Abstract—In this article, we consider a class of submodular maximization problems in which decision-makers have limited access to the objective function. We explore scenarios where the decision-maker can observe only pairwise information, i.e., can evaluate the objective function on sets of size two. We begin with a negative result that no algorithm using only \( k \)-wise information can guarantee performance better than \( \frac{k}{n} \). We present two algorithms that utilize only pairwise information about the function and characterize their performance relative to the optimal, which depends on the curvature of the submodular function. Additionally, if the submodular function possesses a property called supermodularity of conditioning, then we can provide a method to bound the performance based purely on pairwise information. The proposed algorithms offer significant computational speedups over a traditional greedy strategy. A by-product of our study is the introduction of two new notions of curvature, the \( k \)-Marginal Curvature and the \( k \)-Cardinality Curvature. Finally, we present experiments highlighting the performance of our proposed algorithms in terms of approximation and time complexity.

Index Terms—Autonomous systems, optimization algorithms, sensor networks, submodular maximization.

I. INTRODUCTION

Submodular maximization has recently generated interest in many decision-making problems, as solutions produced by greedy algorithms can provide strong performance guarantees for computationally difficult problems. Submodular functions are set functions that exhibit the property of diminishing returns. Submodular optimization is a well-studied subject, as these functions model many real-world problems in controls [1], [2], robotics [3], [4], data processing [5], [6], and machine learning [7], [8].

One practical difficulty in implementing algorithms for submodular maximization in complex settings is that the required function evaluations are computationally expensive. This can be attributed to the large-scale characteristics of the system [9], application-specific constraints, such as communication limits [10], or the type of data the objective is required to evaluate [11]. In its most common form, the submodular function is treated as a value oracle, which is repeatedly queried by a greedy strategy to maximize the objective function. Therefore, it is inherently assumed that one can evaluate the function for sets of any size. In practice, however, it may only be possible to evaluate the functions on smaller set sizes due to computational cost.

For example, consider the setting where a company is selecting locations for several new retail stores. The total revenue received by a set of store locations can be modeled as a submodular function: as more stores are added, the marginal benefit of adding a new store is reduced. In the classical greedy algorithm for submodular maximization, we assume we have access to a value oracle to evaluate subsets of store locations. Armed with this oracle, we iteratively add a new store \( s_k \) to the existing set \( \{s_1, \ldots, s_{k-1}\} \) by selecting the location \( s_k \) that maximizes the marginal benefit \( f(s_1, \ldots, s_{k-1}, s) - f(s_1, \ldots, s_{k-1}) \). To evaluate this quantity, the oracle must accurately model the revenue of \( k \) stores, which can be challenging in practice due to their complex interactions: for example, \( s_k \) may reduce the revenue at some \( s_i \), which then may affect some other store’s revenue.

Motivated by the lack of access to the full value oracle in practical settings, in this article, we seek to determine how well we can approximate the maximum value of a submodular function when we have access to a limited set of function values. For most parts of this article, we focus on the case where we can access function values for single elements \( f(s_i) \) and for pairs of elements \( f(s_i, s_j) \). We refer to this as pairwise information. In the motivating example, this corresponds to knowing the total revenue for a single store and the total revenue for any two stores together and nothing more. Note that this restriction on information is severe. A submodular function on a base set of \( N \) elements can be represented as a look-up table with \( 2^N \) values. If only singleton and pairwise information are available, this means we have access to only \( N(N+1)/2 \) values. While we focus on pairwise information, we also extend most results to the case of \( k \)-wise information, where we can evaluate any set of size at most \( k \).
Greedy algorithms have also been applied to submodular maximization in applications including the following:

1) large-scale distributed coverage, where low-power robots seek locations that maximize the area covered by their sensors [12], [13];
2) large-scale transportation networks, where sensing infrastructure locations must be selected to best observe large traffic networks [14] (a similar problem is explored in detail in the simulation results of this article);
3) or sensor selection for state estimation of a linear dynamical system [15].

In these applications, function evaluations may be limited due to computational resources, modeling challenges, sensor limitations, or communication constraints. For example, in sensor coverage, a function evaluation requires an agent to compute the set difference between their own sensor footprint and the union of all other sensor footprints. This requires accurate position information for all other agents. In contrast, pairwise function evaluations require only the distance to each agent (provided by a range sensor), and computation is limited to the intersecting two sets. In transportation networks, accurately modeling the interactions of a large number of agents poses similar challenges to the revenue example above. This article sheds light onto the fundamental question of how well one can optimize a submodular function when given only limited access to its values.

A. Statement of Contributions

We consider the submodular maximization problem where information on the underlying function is limited, in that we only have access to evaluations of sets of size at most \( k \). Let \( X \) be the base set of elements we are optimizing over and \( n \) be the maximum number of elements that can be in our solution set. We begin with a negative result; namely, there exists a class of submodular functions for which no algorithm subject to this constraint can guarantee performance better than \( k/n \) of the optimal value. In light of this, we propose a subclass of functions where we can upper and lower bound the marginal gains the objective function in terms of pairwise information. Using these bounds, we propose two simple greedy algorithms that utilize only pairwise information. We introduce two new notions of curvature named the \( k \)-Marginal Curvature and the \( k \)-Cardinality Curvature, which capture “how submodular” a function is. We then adapt a previous result for approximate value oracles to prove performance bounds for the two algorithms in terms of our new notions of curvature. The two notions provide a new way to understand submodular functions and may be of independent interest. We also show that, using only pairwise information and an additional assumption called supermodularity of conditioning on the function, we can compute optimality bounds for a given solution. We illustrate that the structure of the lower and upper bound estimates of the marginal gains can be exploited to produce an algorithm that runs in exactly \( O(|X| \cdot n) \) without the assumption of a value oracle. Finally, we show experimental results for an autonomous ride service coverage problem that highlights the effectiveness of the algorithms and the time complexity advantages.

B. Related Work

It is well known that maximizing a submodular function subject to a cardinality constraint is NP-hard [16], but if the function is normalized and monotone, then a greedy algorithm provides an approximation factor of \( (1 - 1/e) \). This article considers a similar problem but with additional information constraints. Some important general submodular functions where the greedy algorithm has been extensively explored are graph cut [6], mutual information [17], set cover [18], and facility location [17]. Other traditional constraints that have been considered for submodular maximization include knapsack [19], budget [6], and matroid constraints [20], [21]. Submodular maximization has also been studied in the context of robotics and controls, being used in applications, such as sensor coverage [17], sensor selection for Kalman filtering [15], [22], [23], multirobot exploration objectives [24], voltage control [1], multiagent target tracking [25], informative path planning [26], and control input selection [27].

Recently, other information constraints are being considered in the context of distributed submodular maximization. In these scenarios, a team of agents are attempting to maximize a submodular objective function collaboratively. Each agent has access to their own set of actions and can observe a limited number of decisions made by other agents [10], [28], [29], [30], [31]. In contrast, we consider the case where each decision-maker has limited access to the function \( f \) itself.

Another related concept is the idea of an approximate value oracle [32]. This refers to a black box that takes as input a set (or set and a new element) and outputs an approximation to the true function value (or an approximation to the marginal gains). Results can then be derived on the quality of the resulting solutions for a greedy algorithm using the approximate value oracle. The performance of these algorithms is a function of the approximation factor for the approximate value oracle. In this article, we extend these approximation oracle results to provide approximation bounds in terms of new notions of curvature.

Finally, another aspect of submodular maximization is the computational efficiency of the greedy strategy [9], [33], [34], [35]. Under the assumption that the strategy has access to a value oracle for the objective that can be computed in constant time, the time complexity is \( O(|X| \cdot n) \) [33]. Even though the time complexity is polynomial, for \( X \) with large cardinality, the greedy strategy can become prohibitively expensive to execute. Consequently, alternative implementations have been provided for the greedy strategy that improve computational efficiency by leveraging streaming techniques to only pass over the set \( X \) once [9], [36], or by realizing the set \( X \) as a tree and pruning nodes and edges to reduce the size of \( X \) [34]. Parallelized implementations of the greedy strategy and strategies that greedily maximize over randomly sampled subsets of \( X \) [33] have addressed computational issues for large problems. Most of these techniques focus on reducing the search space being
II. PROBLEM DEFINITION AND INAPPROXIMABILITY

Let $X$ be a set of elements and $2^X$ be the power set of those elements. A set function $f : 2^X \to \mathbb{R}_{\geq 0}$ is submodular if the following property of diminishing returns holds: For all $A \subseteq B \subseteq X$ and $x \in X \setminus B$ we have

$$f(A \cup \{x\}) - f(A) \geq f(B \cup \{x\}) - f(B).$$

We refer to $f(A \cup \{x\}) - f(A)$ as the marginal return of $x$ given $A$, denoted by $f(x|A)$. For simplicity, we denote the objective value of a singleton $f(\{x\})$ by $f(x)$. We also denote marginal return of $x$ with respect to a singleton set $A = \{y\}$ by $f(x|y)$ and refer to it as the pairwise marginal return of $x$ given $y$. In addition to submodularity, throughout this article, we assume that the functions satisfy:

1) Monotonicity: For all $A \subseteq B \subseteq X$, $f(A) \leq f(B)$.

