectors too (by (P2)). Also, due to the Cayley-Hamilton theorem (P1), $x \in X_i^b \implies x \in \text{span}(X_i)$. Since there are k LI vectors in the span of X^i, these vectors themselves span the space. As a result, every vector in X_i is in $\text{span}(X_i^b)$.

Therefore, if A is full rank, then every row of Θ is in the span of the rows of B, which means that $\text{rank}(B) \geq \text{rank}(\Theta) = n$. Since B is an $mn \times n$ matrix, $\text{rank}(B) \leq n$. So $\text{rank}(B) = n$. □

Using this result, we now give a constructive proof that shows the existence of a uniformly detectable sequence.

Theorem 5.1.3. *Given that (A, C) is detectable, then the sequence of measurements $\sigma = (1, \ldots, 1, 2, \ldots, 2, \ldots, m, \ldots, m)$ is uniformly detectable.*

Proof. Let T be defined as in (2.5). Transforming the system into observable standard form gives the STM $\bar{A}$ with separate observable and unobservable components. Let z and l be the number of zero and stable eigenvalues respectively of A_o which is a $p \times p$ matrix. Assume that the generalized eigenvectors of the observable component, A_o, are ordered such that $\{v_1, \ldots v_z\}$ correspond to the zero eigenvalue, $\{v_1, \ldots v_l\}$ correspond to the stable eigenvalues, and $\{v_{l+1}, \ldots v_p\}$ correspond to the unstable eigenvalues. Let

$V := \begin{bmatrix} v_1 & \dots & v_p \end{bmatrix}$. Consider the following transform.

$$u = Q^{-1}x = \begin{bmatrix} u_0 \\ u_1 \\ u_2 \\ u_3 \end{bmatrix} \quad \text{where } Q := TP, \quad P = \begin{bmatrix} I & 0 \\ 0 & V_{p\times p} \end{bmatrix}_{n\times n}$$
$$\tilde{A} = Q^{-1}AQ = \begin{bmatrix} A_{\bar{o}} & & A_{12}V \\ & \begin{bmatrix} A_{on} & 0 & 0 \\ 0 & A_{os} & 0 \\ 0 & 0 & A_{ou} \end{bmatrix} & \\ 0 & & \end{bmatrix} \qquad \tilde{C} = CQ = \begin{bmatrix} 0 & C_1 & C_2 & C_3 \end{bmatrix} \tag{5.4}$$

This system has the following properties:

- $\tilde{\Theta} = \begin{bmatrix} 0 & \Theta_1 & \Theta_2 & \Theta_3 \end{bmatrix}$ and $\tilde{B}(t, t+s) = \begin{bmatrix} 0 & B_1 & B_2 & B_3 \end{bmatrix}$. Note that $\begin{bmatrix} \Theta_1 & \Theta_2 & \Theta_3 \end{bmatrix}$ is full rank since this part corresponds to the observable subsystem.
- A_{os} and A_{ou} have stable and unstable eigenvalues respectively, are both full rank and are composed of Jordan blocks.
- A_{on} has all zero eigenvalues and is nilpotent (it is not necessarily the 0 matrix since it is composed of Jordan blocks). As a result, $A_{on}^z = 0$.
- By definition of detectability, $A_{\bar{o}}$ is stable.

For detectability of the sequence to hold, there should exist (s, r, α, β) such that (5.1) is satisfied. Without loss of generality, consider only initial states of unit norm ($\|u\| = 1$).

Case 1: $u_3 = 0$ or A_{ou} does not exist (i.e., A is stable): In this case, only the stable modes are active and so the state approaches 0 exponentially. As a result, for any $\alpha \in (0, 1)$, there exists $r > 0$ such that $\left\| \tilde{A}^r u \right\| < \alpha$ for all u. A more rigorous relationship between r and α can be constructed using ideas similar to the technique in the proof of Lemma 2.8.6.

Case 2: $\|u_3\| > 0$: In this case, (s, β) can be chosen so that $\left\| \tilde{B}(t, t+s)u \right\| \geq \beta$ irrespective of what the chosen values of (r, α) are.

Take $s = 2mn$ so that $\tilde{B}(t, t+s)$ always contains the full sequence σ. As a result, the *sequence observability matrix* for s time steps, $\tilde{B}(t, t+s)$, will always contain the rows of $\tilde{B}_\sigma \tilde{A}^k$ for some $k > 0$, where $\tilde{B}_\sigma$, defined similarly to (5.3), can be represented as

$$\tilde{B}_\sigma = \begin{bmatrix} \tilde{c}^1 \\ \vdots \\ \tilde{c}^1\tilde{A}^{n-1} \\ \vdots \\ \tilde{c}^m\tilde{A}^{mn-1} \end{bmatrix} = \begin{bmatrix} 0 & c_1^1 & c_2^1 & c_3^1 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & c_1^1A_{on}^{z-1} & \vdots & \vdots \\ 0 & 0 & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & c_2^1A_{os}^{n-1} & c_3^1A_{ou}^{n-1} \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & c_2^mA_{os}^{mn-1} & c_3^mA_{ou}^{mn-1} \end{bmatrix} =: \begin{bmatrix} 0 & B_1 & B_2 & B_3 \end{bmatrix},$$

where $\tilde{c}^i$ is the i^{th} row of $\tilde{C}$. Note that

$$\tilde{B}_\sigma\tilde{A}^k = \begin{bmatrix} 0 & B_1 & B_2 & B_3 \end{bmatrix} \begin{bmatrix} A_{\bar{o}}^k & & \cdots & \\ & \begin{bmatrix} A_{on}^k & 0 & 0 \\ 0 & A_{os}^k & 0 \\ 0 & 0 & A_{ou}^k \end{bmatrix} & & \end{bmatrix} = \begin{bmatrix} 0 & B_1A_{on}^k & B_2A_{os}^k & B_3A_{ou}^k \end{bmatrix}.$$

Note that $\begin{bmatrix} \Theta_2 & \Theta_3 \end{bmatrix}$ is full rank and, since both A_{os} and A_{ou} are full rank, $\begin{bmatrix} B_2 & B_3 \end{bmatrix}$ is also full rank (using the same argument as in the proof of (5.1.2)) and so is $\begin{bmatrix} B_2A_{os}^k & B_3A_{ou}^k \end{bmatrix}$. Note that $B_1A_{on}^k = 0$ for $k \geq z$.

