# Online Distributed Sensor Selection

Daniel Golovin
Caltech

Matthew Faulkner
Caltech

Andreas Krause
Caltech

## ABSTRACT

A key problem in sensor networks is to decide which sensors to query when, in order to obtain the most useful information (e.g., for performing accurate prediction), subject to constraints (e.g., on power and bandwidth). In many applications the utility function is not known a priori, must be learned from data, and can even change over time. Furthermore for large sensor networks solving a centralized optimization problem to select sensors is not feasible, and thus we seek a fully distributed solution. In this paper, we present *Distributed Online Greedy* (DOG), an efficient, distributed algorithm for repeatedly selecting sensors online, only receiving feedback about the utility of the selected sensors. We prove very strong theoretical no-regret guarantees that apply whenever the (unknown) utility function satisfies a natural diminishing returns property called *submodularity*. Our algorithm has extremely low communication requirements, and scales well to large sensor deployments. We extend DOG to allow observation-dependent sensor selection. We empirically demonstrate the effectiveness of our algorithm on several real-world sensing tasks.

## Categories and Subject Descriptors

C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design; G.3 [**Probability and Statistics**]: Experimental Design; I.2.6 [**AI**]: Learning

## General Terms

Algorithms, Measurement

## Keywords

Sensor networks, approximation algorithms, distributed multiarmed bandit algorithms, submodular optimization

## 1. INTRODUCTION

A key challenge in deploying sensor networks for real-world applications such as environmental monitoring [19], building automation [25] and others is to decide when to activate the sensors in order to obtain the most useful information from the network (e.g., accurate predictions at unobserved locations) and to minimize power consumption. This sensor selection problem has received considerable attention [1, 32, 9], and algorithms with performance guarantees have been developed [1, 16]. However, many of the existing approaches make simplifying assumptions. Many approaches assume (1) that the sensors can perfectly observe a particular sensing region, and nothing outside the region [1]. This assumption does not allow us to model settings where multiple noisy sensors can help each other obtain better predictions. There are also approaches that base their notion of utility on more detailed models, such as improvement in prediction accuracy w.r.t. some statistical model [9] or detection performance [18]. However, most of these approaches make two crucial assumptions: (2) The model, upon which the optimization is based, is known in advance (e.g., based on domain knowledge or data from a pilot deployment) and (3), a centralized optimization selects the sensors (i.e., some centralized processor selects the sensors which obtain highest utility w.r.t. the model). We are not aware of any approach that simultaneously addresses the three main challenges (1), (2) and (3) above and still provides theoretical guarantees.

In this paper, we develop an efficient algorithm, called *Distributed Online Greedy* (DOG), which addresses these three central challenges. Prior work [17] has shown that many sensing tasks satisfy an intuitive diminishing returns property, submodularity, which states that activating a new sensor helps more if few sensors have been activated so far, and less if many sensors have already been activated. Our algorithm applies to any setting where the true objective is submodular [23], thus capturing a variety of realistic sensor models. Secondly, our algorithm does not require the model to be specified in advance: it learns to optimize the objective function in an online manner. Lastly, the algorithm is distributed; the sensors decide whether to activate themselves based on local information. We analyze our algorithm in the no-regret model, proving convergence properties similar to the best bounds for any centralized solution.

**A bandit approach toward sensor selection.** At the heart of our approach is a novel distributed algorithm for multi-armed bandit (MAB) problems. In the classical multiarmed bandit [24] setting, we picture a slot machine with multiple arms, where each arm generates a random payoff with unknown mean. Our goal is to devise a strategy for pulling arms to maximize the total reward accrued. The difference between the optimal arm's payoff and the obtained payoff is called the *regret*. Known algorithms can achieve average per-round regret of $\mathcal{O}(\sqrt{n \log n}/\sqrt{T})$ where $n$ is the number of arms, and $T$ the number of rounds (see e.g. the survey of [12]). Suppose we would like to, at every time step, select $k$ sensors. The sensor selection problem can then be cast as a multiarmed bandit problem, where there is one arm for each possible set of $k$ sensors, and the payoff is the accrued utility for the selected set. Since the number of possible sets, and thus the number of arms, is exponentially large, the resulting regret bound is $\mathcal{O}(n^{k/2}\sqrt{\log n}/\sqrt{T})$, i.e., exponential in $k$. However, when the utility function is submodular, the payoffs of these arms are correlated. Recent results [28] show that this correlation due to submodularity can be exploited by reducing the $n^k$-armed bandit problem to $k$ separate $n$-armed bandit problems, with only a bounded loss in performance. Existing bandit algorithms, such as the widely used EXP3 algorithm [2], are centralized in nature. Consequently, the key challenge in distributed online submodular sensing is how to devise a distributed bandit algorithm. In Sec. 4 and 5, we develop a distributed variant of EXP3 using novel algorithms to sample from and update a probability distribution in a distributed way. Roughly, we develop a scheme where each sensor maintains its own weight, and activates itself *independently from all other sensors* purely depending on this weight.

**Observation specific selection.** A shortcoming of centralized sensor selection is that the individual sensors' current measurements are not considered in the selection process. In many applications, obtaining sensor measurements is less costly than transmitting the measurements across the network. For example, cell phones used in participatory sensing [5] can inexpensively obtain measurements on a regular basis, but it is expensive to constantly communicate measurements over the network. In Sec. 6, we extend our distributed selection algorithm to activate sensors depending on their observations, and analyze the tradeoff between power consumption and the utility obtained under observation specific activation.

**Communication models.** We analyze our algorithms under two models of communication cost: In the *broadcast* model, each sensor can broadcast a message to all other sensors at unit cost. In the *star network* model, messages can only be between a sensor and the base station, and each message has unit cost. In Sec. 4 we formulate and analyze a distributed algorithm for sensor selection under the simpler broadcast model. Then, in Sec. 5 we show how the algorithm can be extended to the star network model.