2) Normalization: $f(\emptyset) = 0$.

Another property of submodular functions, we utilize is the notion of curvature [37], [38]. The curvature of a submodular function $f$ is defined as

$$c = 1 - \min_{A \subseteq X, x \in X \setminus A} \frac{f(x|A)}{f(x)}.$$  

(1)

Note that if the value of $c = 0$, the function is modular.

We start by recalling the problem of maximizing a submodular function subject to a cardinality constraint. Let $X$ be a set of elements and let $f : 2^X \to \mathbb{R}_{\geq 0}$ be a monotone normalized submodular function. We wish to solve the following problem:

$$\max_{S \subseteq X} f(S)$$

s.t. $|S| \leq n$.  

(2)

This is the classical submodular maximization problem that can be solved to an approximation factor of $(1 - 1/e)$ using the following simple greedy algorithm, see [16]:

$$x_i = \arg \max_{x \in X \setminus S_{i-1}} f(x|S_{i-1})$$

$$S_i = S_{i-1} \cup \{x_i\}$$  

(3)

where $i$ is the iteration of the algorithm and $S_i$ is the solution produced after $i$ iterations. In what follows, we often refer to this strategy as the full information greedy algorithm. A key focus of our contributions is to understand the limitations of algorithms that only have access to partial information about the objective function. We make this precise in the following definition.

**Definition 2.1 (k-wise Information):** Given a submodular function $f$, the $k$-wise information set is defined as the set of tuples $\{(S, f(S)) | S \subseteq X, |S| \leq k\}$. When $k = 2$, we refer to this as pairwise information.

In this work, we consider algorithms that take as input a base set of elements $X$, a function $f : 2^X \to \mathbb{R}_{\geq 0}$ as a black box, and a number $n$ and which outputs a subset of $S$ with cardinality $|S| \leq n$ in finite time. An algorithm that has access to k-wise information can utilize the black box for $f$ only to evaluate $f$ on sets of size $k$ or less. We denote the class of such algorithms by $\Pi_{k\text{-wise}}$, or $\Pi_{\text{pairwise}}$ when $k = 2$.

The main objective that we have in mind is to study Problem (2) with such limitations. We now present a negative result that addresses the inapproximability of this problem.

**Proposition 2.2:** Consider Problem (2) with $k$-wise information. Then, for every algorithm $\pi \in \Pi_{k\text{-wise}}$, there exists a submodular function $f : 2^X \to \mathbb{R}$ with $|X| \geq 2n$ and nonzero curvature such that

$$f(S^*) \leq \frac{k}{n} f(S^\pi)$$

where $S^\pi$ is the solution constructed by $\pi$ and $S^*$ is the optimal solution.

**Proof:** We begin by constructing a normalized, monotone submodular function $f$. Consider a set $X$ that is partitioned into two disjoint sets $X = V \cup V^*$, where $|V^*| = n$ and $|V| \geq n$. We define the function $f : 2^X \to \mathbb{R}_{\geq 0}$ as

$$f(S) = \min\{|S \cap V|, k\} + |S \cap V^*|.$$  

(4)

This function is normalized and monotone, and given $k$, it assigns a value of $k$ to all sets of size $k$. The set $V$ can be thought of as the general set and $V^*$ is a special set where you are guaranteed to get value if you selected an element from $V^*$. The function, counts the number of elements of $S$ that are in $V^*$. However, all sets $S$ where $|S| \leq k$ get mapped to their cardinality. We now show that $f$ is also submodular. Consider any two sets $A \subseteq B \subseteq X$ and an element $x \in X \setminus B$. We show that

$$f(x|A) \geq f(x|B).$$

First notice that $f(x|A)$ and $f(x|B)$ are each either 0 or 1, since adding an element can increase the function value by at most one. There are two cases to consider.

**Case 1 ($x \in V^*$):** In this case $f(x|A) = 1$, since $A \cup \{x\}$ has one more element in $V^*$ than $A$. Since $f(x|B) \leq 1$, the result follows.

**Case 2 ($x \in V$):** We assume that $f(x|B) = 1$, as otherwise the result holds. Since $f(x|B) = 1$ and $x \in V$, we must have $|B \cap V| < k$. But this implies that $|A \cap V| < k$ since $A \subseteq B$. Thus, $f(x|A) = 1$ and the result holds.

For any set $S$ with $|S| \leq k$, we have that $f(S) = |S|$, which reveals no information on which elements of $S$ are in $V$ or $V^*$. Hence, for any algorithm in $\Pi_{k\text{-wise}}$, the values returned by the black box for $f$ are independent of the choice of $V$ and $V^*$. Given an algorithm $\pi \in \Pi_{k\text{-wise}}$, we can provide the black box containing $k$-wise information (i.e., a black box implementing $f(S) = |S|$) and observe the resulting solution $S^\pi$. We then select any $v^* \in X$ of size $n$ such that $S^\pi \cap V^* = \emptyset$. Equation (4) then defines a function such that $S^\pi$ is a solution returned by $\pi$ with value $f(S^\pi) = k$, and $S^* = V^*$ is the optimal solution with value $f(S^*) = n$. This proves the desired result.  

This result highlights the challenges that arise under $k$-wise information constraints. As shown in the proof, there exists functions where their marginal returns with respect to sets of size $k$ or less, tell you nothing about the marginals with respect to sets with sizes greater than $k$. As we see in the sensor coverage example depicted in Fig. 1, the marginal returns of a single sensor given subset of other sensors can be approximated by taking its area and subtracting the pairwise overlaps of itself to the rest of the element in the subset. These overlaps can

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
be derived from utilizing only pairwise marginals, therefore, pairwise information provides us with information about the higher order marginals.

In the following two sections, we characterize functions where the marginals with respect to sets of size $k$ or smaller are informative of the higher order marginals using new notions of curvature.

### III. PAIRWISE ALGORITHMS

Our main objective in what follows is to leverage pairwise information to find an approximate solution to Problem (2).

#### A. Optimistic Algorithm

A natural strategy is to greedily select elements that maximize the estimated marginal return using only pairwise information. First, note that

$$\min_{x_j \in A} f(x|x_j) \geq f(x|A) \tag{5}$$

which holds by submodularity of $f$, because for all $x_j \subseteq A$, we have $f(x|x_j) \geq f(x|A)$. We will define a simple estimate of the marginal returns of $f$ as the left hand side of (5)

$$\tilde{f}(x|A) := \min_{x_j \in A} f(x|x_j).$$

The pairwise marginal for all $x_j \in A$ upper bounds $f(x|A)$ and hence we choose the minimum as it is the best available estimate of the true value of $f(x|A)$. In a nearly identical style to the classical greedy strategy, we now define an algorithm as follows:

$$x_i = \arg \max_{x \in X \setminus S_{i-1}} \tilde{f}(x|S_{i-1})$$

$$S_i = S_{i-1} \cup \{x_i\}. \tag{6}$$

Throughout this article, we will refer to (6) as the optimistic algorithm. In essence, the optimistic algorithm aims to greedily select elements with maximum potential marginal return.

#### B. Approximate Value Oracles

To characterize the performance of the optimistic algorithm given by (6), we consider the problem through the lens of maximizing a submodular objective function via surrogate objective functions. Following [38], we will discuss how to determine performance guarantees when using such surrogates.

Let $S \subseteq X$ be the result set produced by an algorithm. Suppose we have an ordering of the elements of $S = \{x_1, \ldots, x_n\}$, then we work with the set $S_i = \{x_1, \ldots, x_i\}$. Now let $\{x_1^g, \ldots, x_n^g\} \subseteq X$ be such that each $x_i^g$ maximizes the marginal return of $f$ conditioned on $S_{i-1}$, i.e.,

$$x_i^g = \arg \max_{x \in X \setminus S_{i-1}} f(x|S_{i-1}).$$

The set $\{x_1^g, \ldots, x_n^g\}$ represents the elements that a greedy algorithm with full information about the objective $f$ would have selected if it had previously selected $S_{i-1}$. Using these values, we can now measure the quality of a given algorithm’s choices compared to that of an algorithm with full information about the objective. We do this by finding $\alpha_i \in \mathbb{R}_+$, for $i \in \{1, \ldots, n\}$ such that

$$\alpha_i f(x_i|S_{i-1}) \geq f(x_i^g|S_{i-1}) \tag{7}$$

By the greedy choice property of $x_i^g$, we have that

$$f(x_i|S_{i-1}) \leq f(x_i^g|S_{i-1}).$$

Hence, $\alpha_i \geq 1$, for all $i \in \{1, \ldots, n\}$. From this point on, we call each $\alpha_i$ the approximation factor associated with $x_i$.

In the general framework proposed in [38], the objective is to greedily maximize multiple surrogate objective functions, and to use these to generate approximate solutions. For our problem of submodular maximization with only pairwise information, we simply maximize using a single surrogate function $f(x|S)$. We provide a simplified version of [38, Th. 1] as follows.