Now, without loss of generality, assume that the sequence σ starts at time t. So, since $k = mn \geq z$,

$$\begin{aligned} \tilde{B}(t, t+s)u = \begin{bmatrix} \tilde{B}_\sigma \\ \tilde{B}_\sigma\tilde{A}^{mn} \end{bmatrix} u &= \begin{bmatrix} 0 & B_1 & B_2 & B_3 \\ 0 & 0 & B_2A_{os}^{mn} & B_3A_{ou}^{mn} \end{bmatrix} \begin{bmatrix} u_0 \\ u_1 \\ u_2 \\ u_3 \end{bmatrix} \\ &= \begin{bmatrix} B_1u_1 + B_2u_2 + B_3u_3 \\ B_2A_{os}^{mn}u_2 + B_3A_{ou}^{mn}u_3 \end{bmatrix} =: \begin{bmatrix} d_1 \\ d_2 \end{bmatrix}. \end{aligned}$$

Note that $d_2 = 0$ if and only if $\begin{bmatrix} u_2 \\ u_3 \end{bmatrix} = 0$. Given that $u_3 \neq 0$, it follows that $\left\| \tilde{B}(t, t+s)u \right\| \geq \|d_2\| > 0$. Therefore,

$$\beta \leq \min_{\{u | u_1 = 0, \|u\| = 1\}} \left\| B_2A_{os}^{mn}u_2 + B_3A_{ou}^{mn}u_3 \right\|,$$

is an appropriate choice to obtain detectability. □

Corollary 5.1.4 (Necessary condition)**.** *Given that (A, C) is not detectable, there does not exist a sequence that is uniformly detectable.*

Proof. If (A, C) is not detectable, then there exists an eigenvalue-eigenvector pair, (λ, v), of A such that $|\lambda| \geq 1$ and $Cv = 0$. For any pair (α, r), assuming $\|v\| = 1$, $\|A^r v\| = |\lambda|^r \geq 1 > \alpha$. So there has to exist (s, β) such that $\|B(t, t+s)v\| \geq \beta$ for uniform detectability. However, the rows of $B(t, t+s)$ consist of vectors of the form $c_i A^k$, for some k, where c_i is a row of C. Now $c_i A^k v = \lambda^k c_i v = 0$ and so $B(t, t+s)v = 0$ no matter what the actual sequence is. Therefore, no sequence of measurements can be uniformly detectable. □

With this result, we can conclude that if and only if the LTI system (A, C) is detectable can a schedule be constructed from the rows of C that is uniformly detectable and, hence, lead to a bounded error covariance. Although a schedule can be naively constructed by just selecting a sequence of measurements that are linearly independent, and then just repeating that sequence, we now seek an algorithm that outputs a uniformly detectable sequence while at the same time trying to optimize the error covariance.

5.2 Modified Greedy

An interesting question to ask is whether or not the greedily constructed schedule is uniformly detectable. The definition of detectability effectively requires that all unstable modes be observable, i.e., be measured at some point in time. Uniform detectability is slightly stricter in that the mode has to be observed within a certain amount of time. If a system has an unstable mode that is unobservable, then the error associated with that mode will grow. Since the greedy choice at every time step is to pick the measurement that decreases the error covariance the most, then at some point in time the "needed" measurement should have enough benefit to be chosen. Therefore, a reasonable expectation is that the greedy schedule will lead to a bounded error covariance.

Example 5.2.1. Consider the pathological system

$$A = I_{3\times 3}, \qquad C = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0.1 & 0 \\ 0 & 0 & 0.01 \end{bmatrix} =: \begin{bmatrix} c_1 \\ c_2 \\ c_3 \end{bmatrix}, \qquad W = V = \sigma_0 = I.$$

Running the greedy algorithm, the resulting value of the objective function is plotted in Figure 5.1. The schedule that the greedy algorithm outputs chooses the measurement 1 at

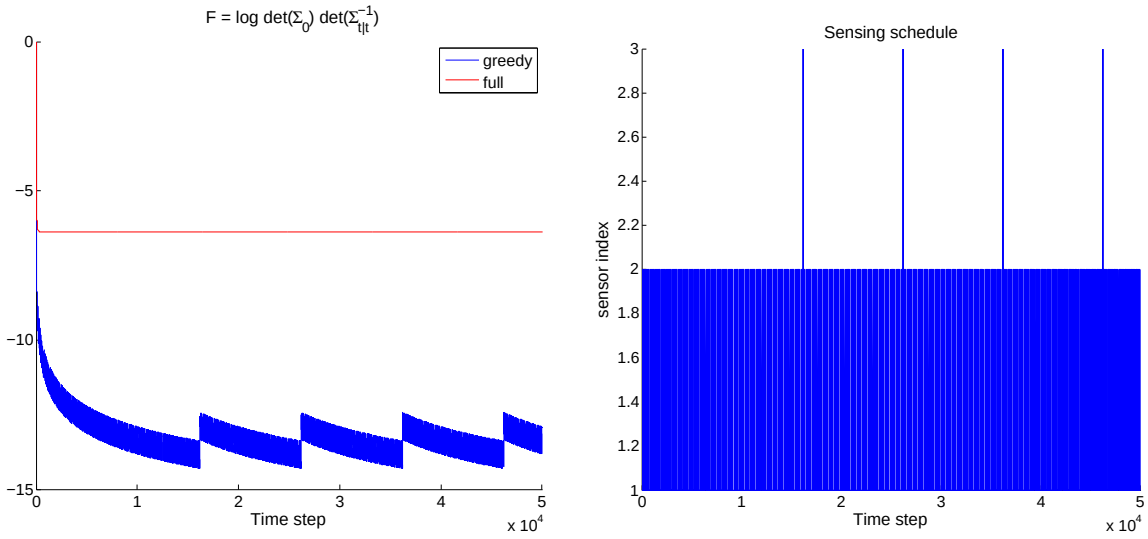


Figure 5.1: (Left) The value of the greedy sequence over time (cf. Example 5.2.1). Also, the "full" sequence is the value if every measurement is made at all times. (Right) The greedy schedule.

almost every time step. The first time measurement 2 is selected is $t = 162$ and then it is repeated approximately every 100 time steps. The first time measurement 3 is selected is $t = 16181$ and then it is repeated approximately every 10000 time steps. As we can see, eventually every measurement is made and so the schedule is actually uniformly detectable (and also seems to give a bounded error covariance) though the time window needed is over 16000 time steps. ▲

Unfortunately, establishing the boundedness of greedy has proved difficult. The main reason appears to be relating the Kalman update performance metric to the sequence observability matrix. As evident from the previous example, even if the greedy algorithm does produce a uniformly detectable sequence, it still does not perform very well. We now present a modified greedy algorithm that ensures that the output sequence is detectable and which attempts to maintain a relatively tight limit for the number of steps required to achieve uniform detectability.

The DETECTABLEGREEDY algorithm is given in Algorithm 3. The idea is to make a greedy choice at each iteration subject to the constraint that the choice of measurement will increase the rank of the matrix M. This matrix emulates the sequence observability matrix $B(t, t+s)$. Once M becomes full rank, a set of the oldest measurements is removed. As a

Algorithm 3: detectableGreedy

Input: F: value function, (A, C, W, V): system parameters, T: time horizon, q: ranks to reduce by.

Output: Sensor schedule with one sensor per time step.

Let $A_{su} := \text{blkdiag}(A_{os}, A_{ou})$ and $C_{su} := \begin{bmatrix} C_2 & C_3 \end{bmatrix}$ (cf. (5.4)).
$p \leftarrow \text{rank}(A_{su})$
`// Construct schedule`
$M \leftarrow 0$
for $t = 1 \dots T$ **do**
    `// Use` (A_{su}, C_{su}) `for constructing` M
    foreach *row* r *of* C_{su} **do**
        if *Appending row* r *multiplied by the proper power of* A_{su} *increases* $\text{rank}(M)$ **then**
            Mark r as valid.
    `// Use` (A, C) `for selection`
    if *None of the rows are valid* **then** Greedily select the best row of C.
    else Greedily select the best row from all valid rows from C.
    Update M with new measurement.
    if $\text{rank}(M) == p$ **then**
        Remove k rows from the top of M such that $\text{rank}(M[k+1 : \text{end}]) = p - q$ but $\text{rank}(M[k : \text{end}]) > p - q$. So $M \leftarrow M[k+1 : \text{end}]A_{su}^{-k}$.

result, M acts as a sliding window and the algorithm attempts to keep this window fully observable. The parameter q controls by how much M is reduced; enough rows are removed such that the rank of M decreases by q (or the observable subspace loses q dimensions). We now show that the algorithm does, in fact, result in a uniformly detectable schedule.

Notation: Henceforth, we will refer to rows of M using the square bracket notation, e.g. $M[2:4]$ is the submatrix formed by extracting rows 2 through to 4 inclusive. Also, parenthesis will be used to refer to a certain time range of M, e.g. if $M(2,4)$ refers to measurements made at times 2 though to 4. In addition, define a rank increasing row (RIR) as a row of the M matrix that increases the rank when considering the rows sequentially starting from the top row. A lot of notation is also borrowed from Theorem 5.1.3: $A_{su} := \text{blkdiag}(A_{os}, A_{ou})$, $C_{su} := \begin{bmatrix} C_2 & C_3 \end{bmatrix}$, $\Theta_{su} := \begin{bmatrix} \Theta_2 & \Theta_3 \end{bmatrix}$, $B_{su} := \begin{bmatrix} B_2 & B_3 \end{bmatrix}$ and $p = \text{rank}(A_{su})$.

Lemma 5.2.2. *The rank of the M matrix will increase within p steps. In other words, consider the matrix at any time such that* $\text{rank}(M(t, t+k)) < p$, *then* $\text{rank}(M(t, t+k+p)) \geq \text{rank}(M(t, t+k)) + 1$.

Proof. The next few measurements will be chosen sequentially from the sequence of matrices $\{C_{su}A_{su}^{k+i}\}_{i=1}^{p}$. Stacking these on top of each other we get $\begin{bmatrix} C_{su} \\ \vdots \\ C_{su}A_{su}^{p-1} \end{bmatrix} A_{su}^{k+1} = \Theta_{su}A_{su}^{k+1}$. Since both Θ_{su} and A_{su} are full rank, $\text{rank}(\Theta_{su}A_{su}^{k+1}) = p$. Therefore, at least one of the rows in $\Theta_{su}A_{su}^{k+1}$ will be LI from the rows of $M(t, t+k)$ and can be added to the sequence. So the rank of M will increase by at least 1 after p more time steps. □

Theorem 5.2.3. *Algorithm 3 produces a uniformly detectable sequence.*

Proof. By Lemma 5.2.2, a RIR will be selected within p times steps and so the maximum size of the M matrix is $p^2 - p + 1$ at the first time M is reduced.

We can prove the following: Before and after any removal of rows (line 12), the first and last rows of M are RIRs, and any RIR is within p steps of the previous RIR.

Assume that before a removal, this property is satisfied. After the removal, the rank will be $p-q$. The new matrix, M_{new}, is a subset of the rows of the old one, M_{old}, multiplied by A_{su}^{-k}. Since A_{su} is full rank, the relationship between rows, in terms of linear independence, will not change (from (P2)), i.e., if two rows are LI in M_{old}, then they are LI in $M_{\text{old}}A_{su}^{-k}$. So a RIR in M_{old} that was not removed will be a RIR in M_{new} since by definition of RIR, it will be LI of all the rows above it. As a result the last row will be a RIR. The first row of M_{new} will also, by definition, be a RIR. Since nothing has been added and the previous RIRs were at most p steps apart, the distance between successive RIRs in M_{new}will continue to be less than or equal to p. Since the rank is $p - q$, a RIR will be added within p steps and since the last row was a RIR, the distance between successive RIRs is still less than or equal to p. Therefore, at the next removal point, the condition will be satisfied.