**Theorem 3.1:** Suppose that $S = \{x_1, \ldots, x_n\} \subseteq X$ is the set of elements selected by an algorithm and $\{\alpha_1, \ldots, \alpha_n\}$ are the set of approximation factors that satisfy (7). Let $S^*$ be the optimal solution to Problem (2). Then

$$f(S) \geq \left(1 - e^{-\frac{1}{\sum_{i=1}^n \alpha_i}} \right) f(S^*). \tag{8}$$

Given that we only maximize one surrogate function and in order to keep this article self-contained, we provide a proof of this result in the Appendix, which is simpler than the general result established in [38]. Note that Theorem 3.1 relies on $f$ being a normalized monotone submodular function. This result can be applied to any algorithm for Problem (2), not just algorithms that only have access to pairwise information. An interesting remark about Theorem 3.1 is that the performance bound depends essentially on the average of the approximation factors. Some of these factors could be large compared to the others, but as long as most of them are small, good performance is maintained.

**Remark 3.2 (On The Ordering Of The Result Set):** As described in this section, we impose an ordering of the elements produced by an algorithm. From the proof provided in the Appendix, the result holds for any ordering. Therefore, the tightest bound produced by an algorithm is achieved by taking the maximum over all permutations of the output set of the algorithm. From a practical perspective, computing the bound for every permutation is infeasible. The algorithms we present later in this work, select elements iteratively. Therefore, the ordering we impose on elements in the resulting solution is the order that the elements were selected.
C. Optimistic Algorithm Approximation Performance

We aim to provide approximation guarantees for the optimistic algorithm. To give an intuition for what we are about to present, we consider the following example.

Example 3.3: Consider the scenario depicted in Fig. 2. Here, we wish to select four sensors to maximize the area of their combined footprints. One of the simplest algorithms that satisfies the pairwise information constraint is the uninformed greedy strategy

\[
\begin{align*}
 x_i &= \arg \max_{x \in X \setminus S_{i-1}} f(x) \\
 S_i &= S_{i-1} \cup \{x_i\}.
\end{align*}
\]

We refer to this algorithm as uninformed because it only uses the most basic information about \( f \), which its values evaluated on single elements. For the uninformed greedy strategy, the scenario described in Fig. 2 could potentially lead to a poor performance. This strategy cannot distinguish between its choices and, therefore, could select four sensors that almost perfectly overlap with each other (i.e., in the same pile), resulting in a low objective value. Alternatively, if we had used the optimistic algorithm, once one element is selected from a pile, the pairwise upper bound on the other elements in a pile would be low. In later iterations, the optimistic algorithm would avoid selecting elements in piles where elements have been previously selected from. Interestingly, we see that for each \( i \), the difference between \( f(x_i|S_{i-1}) \) and \( f(x_i|S_i) \) is small. We notice that in these scenarios, the value of \( f(x_i|S_{i-1}) \) provides accurate information about the value of \( f(x_i|S_i) \). This is the idea that we want to capture in the following result.

Theorem 3.4: Let \( S_{i-1} \subseteq X \) be the partial solution of optimistic algorithm after \((i-1)\) iterations, and let \( x_i \in X \) be the element selected at the \( i \)th iteration. Then, we have that

\[
\alpha_i = \begin{cases} 
 1 & i \in \{1, 2\} \\
 \frac{f(x_i|S_{i-1})}{f(x_i|S_i)} & i > 2
\end{cases}
\]

satisfy (7) for all \( i \leq n \).

Proof: Let \( x_i^\alpha \) be the true greedy choice at iteration \( i \) given \( S_{i-1} \). For \( i = \{1, 2\} \), we have that

\[
\bar{f}(x|S_{i-1}) = f(x|S_{i-1}).
\]

Hence, \( x_i = x_i^\alpha \) and therefore, we can let \( \alpha_1^\alpha = \alpha_2^\alpha = 1 \). For \( i > 2 \), based on from (7) let \( \alpha_i^\text{min} \) be the smallest value such that (7), which can be written as

\[
\alpha_i^\text{min} = \frac{f(x_i^\alpha|S_{i-1})}{f(x_i|S_{i-1})} \leq \frac{\bar{f}(x_i|S_i)}{f(x_i|S_{i-1})} \leq \frac{\bar{f}(x_i|S_i)}{f(x_i|S_{i-1})} \leq \frac{\bar{f}(x_i|S_i)}{f(x_i|S_{i-1})}
\]

where (12) holds by definition and (13) holds by the greedy choice property of the optimistic algorithm. Setting \( \alpha_i^\alpha \) to be the right-hand side of (13), we conclude the proof.

The following corollary is an immediate consequence of Theorem 3.4.

Corollary 3.5: Let \( S \subseteq X \) be the solution produced by the optimistic algorithm and \( S_{i-1} \subseteq S \) be the partial solution after \((i-1)\) iterations of the optimistic algorithm and let \( x_i \in S \) be the element selected at the \( i \)th iteration, then we have

\[
f(S) \geq 1 - e^{-\frac{1}{2} \left(2 + \sum_{i=1}^{\infty} \frac{f(x_i|S_i)}{f(x_i|S_{i-1})}\right)} f(S^*).
\]

We see that the approximation performance of the algorithm is dictated by the sum in the exponent. We can interpret the exponent as the mean of the set

\[
\left\{1, 1, \frac{f(x_2|S_2)}{f(x_3|S_2)}, \ldots, \frac{f(x_{n}|S_{n-1})}{f(x_3|S_{n-1})}\right\}.
\]

That implies that, to get adequate performance from the optimistic algorithm, we need the value of \( \bar{f}(x_i|S_i) \) to be close to \( f(x_i|S_{i-1}) \) on average.

We also see that the \( \frac{f(x_i|S_{i-1})}{f(x_i|S_i)} \) is closely related to the traditional notion of curvature. Let us define the following quantity.

Definition 3.6 (k-Marginal Curvature): The \( k \)-marginal curvature of \( f \) given \( S \subseteq X \) and \( x \in X \setminus S \) is defined as

\[
c_k(x|S) = 1 - \max_{A \subseteq S; |A| < k} \frac{f(x|S)}{f(x|A)}.
\]

To analyze the optimistic algorithm that only has access to pairwise information, we will work with the 2-marginal curvature, which can be written as

\[
c_2(x|S) = 1 - \max_{y \in S} \frac{f(x|S)}{f(x|y)} = 1 - f(x|S \cup \{y\}).
\]

Remark 3.7: This 2-marginal curvature is similar to the traditional notion of curvature (1), but characterizes the relationship between the values of the pairwise upper bounds \( \bar{f} \) and true values of \( f \). A key difference between the two curvatures, is that there exist functions where the values of the 2-marginal curvatures can be close to 0 even though the value of traditional
curvature is close to 1. The sensor coverage function described in Fig. 2, is an example of a function where the traditional curvature is close to 1 and the values of the 2-marginal curvatures are close to 0.

This allows us to rewrite (14) as follows:
\[
  f(S) \geq \left( 1 - e^{-\frac{1}{k} \left( 2 + \sum_{i=1}^{n} c_2(x_i|S_{-1}) \right)} \right) f(S^*).  
\]  

We now can characterize the worst case performance in terms of the average of the 2-marginal curvatures, which capture the intuition from Example 3.3. In Fig. 2, the elements will have 2-marginal curvatures close to zero, resulting in a strong approximation bound.

Note that, this result is consistent with Proposition 2.2. The bound in (16) is dependent the 2-marginal curvatures of the function. Given the function in Proposition 2.2, if we let \( S_n \) be the worst-case solution as described in the proof, then the worst-case lower bound provided in Corollary 3.5 is consistent with the worst-case in Proposition 2.2.

Corollary 3.5, provides a worst-case performance for the optimistic strategy, given a function with a particular set of 2-marginal curvatures. The corollary shows that there exist functions, where an algorithm with only pairwise information can provide strong approximation guarantees.

D. Extension to k-Wise Information

The analysis from Section III-C can be naturally extended to the problem of observing k-wise information. Suppose that we wish to approximately solve Problem (2) using an algorithm that only has k-wise information available to it. We extend the pairwise optimistic algorithm to the k-wise optimistic algorithm as follows. Let us define an upper bound on the marginal returns using k-wise information. Let \( x \in X \) and \( S \subseteq X \) then we have the following upper bound
\[
  \min_{A \subseteq S, |A| < k} f(x|A) \geq f(x|S) 
\]  

which holds by submodularity of \( f \). We will denote the left hand side of (17) as
\[
  \tilde{f}_k(x|S) := \min_{A \subseteq S, |A| < k} f(x|A). 
\]

We can now define the k-wise optimistic algorithm as
\[
  x_t = \arg \max_{x \in X \setminus S_{i-1}} \tilde{f}_k(x|S_{i-1}) 
  
  S_i = S_{i-1} \cup \{ x_t \}. 
\]

By submodularity, we have that \( f(x|S) \geq \tilde{f}_k(x|S) \geq f(x|S) \) for \( k > 1 \). Our result is stated next.

**Theorem 3.8:** Let \( S_{i-1} \subseteq X \) be the partial solution of k-wise optimistic algorithm after \((i-1)\) iterations and \( x_i \in X \) be the element selected during the \( i \)th iteration. Then, we have that
\[
  \alpha_i^{opt,k} = \left\{ \begin{array}{ll}
  1 & \text{if } i \leq k \\
  \frac{\tilde{f}_k(x_i|S_{i-1})}{f(x_i|S_{i-1})} & \text{if } i > k
\end{array} \right.
\]

satisfy (7) for all \( i \leq n \).