Using this property, since after removal of rows M is rank $p - q$ and all the RIRs are within p steps of each other, the size after removal is $\leq (p-q-1)p+1 = p^2-(q+1)p+1$. Also, since at most pq rows will be added, the size of the matrix will be $\leq p^2-(q+1)p+1+qp = p^2 - p + 1$ at the next removal point. As a result, the length of M will never exceed p^2.

To show the sequence is uniformly detectable, we can follow the same argument as in the proof of Theorem 5.1.3, i.e., we need to show that $\left\|\tilde{B}(t, t+s)u\right\| > 0$ for some s and

for any vector that has $u_3 \neq 0$. Note that the matrix M in the algorithm corresponds to sections of $B_{su}(t, t+s)$.

M is a sliding window that has size at most p^2. Let $\{M_i\}_i$ be the sequence of full rank matrices that are constructed as the algorithm is run, i.e., M_1 is the first full rank matrix, M_2 is the matrix the second time the algorithm reaches line 12, and so on. So for any time t, $B_{su}(t, t+p^2)$ will contain $M_i A_{su}^k$ for some i and k. Note that $M_i A_{su}^k$ is full rank and therefore so is $B_{su}(t, t+p^2)$.

Take $s = z + 2p^2$. Looking at $\tilde{B}(t, t+s)$, the rows $z+1$ through to $z+2p^2$ will have to contain $M_i A_{su}^k$ for some i and k. Expanding $\tilde{B}(t, t+s)u$,

$$\begin{bmatrix} \tilde{B}(t, t+k-1) \\ \tilde{B}(t+k, t+s)\tilde{A}^k \end{bmatrix} u = \begin{bmatrix} 0 & W & X \\ 0 & 0 & M_i A_{su}^k \\ 0 & 0 & Z \end{bmatrix} \begin{bmatrix} u_0 \\ u_1 \\ u_{23} \end{bmatrix} = \begin{bmatrix} W u_1 + X u_{23} \\ M_i A_{su}^k u_{23} \\ Z A_{su}^k u_{23} \end{bmatrix} =: \begin{bmatrix} d_1 \\ d_2 \\ d_3 \end{bmatrix}.$$

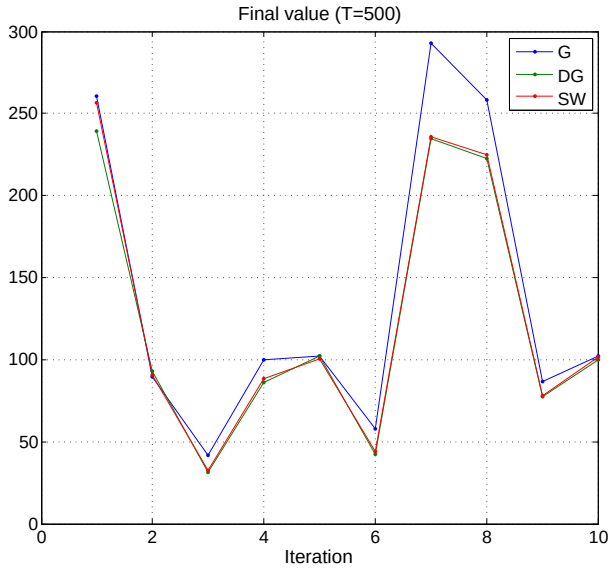
Again, $d_2 \neq 0$ since $u_{23} > 0$ (as we are considering only states with unstable modes). Therefore,

$$\beta \leq \min_{\substack{\{u | u_1 = 0, \|u\| = 1\} \\ j = 1, \ldots, s \\ M}} \left\| M A_{su}^j u_{23} \right\|,$$

where the minimum is taken over all possible M, i.e., all possible full rank M that have $p, \ldots, p^2$ rows. The minimum exists since M has only a finite number of choices as the number of possible measurements is finite. □

Remark 5.2.4 (Complexity of algorithm). The DETECTABLEGREEDY algorithm is basically just a greedy algorithm with two extra steps. Following from the proof of Theorem 4.3.10, the greedy part runs in $O(Tmn^2 + Tn^3)$. In addition, there are two extra steps: check which of the measurements increase rank, and reduce the M matrix. For determining valid rows to choose, a matrix multiplication has to be performed for each row of C ($O(mn^2)$) and the rank of a matrix that is at most $n^2 \times n$ needs to be calculated m times (better than $O(mn^4)$ [28]). For the reduction, the rank of a matrix that is at most $n^2 \times n$ needs to be calculated at most n^2 times ($O(n^6)$). Therefore, the total complexity becomes $O(T(mn^2 + n^3 + mn^2 + mn^4 + n^6))$. This is definitely much worse than the actual complexity. If we assume that in practice M will have $O(n)$ rows, then the complexity of the algorithm is $O(T(mn^3 + n^4))$. Using better algorithms to calculate the rank can improve this further. •

Although this shows that the sequence will indeed result in a bounded error covariance, we do not know what this bound is and whether it will be better than the regular greedy approach.



	1	2	3
G	1	0	9
DG	8	1	1
SW	1	9	0

Figure 5.2: (Left) Final value of each algorithm over all iterations. (Right) Number of times each algorithm achieved each ranking.

5.3 Simulations

Here we present some simulations to investigate the properties of Algorithm 3 and how detectability impacts the bound for the error covariance. We also compare the algorithm to other known techniques. All the simulations are performed using the average trace of the a priori estimate,

$$F(\sigma) = \lim_{T\to\infty} \frac{1}{T} \sum_{t=0}^{T-1} \text{trace}(\rho^T(\Omega_t^\sigma)).$$

For this section, we will refer to the GREEDY algorithm as G and the DETECTABLE-GREEDY algorithm as DG.