The proof is nearly identical to the one of Theorem 3.4 and can be found in the Appendix. The following result is an immediate consequence of Theorem 3.8.

**Corollary 3.9:** Let \( S \subseteq X \) be the solution produced by the k-wise optimistic algorithm, then we have
\[
  f(S) \geq \left( 1 - e^{-\frac{1}{k} \left( 2 + \sum_{i=k+1}^{n} c_k(x_i|S_{i-1}) \right)} \right) f(S^*). 
\]

Note that using this result, we can rewrite (20) as
\[
  f(S) \geq \left( 1 - e^{-\frac{1}{k} \left( 2 + \sum_{i=k+1}^{n} c_k(x_i|S_{i-1}) \right)} \right) f(S^*). 
\]

Comparing this to the scenario with pairwise information, we see that access to more information improves approximation guarantees. In particular, since \( c_2(x|S) \geq c_k(x|S) \) for all \( S \subseteq X \) and \( x \in X \setminus S \), we can guarantee that
\[
  \frac{1}{n} \left( k + \sum_{i=k+1}^{n} 1 - c_k(x_i|S_{i-1}) \right) \geq \frac{1}{n} \left( 2 + \sum_{i=2}^{n} 1 - c_2(x_i|S_{i-1}) \right). 
\]

This implies that approximation bound in Corollary 3.9 is stronger than Corollary 3.5.

E. Computational Cost of k-Wise Optimal Algorithm

Having access to k-wise information provides us with stronger approximation bounds, but we tradeoff computation performance. We are required to compute the minimum marginal overall subsets \( A \subseteq S_{i-1} \), where \( |A| < k \) for each \( x \in X \). When \( k < |S_i| \), we need to check \( \binom{|S_i|}{k} \) subsets of \( S \) to find the minimum. This becomes expensive to do computationally as \( S_{i-1} \) grows larger. If \( k = 3 \), the computation of each marginal is quadratic in \( |S_{i-1}| \) and can be expensive to compute. In general, a simple implementation of the k-wise optimistic algorithm has a time complexity of \( O(|X| \cdot n \cdot \binom{n}{k}) \). From a practical perspective, we can actually compute the pairwise optimistic algorithm efficiently; we will discuss this in Section V.

IV. PAIRWISE ALGORITHMS UTILIZING SUPERMODULARITY OF CONDITIONING

In this section, we introduce an additional property that a submodular function can possess, which is useful when we only have access to pairwise information. This property which is called supermodularity of conditioning and is related to monotonicity, allows us to compute performance bounds for an algorithm “post hoc” using only pairwise information.

A. Post Hoc Performance Bounds

To characterize the approximation performance of an algorithm \( \pi \in \Pi_{\text{pairwise}} \) using Theorem 3.4, we are required to compute the full marginal of the function \( f \), which may not be available in practice. Alternatively, after we execute an algorithm \( \pi \) to produce a solution with ordering
\[
  S^\pi = \{ x_1^\pi, \ldots, x_n^\pi \} \subseteq X 
\]
we can determine \( \gamma \in \mathbb{R}_{\geq 0} \) such that
\[
  f(S^\pi) \geq \gamma f(S^\ast). 
\]

This is done by bounding \( \alpha_i \) in (7) using only pairwise information. As described in the proof of Theorem 3.4, the smallest value of \( \alpha_i \) that will satisfy (7) is \( \alpha_i^{\text{min}} \).
Let
\[ S^n_\pi = \{x^n_1, \ldots, x^n_T\} \subseteq X \]
be the partial solution of \( S^n \). Then, we have that
\[ \alpha^{\text{min}}_i = \frac{f(x^n_i | S^n_{\pi - 1})}{f(x^n_i | S^n_{\pi - 1}) - f(x^n_i | S^n_{\pi - 1})}. \]

By lower bounding \( f(x^n_i | S^n_{\pi - 1}) \) using pairwise information, we obtain an \( \alpha_i \) that satisfies (7), and therefore, Theorem 3.1 allows us to find \( \gamma \) that satisfies (23).

If we impose an additional monotonicity property on \( f \) called supermodularity of conditioning, then we are able to find a lower bound on the marginal returns of \( f \) using only pairwise information.

**Definition 4.1 (Supermodularity of Conditioning):** A submodular function \( f \) possess the property of supermodularity of conditioning if for all \( S \subseteq X, A \subseteq B \subseteq X \) and \( C \subseteq X \backslash B \), we have that
\[ f(S | A) - f(S | A \cup C) \geq f(S | B) - f(S | B \cup C). \quad (24) \]

Supermodularity of conditioning is a higher order monotonicity property, which describes how the redundancy of two sets are affected by conditioning. Suppose that \( A = \emptyset \), the redundancy between \( S \) and \( C \) is \( f(S) - f(S | C) \), then by further conditioning by \( B \) reduces the redundancy. Supermodularity of conditioning has been used in the context of distributed submodular maximization in [10]. Some notable examples of functions that exhibit supermodularity of conditioning are set coverage, area coverage, and probabilistic set coverage. We recall the following result from [10].

**Lemma 4.2 (Pairwise Redundancy Bound):** Let \( f \) be a submodular function on \( X \) that exhibits supermodularity of conditioning and let \( A, B, C \subseteq X \) be disjoint subsets. Then
\[ f(A | B) - f(A | B \cup C) \leq \sum_{c \in C} (f(c) - f(c | A)). \quad (25) \]

We can now state a result establishing a lower bound on the marginal return.

**Theorem 4.3 (Pairwise Marginal Lower Bound):** Let \( f \) be a submodular function that exhibits supermodularity of conditioning. Then, for \( x \in X \) and \( S \subseteq X \)
\[ f(x | S) \geq f(x) - \sum_{x_j \in S} (f(x) - f(x | x_j)). \quad (26) \]

**Proof:** Since \( f \) exhibits supermodularity of conditioning, applying Lemma 4.2 with \( A = \{x\}, B = \emptyset \) and \( C = S \), we have
\[ f(x) - f(x | S) \leq \sum_{j \in S} (f(x_j) - f(x_j | x)) = \sum_{j \in S} (f(x) - f(x | x_j)) \quad (27) \]
where the last equality hold by the definition of the marginal return, yielding the result. 

For \( x \in X \) and \( S \subseteq X \), we define
\[ \int f(x | S) := f(x) - \sum_{x_j \in S} (f(x) - f(x | x_j)). \]

We can now directly use this lower bound on the marginal return to bound \( \alpha^{\text{min}}_i \). We now have that \( \alpha^{\text{min}}_i \leq \alpha_i^{\text{pairwise}} \), where
\[ \alpha_i^{\text{pairwise}} = \left\{ \begin{array}{ll}
\max_{x \in \mathcal{X} \setminus S_i} \frac{f(x | S^n_{\pi - 1})}{f(x | S^n_{\pi - 1}) - f(x | S^n_{\pi - 1})} & \text{if } f(x | S^n_{\pi - 1}) \geq 0 \\
\infty & \text{otherwise}
\end{array} \right. \]
\[ \text{Algorithm 1: Pairwise Information Post-Hoc Bound.} \]
**Input:** \( S^n, X \)
**Result:** \( \gamma \) such that \( f(S) \geq \gamma f(S^*) \)
1. \( S_0^n \leftarrow \emptyset; \)
2. for \( i = 1, \ldots, n \) do
3. \quad select \( x^n_i \) from \( S^n \setminus S^n_{i-1}; \)
4. \quad if \( f(x^n_i | S^n_{i-1}) \geq 0 \) then
5. \quad \quad \alpha_i \leftarrow \max_{x \in \mathcal{X} \setminus S^n_{i-1}} \frac{f(x | S^n_{i-1})}{f(x | S^n_{i-1}) - f(x | S^n_{i-1})};
6. \quad else
7. \quad \quad \alpha_i \leftarrow \infty;
8. \quad S^n_i \leftarrow S^n_{i-1} \cup \{x^n_i\};
9. end
10. \quad \gamma \leftarrow 1 - e^{-\frac{1}{n} \sum_{i=1}^{n} \alpha_i};

We now present an algorithm that, given \( \pi \in \Pi^{\text{pairwise}} \) and pairwise information about \( f \), produces a worst-case performance bound \( \gamma \) such that the solution \( S^n \) satisfies \( f(S^n) \geq \gamma f(S^*) \).

Algorithm 1 provides us the means to find performance bounds for an arbitrary algorithm \( \pi \) given only pairwise information about \( f \). This does not guarantee the performance before execution, but it does provide a way to verify performance of an algorithm without having to explicitly compute \( f(S^n) \) or \( f(S^*) \).

We end this section with a remark about supermodularity of conditioning.

**Remark 4.4 (On Supermodularity of Conditioning):** Note that assuming \( f \) possess supermodularity of conditioning does not affect the hardness results for submodular maximization. It is shown [10] that the weighted set cover problem possesses the supermodularity of conditioning while still satisfying the hardness results.