5.3.1 Comparison with Sliding Window

In [55], an optimal algorithm is presented to minimize the average trace of the covariance over a finite horizon. The entire tree of possibilities is searched by employing a tree pruning technique that reduces the number of possible branches at every time step using certain

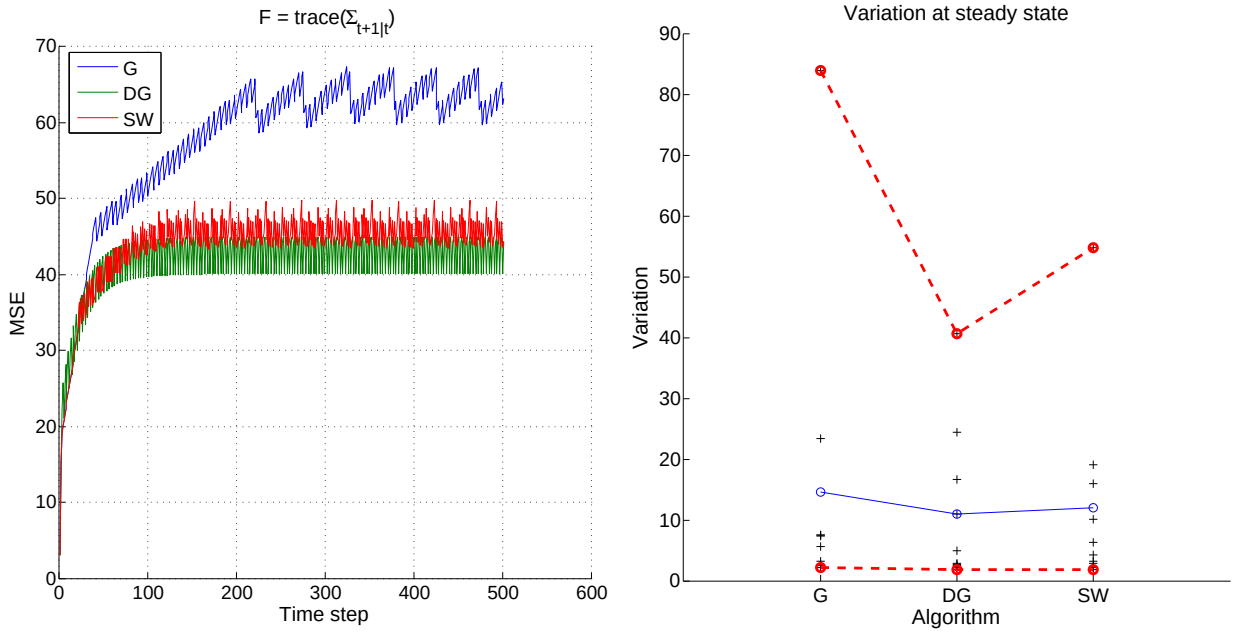
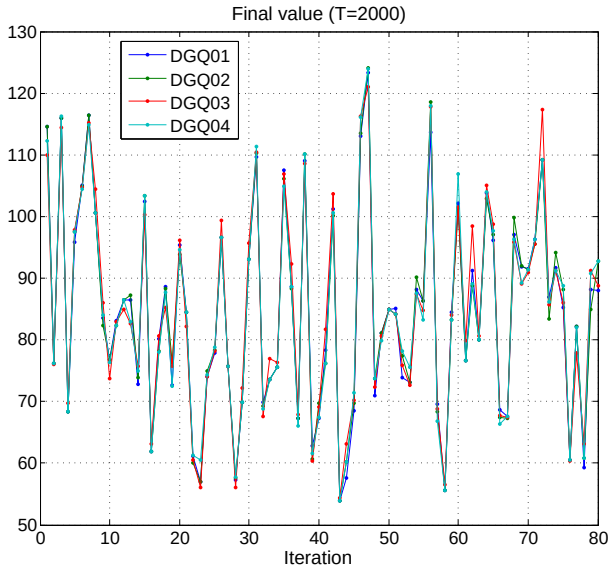


Figure 5.3: (Left) A sample result. The value being plotted is the trace of the covariance (MSE) at each time step. (Right) The variation in the MSE after the covariance settles to a steady cycle for each system. The mean, max and min are also shown.

properties of the covariance update equations. The drawback of this approach is that in order to determine which branches to cut, a convex optimization problem has to be solved for every matrix currently under consideration. This is a time consuming operation and so we were not able to run useful simulations for a decent size time horizon.

Instead, we use a sliding window approximation for comparison. The SLIDINGWINDOW (SW) algorithm is basically an extended greedy such that the optimal is calculated over a certain time window by considering every possible sequence. This is repeated continuously until the desired time horizon is met.

For this experiment, we ran simulations comparing three algorithms: G, DG and SW. For ten randomly generated systems, all three algorithms were executed until $T = 500$. This time horizon was chosen to allow the error covariance to settle to a steady value. The number of states as well as the number of measurement vectors was taken to be 3. The A matrix was taken to be the identity. The C matrix had all its entries uniformly randomly distributed in $[0, \frac{1}{3}]$. The process noise had all its entries uniformly distributed in $[0, 5]$ and the measurement noise was a diagonal matrix with the individual variances chosen uniformly randomly in $[0.5, 1.5]$.



	1	2	3	4	5
G	9	4	5	7	55
DGQ01	26	11	21	16	6
DGQ02	17	27	11	20	5
DGQ03	16	21	12	24	7
DGQ04	12	17	31	13	7

G	9	4	5	7	55
DGQ01	26	11	21	16	6
DGQ02	23	24	9	21	3
DGQ03	16	21	12	24	7
DGQ04	23	18	22	10	7

Figure 5.4: (Left) Final value of each algorithm over all iterations. (Right Top) Number of times each algorithm achieved each ranking. (Right Bottom) Two or more algorithms with the same value are assigned the same ranking.

Figure 5.2 shows the final values of each algorithm over a time horizon of 500. Each of the algorithms was ranked at each iteration and the number of times each rank was achieved is given in the table. As we can see, the DG algorithm seems to outperform the other two; although the output of DG is very similar to SW. The drawback of the sliding window approach is that although the optimal is achieved over the window size, the approximation may get worse the larger the time horizon.

Figure 5.3 shows a sample result for one of the iterations. The value being plotted is not the average trace but the trace of the covariance at every time step, i.e., the mean squared error (MSE). We can see that after an initial transition period, the covariance update becomes somewhat periodic. The second graph gives the amount of fluctuation that happens for each algorithm for each system (as well as the mean, minimum and maximum over all systems) once it has reached the steady state value. As we can see, on average all the algorithms have similar amount of fluctuations, though DG was the best in terms of the worst case.