### B. Pessimistic Algorithm

We next propose another pairwise algorithm which we will call the pessimistic algorithm, given by
\[ x_i = \arg \max_{x \in \mathcal{X} \setminus S_{i-1}} f(x | S^n_{i-1}) \]
\[ S_i = S_{i-1} \cup \{x_i\}. \quad (29) \]

This algorithm enjoys similar guarantees as the optimistic algorithm when curvature assumptions are made but can outperform the optimistic algorithm in certain scenarios. Similar to the optimistic algorithm, we greedily select elements with the highest guaranteed value, which is exactly greedily minimizing the
approximation factor $\alpha_i$ in Algorithm 1. What differs from the
optimistic algorithm is that we require the additional assumption
of supermodularity of conditioning on the objective function, and
take advantage of it. Later in our experimental results, we will show
the effectiveness of the pessimistic algorithm for a probabilistic
coverage problem.

Using an alternative definition of curvature, we can produce
a similar performance bound as the optimistic algorithm. Let us
define the $k$-cardinality curvature as follows.

**Definition 4.5** ($k$-Cardinality Curvature): We define the $k$-
function at $x \in X$ as

$$
\tau_k = 1 - \min_{x \in x, A \subseteq X, |A| < k} \frac{f(x|A)}{f(x)}.
$$

(30)

What differs between this notion of curvature and the $k$-
marginal curvature is that it compares the values of the marginals
of $x$ with respect to sets of size less than $k$ to the values of $f$
evaluated on a singletons.

The 2-cardinality curvature can be written as

$$
\tau_2 = 1 - \min_{x \in X, y \in X \setminus \{x\}} \frac{f(x|y)}{f(x)}.
$$

This quantity satisfies

$$
\tau_2 \geq 1 - \frac{f(x|y)}{f(x)}
$$

for all $x, y \in X$.

Using the lower bound (26), we have that

$$
f(x|S) = f(x) - \sum_{x_j \in S} f(x) - f(x|x_j)
$$

$$
= f(x) \left(1 - \sum_{x_j \in S} \left(1 - \frac{f(x|x_j)}{f(x)}\right)\right)
$$

$$
\geq f(x)(1 - |S|\tau_2).
$$

Since we know that $f(x|S) \geq 0$

$$
f(x|S) \geq f(x)(1 - \min\{|S|\tau_2, 1\}).
$$

(31)

This leads to the following result.

**Theorem 4.6:** Let $f$ be a normalized monotone submodular
function that possesses supermodularity of conditioning. For the
solution produced by the pessimistic algorithm the approximation
factors

$$
\alpha_i = \begin{cases}
1 & i \leq 2 \\
1 & i > 2
\end{cases}
$$

satisfy (7) for all $i \leq n$.

The proof can be found in the appendix. The following
corollary immediately follows.

**Corollary 4.7:** Let $S \subseteq X$ be the solution produced by the
pessimistic algorithm then we have

$$
f(S) \geq \left(1 - e^{-\frac{1}{n} \sum_{i=1}^{n} (1 - \min\{(i-1)\tau_2, 1\})}\right) f(S^*)
$$

(33)

Similar to the optimistic algorithm, we see that if $\tau_2$ is small,
$f(x|S)$ closely represents the true value of $f(x|S)$. This bound
on performance can be loose relative to the bound produced by
Algorithm 1 due to (31) being coarse. The post hoc bound
produced by Algorithm 1 will provide a tighter bound on per-
formance than Corollary 4.7.

**Remark 4.8:** The new notions of curvature in Definitions 3.6
and 4.5 are related to the traditional definition of curvature. Let $c$
be the traditional curvature as described in (1), $S \subseteq X$ and
$x \in X \setminus S$, both the $k$-marginal and $k$-cardinality curvature have
similar inequalities

$$
c \geq c_k(x|S) \quad \text{and} \quad c \geq \tau_k.
$$

(34)

There are also scenarios where $c$ can be 1 and either $c_2(x|S)$ or
$\tau_2$ can be small. In Fig. 1, we see that $\tau_2$ will be small but $c(x|S)$
can be large. Suppose that the disks in Fig. 1, have area 1, and
let $x$ be disk 6 and $S$ be disk 5, 7, and 8. Then, $c(x|S)$ will be
large because, $f(x|S) \approx 2/3$ and $f(x|S)$ is small resulting in a
larger 2-marginal curvature. As previously described, $\tau_2$ is small
in this example because for the disks, $x$ and $y$ that have the most
overlap, we have $f(x|y) \approx 2/3$ and $f(x) = 1$. Fig. 2, describes
the opposite case where $\tau_2 \approx 1$ and $c(x|S) \approx 0$ for any $S \subseteq X$ and
$x \in X \setminus S$.

**C. Comparison of Optimistic and Pessimistic Algorithms**

1) **Tightness of Bounds:** We can compare the perfor-
mances of the optimistic and pessimistic algorithms by compar-
ing the exponents in (16) and (33). Note that the algorithms will
have the best approximation bound if the exponents evaluate to
−1. The performance of each algorithm is dependent on the cor-
responding notions of curvature. For the optimistic algorithm,
we wish that the 2-marginal curvatures are close to zero for each
$x_i$ and $S_{i-1}$. For the pessimistic algorithm, we instead wish that
the 2-cardinality curvature is close to zero. One downside that
the pessimistic algorithm has is that each term of the sum has the
2-cardinality curvature multiplied by $i - 1$. This means that
when the cardinality constraint $n$ is large, the min $\{i - 1, \tau_2, 1\}$
will saturate and result in the later terms of the sum to be
evaluated to 0, hindering the guaranteed performance of the
pessimistic algorithm. Fig. 2 depicts an example where the
optimistic algorithm will have tighter performance bounds,
and Fig. 1 conversely shows an example where the pessimistic
algorithm will have tighter performance bounds provided by on
our notions of curvature.

2) **Assumptions Required:** It is important to note that the
pessimistic algorithm requires that the function $f$ possesses the
property of supermodularity of conditioning. This is a strong
assumption on the functions and limits the number of applica-
tions the pessimistic algorithm can be applied. The optimistic
algorithm on the other hand can be applied to arbitrary sub-
modular functions. An advantage of the pessimistic algorithm is
that the performance bound is computable using only pairwise
information. The performance bounds of the optimistic requires
the ability to compute the objective function on sets of arbitrary
size.

3) **Empirical Results:** As we show in our experimental
results in Section VI, for the particular problem, we explore,
the pessimistic algorithm tends to out perform the optimistic
algorithm in terms of approximation performance. Our experi-
ments by no means show how the algorithms perform in every
situation, but highlight the achievable performance of the two pairwise algorithms.

V. TIME COMPLEXITY COMPARISON OF PAIRWISE ALGORITHMS TO CLASSICAL GREEDY

A practical issue with the classical greedy strategy is that for large problems, the strategies are expensive to compute [39]. It is a common assumption in the literature that we have a value oracle for the submodular objective function that is computable in constant time. This assumption leads to a time complexity of \( O(|X| \cdot n) \). In practical applications, the objective must be computed using a polynomial-time algorithm. This would result in a time complexity of \( O(|X| \cdot n \cdot T_{\text{eval}}(n)) \) where \( T_{\text{eval}}(n) \) is the cost of computing \( f(S) \) given that \( |S| = n \).

The pairwise algorithms can exploit evaluation of \( f(A) \) only on sets \( A \subseteq X \) and \( |A| \leq 2 \) to get significant time complexity improvements. The cost of computing each pairwise marginal \( f(x|x_j) \) is \( T_{\text{eval}}(2) \) and we can take advantage of this fact to improve our algorithms.

Let \( S = \{x_1, \ldots, x_n\} \subseteq X \) be the set selected by a pairwise algorithm and \( S_i = \{x_1, \ldots, x_i\} \). The pairwise algorithms obtain their performance gains from the fact we can write \( f(x|S_i) \) and \( f(x|x_i) \) recursively. For the upper bound we have that for \( x \in X \)
\[
\bar{f}(x|S_i) = \min_{x_j \in S_i} \{f(x|x_j)\} = \min \left\{ \min_{x_j \in S_i} \{f(x|x_j)\}, f(x|x_i) \right\} = \min \left\{ \bar{f}(x|S_{i-1}), f(x|x_i) \right\}.
\]

Similarly, for each \( x \in X \) for the lower bound have the following:
\[
\underline{f}(x|S_i) = f(x) - \sum_{x_j \in S_i} f(x) - f(x|x_j)
= f(x) - \sum_{x_j \in S_{i-1}} f(x) - f(x|x_j) - (f(x) - f(x|x_i))
= f(x|S_{i-1}) - (f(x) - f(x|x_i)).
\]

Finding \( \bar{f}(x|S_i) \) and \( \underline{f}(x|S_i) \) can be computed in constant time as a function of \( f(x|x_i) \) and \( f(x|S_{i-1}) \) or \( f(x|S_{i-1}) \), respectively. We can leverage this fact to compute each iteration of the pairwise greedy algorithms efficiently. For each iteration of the optimistic algorithm, we compute the set \( E_i = \{f(x|S_{i-1}) : x \in X \setminus S_{i-1}\} \). We compute \( x_i \) by finding the element with maximum value in \( E \). We can compute \( E_{i+1} \) via the recursive definition (35), which can be done in linear time. Since we can compute both \( E_{i+1} \) and the maximum of \( E_i \) in linear time with respect to the size of \( X \), the cost of each iteration of the algorithm is \( O(|X| \cdot n) \). The exact same procedure can be done for the pessimistic algorithm. Therefore, both the pairwise algorithms have time complexity of exactly \( O(|X| \cdot n) \).