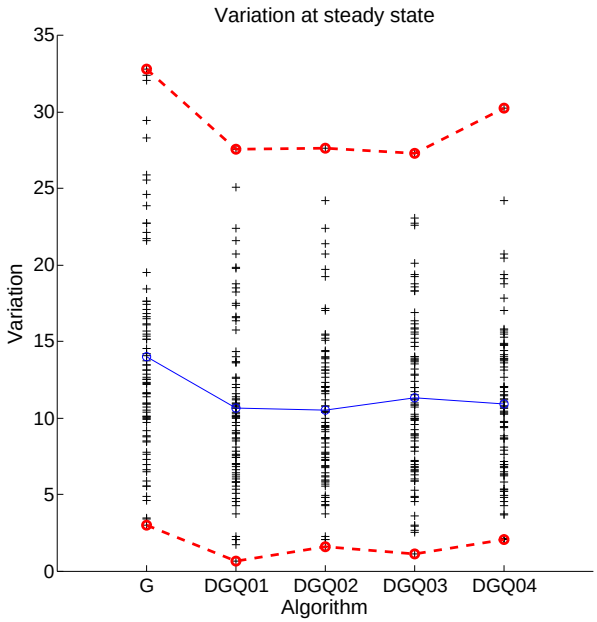


Figure 5.5: The variation in the MSE after the covariance settles to a steady cycle for each system. The mean, max and min are also shown.

5.3.2 Experiment with Reduction Size

The DG algorithm has a parameter q that controls by how much the memory of chosen measurements is reduced. We now investigate the effects of this parameter. In this case, the G and GD (with four different values of q) algorithms were run on 80 systems for a time horizon of $T = 2000$. The number of states was taken to be 4 and the number of possible measurements was 40. Apart from that, the system (A, C, W, V) is set up in the same way as in the previous comparison. The four values of q used are: 1 (DGQ01), 2 (DGQ02), 3 (DGQ03) and 4 (DGQ04).

Figure 5.4 shows the final values of each algorithm over a time horizon of 2000. Each of the algorithms was ranked at each iteration and the number of times each rank was achieved is given in the table. All of the DG algorithms perform better than the G algorithm. Although DQG01 was ranked 1 the most number of times, this was only $\approx 33\%$ of the total number of iterations. There does not seem to be any obvious benefit of varying the parameter q as, generally they all seem to have final values that are close together.

The second graph gives the amount of fluctuation that happens for each algorithm for each system (as well as the mean, minimum and maximum over all systems) once it has

reached the steady state value.

Figure 5.5 gives the amount of fluctuation in the MSE that happens for each algorithm for each system (as well as the mean, minimum and maximum over all systems) once it has reached the steady state value. Again, although all the DG algorithms beat the G algorithm, varying q does not cause any noticeable change in the size of fluctuations.

Chapter 6

Conclusions and Future Directions

6.1 Submodular max-TSP

We examined the max-TSP problem for a submodular objective function. We considered two algorithms; a greedy algorithm which achieves a $\frac{1}{2+\kappa}$ approximation and a 2-matching-based algorithm which achieves a $\max\{\frac{2}{3(2+\kappa)}, \frac{2}{3}(1-\kappa)\}$ approximation (where κ is the curvature of the function). Both algorithms have a complexity of $O(|V|^3)$ in terms of number of oracle calls. We also discussed extending Serdyukov's algorithm. We extended these results to directed graphs and presented simulation results to empirically compare their performance as well as their dependence on curvature. We also considered an approach to integrate edge costs into the problem.

There are several directions for future work. The class of submodular functions is very broad and so adding further restrictions to the properties of the objective function may help give a better idea of how the bounds change for specific situations. For instance, utilizing some sort of locality property to define relationships between edges. Although an attempt was made to extend Serdyukov's algorithm, the question still stands as to whether or not the strategy makes sense in the case of high curvature. Additionally, there are many other simple strategies that could also be extended such as best neighbor or insertion heuristics. One other problem to investigate would be to the case where multiple tours are needed (such as with multiple patrolling robots).

6.2 Sensor Scheduling

For the problem of scheduling sensors to obtain the best state estimate of an underlying process over a finite horizon we showed that, under certain restrictive conditions, the log det of the a posteriori error covariance is a submodular function. Therefore, greedily selecting sensors at each time step yields a $1 - \frac{1}{e^{1-1/e}}$-approximation.

Next, we used the concept of uniform detectability to construct an algorithm that guarantees that the error covariance will be bounded and that the error will die out exponentially when the time horizon is large or infinite. We gave some empirical results comparing the performance of the algorithm against other known techniques.

Although we were able to give a sufficient condition for submodularity of the metric, the question still stands as to whether or not this is a necessary condition or, if not, what a necessary condition would be. Due to the complexity of the Kalman update equations, it is possible that a simple condition does not exist; however, it seems reasonable that if the measurement vectors are large relative to the noise then the desired properties to obtain submodularity and monotonicity may be satisfied.

The DETECTABLEGREEDY algorithm is currently only specified for the case where a single measurement is made at each time step. Extending this to select k sensors would be desirable for a more complete result. A further generalization would be to consider how non-scalar measurements will affect the results. Also, a technique to check detectability at runtime without having to calculate a system decomposition should be investigated. Another thing is that the algorithm outputs a sequence that is bounded but we have not given an actual bound. Finding a way to determine the deviation from optimality will help to quantify the performance. Also, investigating how to combine the uniform detectability condition with other optimization techniques (instead of just using greedy) will help to determine the practicality of this approach.

References

[1] Saeed Alaei and Azarakhsh Malekian. Maximizing sequence-submodular functions and its application to online advertising. *CoRR*, abs/1009.4153v1, 2010.

[2] B. D. O. Anderson and J. B. Moore. *Optimal Filtering*. Dover Publications, New York, 2005.

[3] BDO Anderson and John B Moore. Detectability and stabilizability of time-varying discrete-time linear systems. *SIAM Journal on Control and Optimization*, 19(1):20–32, 1981.

[4] Jonathan Binney, A. Krause, and G. Sukhatme. Informative path planning for an autonomous underwater vehicle. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pages 4791–4796, 2010.