The efficiency of the pairwise algorithms introduces a new tradeoff for practical applications of submodular maximization. The pairwise algorithms can avoid paying the cost \( T_{\text{eval}}(n) \) while computing each marginal return. We can tradeoff approximation performance guarantees for Problem (2) for execution speed improvements. This can be useful in scenarios where a user needs to repeatedly and quickly obtain approximate solutions to a submodular maximization problem, and are not that sensitive to the quality of the solution. As we will show in our experimental results, the execution time improvements in using the pairwise algorithms can be significant while simultaneously still providing relatively strong approximation performance. If the function being maximized has favorable curvature conditions then the losses in the guaranteed performance from using the pairwise algorithms can be minimal.

VI. SIMULATION RESULTS

In this section, we benchmark the proposed algorithms in a simulated application of providing autonomous ride services in New York City with electric vehicles. We focus on a coverage problem of selecting a set of charging locations for vehicles for which they can best respond to customer demand. From a historical dataset provided by the NYC Taxi & Limousine Commission [40], we know that throughout the day the geographical distribution of customer demand is changing.

New York City is split into 263 Taxi districts and we assume that the charging stations are located at the centroids of each of these districts, see Fig. 3. We wish to select a subset of charging locations that maximize the expected customer demand that can efficiently serviced from these locations. We say a customer can be efficiently serviced if it can be picked up with a delay of at most \( t \) minutes. Let \( E \) be the set of districts and let \( X \) be the set of stations. Let \( p_{\text{pick}}^e \) be the probability that a vehicle deployed from station \( x \in X \) can pick up a passenger in district \( e \in E \) in \( t \)-minutes. For simplicity, we assume that the ride requests originate from the centroids of the districts. Finally, let \( v_e \) be the demand in district \( e \), which is modeled as the estimated number of pick up requests in district \( e \) in a specified time interval, based on historical data.

Let \( S \subseteq X \) be a set of stations. Then, the objective we want to maximize, which we will call the hidden objective function
f, is written as follows:

\[ f(S) = \sum_{e \in E} \left( 1 - \prod_{x \in S} (1 - p_e(x)) \right) v_e. \]  

(37)

The hidden objective function is more formally known as the probabilistic coverage function and was used for a related sensor coverage problem in [10].

Suppose, we do not have access to the entire hidden objective function due to computation and/or modeling challenges, and thus, the optimization is solved using only pairwise information. Given the pairwise information constraint, we can compute the expected demand that can be serviced by a single station and a pair of stations as

\[ f_1(x) := f(x) = \sum_{e \in E} p_e(x) v_e \]

and

\[ f_2(x, y) := f(x, y) = \sum_{e \in E} (1 - (1 - p_e(x))(1 - p_e(y))) v_e. \]

We will model \( p_e \) using a Gaussian Kernel function

\[ p_e = \frac{e^{-d(x, e)^2}}{r_a^2} \]

where \( d : \mathbb{R}^2 \times \mathbb{R}^2 \rightarrow \mathbb{R}_+ \) is a distance metric, and \( r_a \) is a tune-able parameter that dictates the range of distances where a vehicle could be quickly deployed to service a ride. For our experiments, we used the Euclidean distance metric for simplicity but the metric could be changed to better model the real system.

The objective function is a normalized monotone submodular function that exhibits supermodularity of conditioning [10], which allows us to apply all of our results. To decide which stations should be selected during different time intervals throughout the day, we estimate \( v_e \) for each \( e \in E \) from the historical data and then attempt to solve

\[ S^* \in \arg \max_{S \subseteq X, |S| \leq n} f_h(S). \]

We compare the optimistic and pessimistic algorithms ability to maximize \( f \) while only given access to \( f_1(x) \) and \( f_2(x, y) \) for \( x, y \in X \) to the full information greedy algorithm, which can evaluate \( f \) on sets of any size. We compute the true objective \( f(S) \) for the pairwise algorithms to compare against the full information greedy algorithm’s performance.

A. Approximation Performance Experimental Results

We tested the performance of the optimistic and pessimistic strategies using historical New York City ride service data. The dataset used included the pick-up times and locations of all the “for hire vehicle” rides in the month of January 2020 [40]. We tested the performance of algorithms on varying distributions and we split the data up by pick-up time. We made 12 sections each corresponding to a unique two-hour window of the day. For each of the subsets, we estimated values \( v_e \) for each district by taking the average number of rides in the district. We executed the three algorithms on each of the 12 sets to compare results, which are summarized in Fig. 4.

Fig. 4 shows the performance for different numbers of charging stations selected. We see that all three algorithms have similar performance for low values of \( n \) but then begin to diverge after 15 stations are selected. The pessimistic strategy performs significantly better than the optimistic strategy. The pessimistic algorithm yielded a value no worse than 90% of the full information greedy algorithm’s value and the optimistic yielded a value no worse than 67% across all trials.

Fig. 5 compares the worst-case lower bounds of the optimistic and pessimistic algorithms computed from the estimated approximation factors in (28) on the same trials as used in Fig. 4 as well as the worst case performance bounds produced by Corollaries 3.5 and 4.7. As the number of stations selected increases, the lower bounds on performance degrade in all cases. For the pessimistic algorithm, this is due to the fact that the lower bounds on the marginals also exhibit diminishing returns and continually selecting elements maximizing the lower bound drives the denominator of (28) down. This causes high values of the estimated \( \alpha_1, \ldots, \alpha_n \) and decreasing approximation bounds. The optimistic algorithm does not actively minimize the estimated approximation factors, which are reflected in both
the percentage of rides covered and approximation bounds. As observed in Fig. 4, we see that both pairwise algorithms are still performing similarly to the full information algorithm even though the lower bounds in Fig. 5 suggests otherwise. We also see that the theoretical performance bound for the optimistic algorithm is relatively close to the bound produced by Algorithm 1 for the pessimistic algorithm. The pessimistic algorithm's theoretical bound is much lower than the rest of the bounds, due to the fact that the average $\tau_2$ for each trial was near 1 with a value of 0.89. This highlights the benefits of utilizing Algorithm 1 when computing performance bounds for the pessimistic algorithm.

This experiment reveals that the bound produced by Algorithm 1 becomes less accurate as $n$ increases. This is due to the fact that the approximation factors measure the multiplicative difference from the true greedy choices. The true marginals for elements selected near the end of execution tend to be smaller. Thus, the difference in the overall objective values could be small, but the multiplicative difference could be large which is reflected in the lower bound on the performance.

Fig. 6 looks at the performance of the algorithms over different subsets of the historical data. We plotted the objective value for 25 stations using each of the algorithms for each of the 12 subsets of data. We also plotted the max value of the objective function each of the algorithms could possibly achieve in the time frame. From Fig. 6, we see that for each time frame, the pessimistic algorithm is essentially as effective as the full information greedy algorithm but the optimistic greedy algorithm is less effective. This shows that our function $f$ is not an example of a function where the performance is near the universal lower bound described in Theorem 3.1: Both algorithms are performing near the full information greedy strategy which is at least 63% of optimal. This is far from approximately 8% of optimal as dictated by Theorem 3.1.

B. Time Complexity Experimental Results

Using the same data used for the performance experiments, we measured the time efficiency of the three algorithms as well. We measured the execution time of the algorithms on each of the 12 data subsets and plotted the average execution time in terms of $n$. Each of the experiments was computed using Python 3.7 on a 2017 Macbook Pro with a 3.1 GHz Dual-Core Intel i5 with 8 GB 2133 MHz LPDDR3 RAM. Fig. 7 summarizes the results from the execution time experiment. For each trial, the size of $|X|$ was the same. The relationship between the execution times of the pairwise algorithms and the value of $n$ is linear. This relationship is as expected given our time complexity analysis in Section V. Using the pairwise greedy strategies and the implementation details from Section V, reduces the time complexity from quadratic to linear time in terms of $n$. The pessimistic and optimistic algorithms share similar execution times, which resulted in the two lines overlapping.

In Fig. 8, we see that for both pairwise algorithms the ratio of the execution time of the full information greedy algorithm and the pairwise algorithms is almost linear. This verifies that the pairwise algorithms result in a reduction in time complexity by almost a factor of $n$ for this particular objective function.
VII. CONCLUSION

In this article, we have studied the problem of submodular maximization when an algorithm has limited access to the objective function. In general, without any additional assumptions about the function an algorithm cannot guarantee a better performance than $k/n$ given $k$-wise information about the objective. We proposed two strategies that utilizes only pairwise information, which have performance guarantees dependent on new notions of curvature. We also provided a method to measure the performance of algorithms with limited access to the objective function in hindsight under the assumption that the function possesses supermodularity of conditioning. The two proposed strategies can be computed efficiently and provide new tradeoffs between approximation performance and time complexity. We also provided experimental results highlighting the performance of both algorithms.

In future work, we plan on extending these results to scenarios in distributed systems where a team of agents can only observe a subset of the other agents’ decisions as well as partially observe the objective values. We also are exploring how $k$-wise information can be leveraged in similar style as pairwise lower bound to produce new limited information greedy strategies.

APPENDIX

The following is a proof of Theorem 3.1.

Proof: Let $S^*$ be the solution to Problem (2). Recall from [16, Proposition 2.1] that $f$ is a monotone submodular set function on $X$ if and only if $f(T) \leq f(S) + \sum_{x_j \in T \setminus S} f(x_j | S)$ for all $T \subseteq X$. If $S$ is empty then we have

$$f(S^*) \leq \sum_{x_i \in S^*} f(x_i) \leq n f(x_1) \leq n \alpha_1 f(x_1). \quad (38)$$

Then, if we apply the lemma again with $S_j$, we have

$$f(S^*) \leq f(S_j) + \sum_{x_j \in S \setminus S_j} f(x_j | S_j). \quad (39)$$

We also know that

$$\alpha_{j+1} f(x_{j+1} | S_j) \geq \max_{x_{j+1} \in S \setminus S_j} f(x_j | S_j) \geq f(x_j | S_j). \quad (40)$$

We now substitute (40) into (39) and write $f(S_j)$ as sum of its marginals to get

$$f(S^*) \leq f(S_j) + \sum_{x_j \in S \setminus S_j} \alpha_{j+1} f(x_{j+1} | S_j) \leq \sum_{i=1}^{j} f(x_i | S_{i-1}) + n \alpha_{j+1} f(x_{j+1} | S_j). \quad (41)$$

Equation (41) holds since $|S \setminus S_j| \leq n$. We will now rearrange (41) to get the following:

$$f(x_{j+1} | S_j) \geq \frac{1}{\alpha_{j+1} n} f(S^*) - \frac{1}{\alpha_{j+1} n} \sum_{i=1}^{j} f(x_i | S_{i-1}). \quad (42)$$

Now we will add $\sum_{i=1}^{j} f(x_i | S_{i-1})$ to both sides of (42) and simplify

$$\sum_{i=1}^{j+1} f(x_i | S_{i-1}) \geq \frac{1}{\alpha_{j+1} n} f(S^*) + \frac{\alpha_{j+1} n - 1}{\alpha_{j+1} n} \sum_{i=1}^{j} f(x_i | S_{i-1}). \quad (43)$$

We will now prove by induction on $j$ that

$$\sum_{i=1}^{j} f(x_i | S_{i-1}) \geq \frac{1}{\alpha_{j+1} n} f(S^*) + \frac{\alpha_{j+1} n - 1}{\alpha_{j+1} n} \sum_{i=1}^{j} f(x_i | S_{i-1}). \quad (44)$$

Then, after rearranging, we arrive at

$$f(S_j) \geq \frac{\prod_{i=1}^{j} (\alpha_i n) - \prod_{i=1}^{j} (\alpha_i n - 1)}{\prod_{i=1}^{j} (\alpha_i n)} f(S^*) \quad (45)$$

proving the base case. Now assuming the claim holds for $j - 1$. We will apply the inductive hypothesis to (43)

$$\sum_{i=1}^{j} f(x_i | S_{i-1}) \geq \frac{1}{\alpha_j n} f(S^*) + \frac{\alpha_j n - 1}{\alpha_j n} \sum_{i=1}^{j-1} f(x_i | S_{i-1}) \quad (46)$$

proving the inductive hypothesis. If we take $j = n$, we arrive at

$$f(S_n) \geq \frac{\prod_{i=1}^{n} (\alpha_i n) - \prod_{i=1}^{n} (\alpha_i n - 1)}{\prod_{i=1}^{n} (\alpha_i n)} f(S^*) \quad (47)$$

We will now lower bound right coefficients on $f(S^*)$ to simplify the bound. We can now cancel out the denominator of the coefficient to get

$$\prod_{i=1}^{n} (\alpha_i n) - \prod_{i=1}^{n} (\alpha_i n - 1) = 1 - \prod_{i=1}^{n} \frac{\alpha_i n - 1}{\alpha_i n} \quad (48)$$

We can now upper bound each term in the product using $1 + x \leq e^x$ with $x = \frac{1}{\alpha_i n}$ to get a lower bound:

$$1 - \prod_{i=1}^{n} \left(1 - \frac{1}{\alpha_i n}\right) \geq 1 - \prod_{i=1}^{n} e^{-\frac{1}{\alpha_i n}} \quad (49)$$

Using this lower bound yields our result. \newline

Proof of Theorem 3.8

Proof: Let $x_i$ be the true greedy choice at iteration $i$ given $S_{i-1}$. For $i \leq k$, we have that $x_i = x_i^k$ by the definition of $\tilde{x}(x_i | S_{i-1})$. Therefore, we have $\alpha_i^{*k} = \cdots = \alpha_k^{*k} = 1$. The minimum possible approximation factor we have can be written as

$$\alpha_i \text{min}^* = \frac{f(x_i^k | S_{i-1})}{f(x_i | S_{i-1})}. \quad (49)$$

Any approximation factor $\alpha_i$ such that $\alpha_i \geq \alpha_i \text{min}^*$ will satisfy (7). We will now upper bound $\alpha_i \text{min}^*$ as follows:

$$\alpha_i \text{min}^* = \frac{f(x_i^k | S_{i-1})}{f(x_i | S_{i-1})} \leq \frac{\tilde{x}_k(x_i^k | S_{i-1})}{f(x_i | S_{i-1})} \quad (50)$$
where (50) holds by the definition of the upper bound. Equation (51) holds by the greedy choice property of the $k$-wise optimistic algorithm. Therefore, the right-hand side of (51) is a valid approximation factor. We set $\alpha^\text{opt,k}_i$ to the right-hand side of (51), we conclude our proof.

Proof of Theorem 4.6.

Proof: Let $x^g_i$ be the true greedy choice at iteration $i$ given $S_{i-1}$. For $i = \{1, 2\}$, we have that

$$f(x_i|S_{i-1}) = f(x|S_{i-1}).$$

Hence, $x_i = x^g_i$ and therefore, we can let $\alpha^\text{pes}_1 = \alpha^\text{pes}_2 = 1$. The minimum approximation factor that we can achieve can be written as follows:

$$\alpha^\text{min}_i = \frac{f(x^g_i|S_{i-1})}{f(x_i|S_{i-1})} \leq \frac{f(x^g_i|S_{i-1})}{f(x_i|S_{i-1})} \leq \frac{f(x^g_i|S_{i-1})}{f(x^g_i|S_{i-1})} \leq \frac{f(x^g_i|S_{i-1})}{f(x^g_i|S_{i-1})} \leq \frac{1}{1 - \min\{i (i - 1) \gamma_2, 1\}} \leq 1 - \min\{i (i - 1) \gamma_2, 1\}$$

(52) holds by the definition of the lower bound, (53) holds by the greedy choice property of the pessimistic strategy and finally (54) holds by (31). Therefore, if we set $\alpha^\text{pes}$ to the right-hand side of (55), then $\alpha^\text{pes}_i$ is a valid approximation factor for Theorem 3.1.