[5] G. Calinescu, C. Chekuri, M. Pal, and J. Vondrak. Maximizing a monotone submodular function subject to a matroid constraint. *SIAM Journal on Computing*, 40(6):1740–1766, 2011.

[6] Mihaela Cardei and Ding-Zhu Du. Improving wireless sensor network lifetime through power aware organization. *Wireless Networks*, 11(3):333–340, 2005.

[7] D. W. Casbeer, D. B. Kingston, R. W. Beard, T. W. Mclain, S.-M. Li, and R. Mehra. Cooperative forest fire surveillance using a team of small unmanned air vehicles. *International Journal of Systems Sciences*, 37(6):351–360, 2006.

[8] J. Clark and R. Fierro. Mobile robotic sensors for perimeter detection and tracking. *ISA Transactions*, 46(1):3–13, 2007.

[9] Michele Conforti and Grard Cornuejols. Submodular set functions, matroids and the greedy algorithm: Tight worst-case bounds and some generalizations of the rado-edmonds theorem. *Discrete Applied Mathematics*, 7(3):251 – 274, 1984.

[10] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. MIT Press, 2 edition, 2001.

[11] Yehuda Elmaliach, Noa Agmon, and Gal A Kaminka. Multi-robot area patrol under frequency constraints. *Annals of Mathematics and Artificial Intelligence*, 57(3-4):293–320, 2009.

[12] Uriel Feige. A threshold of ln n for approximating set cover. *J. ACM*, 45(4):634–652, July 1998.

[13] Uriel Feige, Vahab S Mirrokni, and Jan Vondrak. Maximizing non-monotone submodular functions. *SIAM Journal on Computing*, 40(4):1133–1153, 2011.

[14] Yuval Filmus and Justin Ward. A tight combinatorial algorithm for submodular maximization subject to a matroid constraint. In *Foundations of Computer Science (FOCS), 2012 IEEE 53rd Annual Symposium on*, pages 659–668, 2012.

[15] M. L. Fisher, G. L. Nemhauser, and L. A. Wolsey. An analysis of approximations for maximizing submodular set functions - II. In *Polyhedral Combinatorics*, volume 8 of *Mathematical Programming Studies*, pages 73–87. 1978.

[16] M. L. Fisher, G. L. Nemhauser, and L. A. Wolsey. An analysis of approximations for finding a maximum weight hamiltonian circuit. *Operations Research*, 27(4):pp. 799–809, 1979.

[17] D. Golovin and A. Krause. Adaptive submodularity: Theory and applications in active learning and stochastic optimization. *Journal of Articial Intelligence Research*, 42:427–486, 2011.

[18] Pranava R. Goundan and Andreas S. Schulz. Revisiting the greedy approach to submodular set function maximization. Working Paper, Massachusetts Institute of Technology, 2007.

[19] C. Guestrin, A. Krause, and A. Singh. Near-optimal sensor placements in Gaussian processes. In *Int. Conf. on Machine Learning*, Bonn, Germany, August 2005.

[20] Anupam Gupta, Aaron Roth, Grant Schoenebeck, and Kunal Talwar. Constrained non-monotone submodular maximization: Offline and secretary algorithms. In *Internet and Network Economics*, pages 246–257. Springer, 2010.

[21] V. Gupta, T. H. Chung, B. Hassibi, and R. M. Murray. On a stochastic sensor selection algorithm with applications in sensor scheduling and sensor coverage. *Automatica*, 42(2):251–260, 2006.

[22] William W Hager and Larry L Horowitz. Convergence and stability properties of the discrete Riccati operator equation and the associated optimal control and filtering problems. *SIAM Journal on Control and Optimization*, 14(2):295–312, 1976.

[23] Refael Hassin and Shlomi Rubinstein. Better approximations for max TSP. *Information Processing Letters*, 75:181–186, 1998.

[24] D. Hausmann, B. Korte, and T. A. Jenkyns. Worst case analysis of greedy type algorithms for independence systems. In *Combinatorial Optimization*, volume 12 of *Mathematical Programming Studies*, pages 120–131. Springer Berlin Heidelberg, 1980.

[25] R. A. Horn and C. R. Johnson. *Matrix Analysis*. Cambridge University Press, 2010.

[26] G.E. Hovland and B.J. McCarragher. Dynamic sensor selection for robotic systems. In *IEEE Int. Conf. Robotics and Automation*, pages 272–277. IEEE, 1997.

[27] M.F. Huber, A. Kuwertz, F. Sawo, and U.D. Hanebeck. Distributed greedy sensor scheduling for model-based reconstruction of space-time continuous physical phenomena. In *Information Fusion, 2009. FUSION '09. 12th International Conference on*, pages 102 –109, 2009.

[28] Oscar H Ibarra, Shlomo Moran, and Roger Hui. A generalization of the fast lup matrix decomposition algorithm and applications. *Journal of Algorithms*, 3(1):45–56, 1982.

[29] Volkan Isler and Ruzena Bajcsy. The sensor selection problem for bounded uncertainty sensing models. In *Proceedings of the 4th international symposium on Information processing in sensor networks*, page 20. IEEE Press, 2005.

[30] Satoru Iwata, Lisa Fleischer, and Satoru Fujishige. A combinatorial strongly polynomial algorithm for minimizing submodular functions. *J. ACM*, 48(4):761–777, July 2001.

[31] T. A. Jenkyns. The greedy travelling salesman's problem. *Networks*, 9(4):363–373, 1979.

[32] S. Joshi and S. Boyd. Sensor selection via convex optimization. *IEEE Transactions on Signal Processing*, 57(2):451–462, 2009.

[33] Ioannis K Kookos and John D Perkins. A systematic method for optimum sensor selection in inferential control systems. *Industrial & engineering chemistry research*, 38(11):4299–4308, 1999.

[34] B. Korte and J. Vygen. *Combinatorial Optimization: Theory and Algorithms*, volume 21 of *Algorithmics and Combinatorics*. Springer, 4 edition, 2007.

[35] A. Krause, C. Guestrin, A. Gupta, and J. Kleinberg. Near-optimal sensor placements: Maximizing information while minimizing communication cost. In *Symposium on Information Processing of Sensor Networks*, pages 2–10, Nashville, TN, April 2006.