REFERENCES

[1] Z. Liu, A. Clark, P. Lee, L. Bushnell, D. Kirschen, and R. Poovendran, “Submodular optimization for voltage control,” IEEE Trans. Power Syst., vol. 33, no. 1, pp. 502–513, Jan. 2018.
[2] J. Qin, I. Yang, and R. Rajagopalan, “Submodularity of storage placement optimization in power networks,” IEEE Trans. Autom. Control, vol. 64, no. 8, pp. 3268–3283, Aug. 2019.
[3] M. Roberts et al., “Submodular trajectory optimization for aerial 3D scanning,” in Proc. IEEE Int. Conf. Comput. Vis., 2017, pp. 5334–5343.
[4] A. Krause and C. Guestrin, “Submodularity and its applications in optimized information gathering,” ACM Trans. Intell. Syst. Technol., vol. 2, no. 4, pp. 1–20, Jul. 2011.
[5] K. Wei, R. Iyer, and J. Bilmes, “Submodularity in data subset selection and active learning,” in Proc. Int. Conf. Mach. Learn., 2015, pp. 1954–1963.
[6] H. Lin and J. Bilmes, “Multi-document summarization via budgeted maximization of submodular functions,” in Proc. Hum. Lang. Technol.: Conf. North Amer. Chapter Assoc. Comput. Linguistics, Los Angeles, CA, USA, 2010, pp. 912–920.
[7] S. Sun, M. Zadimoghaddam, A. Krause, and A. Karbasi, “Probabilistic submodular maximization in sub-linear time,” in Proc. Int. Conf. Mach. Learn., Sydney, NSW, Australia, 2017, pp. 3241–3250.
[8] D. Sharma, A. Kapoor, and A. Deshpande, “On greedy maximization of entropy,” in Proc. Int. Conf. Mach. Learn., 2015, pp. 1330–1338.
[9] A. Badanidiyuru, B. Mirzasoleiman, A. Karbasi, and A. Krause, “Streaming submodular maximization: Massive data summarization on the fly,” in Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining, New York, NY, USA, 2014, pp. 671–680.
[10] M. Corah and N. Michael, “Distributed submodular maximization on partition matroids for planning on large sensor networks,” in Proc. IEEE Conf. Decis. Control, 2018, pp. 6792–6799.
[11] A. Krause, J. Leskovec, C. Guestrin, J. Vanbriesen, and C. Faloutsos, “Efficient sensor placement optimization for securing large water distribution networks,” J. Water Resour. Plan. Manage., vol. 134, no. 6, pp. 516–526, 2008.
[12] M. Rubenstein, C. Ahler, N. Hoff, A. Cabrera, and R. Nagpal, “Kilobots: A low cost robot with scalable operations designed for collective behaviors,” Robot. Auton. Syst., vol. 62, no. 7, pp. 966–975, 2014.
[13] X. Sun, C. G. Candras, and X. Meng, “A submodularity-based approach for multi-agent optimal coverage problems,” in Proc. IEEE Conf. Decis. Control, 2017, pp. 4082–4087.
[14] N. Mehr and R. Horowitz, “A submodular approach for optimal sensor placement in traffic networks,” in Proc. Annu. Amer. Control Conf., 2018, pp. 6553–6558.
[15] A. Hashemi, M. Ghasemi, H. Vikalo, and U. Topcu, “Randomized greedy sensor selection: Leveraging weak submodularity,” IEEE Trans. Autom. Control, vol. 66, no. 1, pp. 199–212, Jan. 2021.
[16] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher, “An analysis of approximations for maximizing submodular set functions—I,” Math. Program., vol. 14, no. 1, pp. 265–294, 1978.
[17] A. Krause and D. Golovin, “Submodular function maximization,” Tractability, vol. 3, pp. 71–104, 2014.
[18] U. Feige, “A threshold of for approximating set cover,” J. ACM, vol. 45, no. 4, pp. 634–652, Jul. 1998.
[19] R. Iyer and J. Bilmes, “Submodular optimization with submodular cover and submodular knapsack constraints,” in Proc. Int. Conf. Neural Inf. Process. Syst. - Volume 2, New York, NY, USA, 2013, pp. 2436–2444.
[20] G. Calinescu, C. Chekuri, M. Pál, and I. Vondrak, “Maximizing a monotone submodular function subject to a matroid constraint,” SIAM J. Comput., vol. 40, no. 6, pp. 1740–1766, 2011.
[21] Q. Hou and A. Clark, “Robust maximization of correlated submodular functions under cardinality and matroid constraints,” IEEE Trans. Autom. Control, vol. 66, no. 12, pp. 6148–6155, Dec. 2021.
[22] L. F. O. Chamon, G. I. Pappas, and A. Ribeiro, “The mean square error in Kalman filtering sensor selection is approximately supermodular,” in Proc. IEEE 56th Annu. Conf. Decis. Control, 2017, pp. 343–350.
[23] S. T. Jauwad and S. L. Smith, “Submodularity and greedy algorithms in sensor scheduling for linear dynamical systems,” Automatica, vol. 61, pp. 282–288, 2015.
[24] M. Corah and N. Michael, “Distributed matroid-constrained submodular maximization for multi-robot exploration: Theory and practice,” Auton. Robots, vol. 43, no. 2, pp. 485–501, Feb. 2019.
[25] L. Zhou, V. Tzoumas, G. J. Pappas, and P. Tokekar, “Resilient active target tracking with multiple robots,” IEEE Robot. Autom. Lett., vol. 4, no. 1, pp. 129–136, Jan. 2019.
[26] S. T. Jauwad and S. L. Smith, “Informativle path planning as a maximum traveling salesman problem with submodular rewards,” Discr. Appl. Math., vol. 186, pp. 112–127, 2015.
[27] D. Sahabandu, A. Clark, L. Bushnell, and R. Poovendran, “Submodular input selection for synchronization in Kuramoto networks,” in Proc. IEEE Conf. Decis. Control, 2020, pp. 5840–5847.
[28] B. Gharesifard and S. L. Smith, “Distributed submodular maximization with limited information,” IEEE Trans. Control Netw. Syst., vol. 5, no. 4, pp. 1635–1645, Dec. 2018.
[29] D. Grimsman, M. S. Ali, J. P. Hespanha, and J. R. Marden, “The impact of information in distributed submodular maximization,” IEEE Trans. Control Netw. Syst., vol. 6, no. 4, pp. 1334–1343, Dec. 2019.
[30] D. Grimsman, M. R. Kirchner, J. P. Hespanha, and J. R. Marden, “The impact of message passing in agent-based submodular maximization,” in Proc. IEEE Conf. Decis. Control, 2020, pp. 530–535.
[31] H. Sun, D. Grimsman, and J. R. Marden, “Distributed submodular maximization with parallel execution,” in Proc. Amer. Control Conf., 2020, pp. 1477–1482.
[32] P. R. Goundan and A. S. Schulz, “Revisiting the greedy approach to submodular set function maximization,” Optim. Online, pp. 1–25, 2007.
[33] B. Mirzasoleiman, A. Badanidiyuru, A. Karbasi, J. Vondrak, and A. Krause, “Lazier than lazy greedy,” in Proc. AAIM Conf. Artif. Intell., 2015, vol. 29, no. 1, pp. 1812–1818.
[34] T. Zhou, H. Ouyang, J. Bilmes, Y. Chang, and C. Guestrin, “Scaling submodular maximization via pruned submodularity graphs,” in Proc. Int. Conf. Artif. Intell. Statist., Fort Lauderdale, FL, USA, 2017, vol. 54, pp. 316–324.
Andrew Downie (Member, IEEE) received the B.A.Sc. degree in mathematics and engineering from Queen’s University, Kingston, ON, Canada, in 2019. He is currently working toward the M.A.Sc. degree in electrical and computer engineering specializing in systems and controls from the University of Waterloo, Waterloo, ON, Canada.

His main research focus is on submodular maximization under different information constraints.

Stephen L. Smith (Senior Member, IEEE) received the B.Sc. degree in engineering physics from Queen’s University, Kingston, ON, Canada, in 2003, the M.A.Sc. degree in electrical and computer engineering from the University of Waterloo, Waterloo, ON, Canada, in 2005, and the Ph.D. degree in mechanical engineering from the University of California, Santa Barbara, Santa Barbara, CA, USA, in 2009.

He is currently a Professor with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada, where he holds a Canada Research Chair in Autonomous Systems. He is also a faculty affiliate with the Vector Institute, Toronto, ON, Canada. From 2009 to 2011, he was a Postdoctoral Associate with the Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, USA. His main research interests lie in control optimization for autonomous systems, with a particular emphasis on robotic motion planning and coordination.

Dr. Smith was the recipient of the Early Researcher Award from the Province of Ontario in 2016, the NSERC Discovery Accelerator Supplement Award in 2015, and Outstanding Performance Awards from the University of Waterloo in 2016 and 2019. He is a Licensed Professional Engineer (P.Eng.), an Associate Editor for the IEEE TRANSACTIONS ON ROBOTICS and the IEEE OPEN JOURNAL OF CONTROL SYSTEMS. He was previously Associate Editor for the IEEE TRANSACTIONS ON CONTROL OF NETWORK SYSTEMS (2017–2022), and was a General Chair of the 2021 30th IEEE International Conference on Robot and Human Interactive Communication (RO-MAN).

Bahman Gharesifard (Member, IEEE) received the B.Sc. degree in mechanical engineering and the M.Sc. degree in control and dynamics from Shiraz University, Shiraz, Iran, in 2002 and 2005, respectively, and the Ph.D. degree in mathematics from Queen’s University, Kingston, ON, Canada, in 2009.

He is currently a Professor with the Electrical and Computer Engineering Department, University of California, Los Angeles, Los Angeles, CA, USA. He was an Associate Professor, from 2019 to 2021, and an Assistant Professor, from 2013 to 2019, with the Department of Mathematics and Statistics, Queen’s University. He was an Alexander von Humboldt Research Fellow with the Institute for Systems Theory and Automatic Control, University of Stuttgart, Stuttgart, Germany, in 2019–2020. He held Postdoctoral positions with the Department of Mechanical and Aerospace Engineering, University of California, San Diego, San Diego, CA, USA, 2009–2012 and with the Coordinated Science Laboratory, University of Illinois at Urbana-Champaign, Champaign, IL, USA, from 2012 to 2013. His research interests include systems and controls, distributed control, distributed optimization, machine learning, social and economic networks, game theory, geometric control theory, geometric mechanics, and applied Riemannian geometry.

Dr. Gharesifard was the recipient of the 2019 CAIMS-PIMS Early Career Award, jointly awarded by the Canadian Applied and Industrial Math Society and the Pacific Institute for the Mathematical Sciences, an Alexander von Humboldt Foundation Research Fellowship for experienced researchers in 2019, an NSERC Discovery Accelerator Supplement in 2019, the SIAG/CST Best SICON Paper Prize in 2021, and the Canadian Society for Information Theory (CSIT) Best Paper Award in 2022. He was a finalist (as an advisor) for the Best Student Paper Award at the American Control Conference in 2017. He was also a recipient of the Engineering and Applied Science First Year Instructor Teaching Award in 2015 and 2017. He is currently an Associate Editor for the IEEE CONTROL SYSTEM LETTERS and IEEE TRANSACTIONS ON NETWORK CONTROL SYSTEMS.