[36] Andreas Krause and Carlos Guestrin. Near-optimal nonmyopic value of information in graphical models. In *Twenty-first conference on uncertainty in artificial intelligence (UAI)*, page 5, 2005.

[37] X. Lan and M. Schwager. Planning periodic persistent monitoring trajectories for sensing robots in gaussian random fields. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA 13)*, pages 2407–2412, May 2013.

[38] J. Lee, V. S. Mirrokni, V. Nagarajan, and M. Sviridenko. Non-monotone submodular maximization under matroid and knapsack constraints. In *ACM Symposium on Theory of Computing*, pages 323–332, Bethesda, MD, 2009.

[39] Jon Lee, Maxim Sviridenko, and Jan Vondrák. Submodular maximization over multiple matroids via generalized exchange properties. *Math. Oper. Res.*, 35(4):795–806, November 2010.

[40] Entao Liu, Edwin K. P. Chong, and Louis L. Scharf. Greedy Adaptive Compression in Signal-Plus-Noise Models. *CoRR*, abs/1202.3913v5, 2012.

[41] Julin Mestre. Greedy in approximation algorithms. In Yossi Azar and Thomas Erlebach, editors, *Algorithms ESA 2006*, volume 4168, pages 528–539. Springer Berlin / Heidelberg, 2006.

[42] Michel Minoux. Accelerated greedy algorithms for maximizing submodular set functions. In J. Stoer, editor, *Optimization Techniques*, volume 7 of *Lecture Notes in Control and Information Sciences*, pages 234–243. Springer Berlin / Heidelberg, 1978.

[43] Y. Mo, R. Ambrosino, and B. Sinopoli. Sensor selection strategies for state estimation in energy constrained wireless sensor networks. *Automatica*, 47(7):1330–1338, 2011.

[44] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher. An analysis of approximations for maximizing submodular set functions - I. *Mathematical Programming*, 14:265–294, 1978.

[45] Katarzyna Paluch, Marcin Mucha, and Aleksander Madry. A 7/9 - approximation algorithm for the maximum traveling salesman problem. In Irit Dinur, Klaus Jansen, Joseph Naor, and Jos Rolim, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, Lecture Notes in Computer Science, pages 298–311. Springer Berlin Heidelberg, 2009.

[46] Alexander Schrijver. A combinatorial algorithm minimizing submodular functions in strongly polynomial time. *Journal of Combinatorial Theory, Series B*, 80(2):346 – 355, 2000.

[47] Anatoliy I Serdyukov. An algorithm with an estimate for the traveling salesman problem of the maximum. *Upravlyaemye Sistemy*, 25:80–86, 1984.

[48] M. Shamaiah, S. Banerjee, and H. Vikalo. Greedy sensor selection: Leveraging submodularity. In *Decision and Control (CDC), 2010 49th IEEE Conference on*, pages 2572 –2577, 2010.

[49] M. Shamaiah, S. Banerjee, and H. Vikalo. Greedy sensor selection under channel uncertainty. *Wireless Communications Letters, IEEE*, 1(4):376–379, 2012.

[50] A. Singh, A. Krause, C. Guestrin, and W. Kaiser. Efficient informative sensing using multiple robots. *Journal of Artificial Intelligence Research*, 34:707–755, 2009.

[51] B. Sinopoli, L. Schenato, M. Franceschetti, K. Poolla, M. I. Jordan, and S. S. Sastry. Kalman filtering with intermittent observations. *IEEE Transactions on Automatic Control*, 49(9):1453–63, 2004.

[52] Ryan N Smith, Mac Schwager, Stephen L Smith, Burton H Jones, Daniela Rus, and Gaurav S Sukhatme. Persistent ocean monitoring with underwater gliders: Adapting sampling resolution. *Journal of Field Robotics*, 28(5):714–741, 2011.

[53] S. L. Smith, M. Schwager, and D. Rus. Persistent robotic tasks: Monitoring and sweeping in changing environments. *IEEE Transactions on Robotics*, 28(2):410–426, April 2012.

[54] Anke Van Zuylen. Deterministic approximation algorithms for the maximum traveling salesman and maximum triangle packing problems. *Discrete Applied Mathematics*, 161(13-14):2142–2157, 2013.

[55] Michael P. Vitus, Wei Zhang, Alessandro Abate, Jianghai Hu, and Claire J. Tomlin. On efficient sensor scheduling for linear dynamical systems. *Automatica*, 48(10):2482–2493, 2012.

[56] Jan Vondrák, Chandra Chekuri, and Rico Zenklusen. Submodular function maximization via the multilinear relaxation and contention resolution schemes. In *Proceedings of the 43rd annual ACM symposium on Theory of computing*, pages 783–792, 2011.

[57] Justin Ward. A (k+3)/2-approximation algorithm for monotone submodular k-set packing and general k-exchange systems. In *29th International Symposium on Theoretical Aspects of Computer Science*, volume 14, pages 42–53, Dagstuhl, Germany, 2012.

[58] J. Weimer, J. Araújo, A. Hernandez, and K.H. Johansson. Periodic constraint-based control using dynamic wireless sensor scheduling. In *Decision and Control and European Control Conference (CDC-ECC), 2011 50th IEEE Conference on*, pages 4789–4796. IEEE, 2011.

[59] J. E. Weimer, B. Sinopoli, and B. H. Krogh. A relaxation approach to dynamic sensor selection in large-scale wireless networks. In *International Conference on Distributed Computing Systems*, pages 501–506, 2008.

[60] Chun Yang, Lance Kaplan, and Erik Blasch. Performance measures of covariance and information matrices in resource management for target state estimation. *Aerospace and Electronic Systems, IEEE Transactions on*, 48(3):2594–2613, 2012.

[61] Leehter Yao, William A Sethares, and Daniel C Kammer. Sensor placement for on-orbit modal identification via a genetic algorithm. *AIAA journal*, 31(10):1922–1928, 1993.

[62] Wei Zhang, Michael P. Vitus, Jianghai Hu, A. Abate, and C.J. Tomlin. On the optimal solutions of the infinite-horizon linear sensor scheduling problem. In *Decision and Control (CDC), 2010 49th IEEE Conference on*, pages 396–401, 2010.