**Our main contributions.**

- Distributed EXP3, a novel distributed implementation of the classic multiarmed bandit algorithm.
- Distributed Online Greedy (DOG) and LAZYDOG, novel algorithms for distributed online sensor selection, which apply to many settings, only requiring the utility function to be submodular.
- OD-DOG, an extension of DOG to allow for observation-dependent selection.
- We analyze our algorithm in the no-regret model and prove that it attains the optimal regret bounds attainable by any efficient centralized algorithm.
- We evaluate our approach on several real-world sensing tasks including monitoring a 12,527 node network.

Finally, while we do not consider multi-hop or general network topologies in this paper, we believe that the ideas behind our algorithms will likely prove valuable for sensor selection in those models as well.

## 2. THE SENSOR SELECTION PROBLEM

We now formalize the sensor selection problem. Suppose a network of sensors has been deployed at a set of locations $V$ with the task of monitoring some phenomenon (e.g., temperature in a building). Constraints on communication bandwidth or battery power typically require us to select a subset $\mathcal{A}$ of these sensors for activation, according to some utility function. The activated sensors then send their data to a server (base station). We first review the traditional offline setting where the utility function is specified in advance, illustrating how submodularity allows us to obtain provably near-optimal selections. We then address the more challenging setting where the utility function must be learned from data in an online manner.

### 2.1 The Offline Sensor Selection Problem

A standard offline sensor selection algorithm chooses a set of sensors that maximizes a known sensing quality objective function $f(\mathcal{A})$, subject to some constraints, e.g., on the number of activated sensors. One possible choice for the sensing quality is based on prediction accuracy (we will discuss other possible choices later on). In many applications, measurements are correlated across space, which allows us to make predictions at the unobserved locations. For example, prior work [9] has considered the setting where a random variable $\mathcal{X}_s$ is associated with every location $s \in V$, and a joint probability distribution $P(\mathcal{X}_V)$ models the correlation between sensor values. Here, $\mathcal{X}_V = [\mathcal{X}_1, \ldots, \mathcal{X}_n]$ is the random vector over all measurements. If some measurements $\mathcal{X}_\mathcal{A} = \mathbf{x}_\mathcal{A}$ are obtained at a subset of locations, then the conditional distribution $P(\mathcal{X}_{V\setminus\mathcal{A}} \mid \mathcal{X}_\mathcal{A} = \mathbf{x}_\mathcal{A})$ allows predictions at the unobserved locations, e.g., by predicting $\mathbb{E}[\mathcal{X}_{V\setminus\mathcal{A}} \mid \mathcal{X}_\mathcal{A} = \mathbf{x}_\mathcal{A}]$. Furthermore, this conditional distribution quantifies the *uncertainty* in the prediction: Intuitively, we would like to select sensors that minimize the predictive uncertainty. One way to quantify the predictive uncertainty is the mean squared prediction error,

$$\mathrm{MSE}(\mathcal{X}_{V\setminus\mathcal{A}} \mid \mathbf{x}_\mathcal{A}) = \frac{1}{n}\sum_{s\in V\setminus\mathcal{A}} \mathbb{E}[(\mathcal{X}_s - \mathbb{E}[\mathcal{X}_s \mid \mathbf{x}_\mathcal{A}])^2 \mid \mathbf{x}_\mathcal{A}].$$

In general, the measurements $\mathbf{x}_\mathcal{A}$ that sensors $\mathcal{A}$ will make is not known in advance. Thus, we can base our optimization on the *expected mean squared prediction error*,

$$\mathrm{EMSE}(\mathcal{A}) = \int dp(\mathbf{x}_\mathcal{A})\,\mathrm{MSE}(\mathcal{X}_{V\setminus\mathcal{A}} \mid \mathbf{x}_\mathcal{A}).$$

Equivalently, we can maximize the *reduction* in mean squared prediction error,

$$f_{\mathrm{EMSE}}(\mathcal{A}) = \mathrm{EMSE}(\emptyset) - \mathrm{EMSE}(\mathcal{A}).$$

By definition, $f_{\mathrm{EMSE}}(\emptyset) = 0$, i.e., no sensors obtain no utility. Furthermore, $f_{\mathrm{EMSE}}$ is monotonic: if $\mathcal{A} \subseteq \mathcal{B} \subseteq V$, then $f_{\mathrm{EMSE}}(\mathcal{A}) \leq f_{\mathrm{EMSE}}(\mathcal{B})$, i.e., adding more sensors always helps. That means, $f_{\mathrm{EMSE}}$ is maximized by the set of all sensors $V$. However, in practice, we would like to only select a small set of, e.g., at most $k$ sensors due to bandwidth and power constraints:

$$\mathcal{A}^* = \arg\max_{\mathcal{A}} f_{\mathrm{EMSE}}(\mathcal{A}) \text{ s.t. } |\mathcal{A}| \leq k.$$

Unfortunately, this optimization problem is NP-hard, so we cannot expect to efficiently find the optimal solution. Fortunately, it can be shown [8] that in many settings[1], the function $f_{\mathrm{EMSE}}$ satisfies an intuitive diminishing returns property called submodularity. A set function $f : 2^V \to \mathbb{R}$ is called *submodular* if, for all $\mathcal{A} \subseteq \mathcal{B} \subseteq V$ and $s \in V \setminus \mathcal{B}$ it holds that $f(\mathcal{A} \cup \{s\}) - f(\mathcal{A}) \geq f(\mathcal{B} \cup \{s\}) - f(\mathcal{B})$. Many other natural objective functions for sensor selection satisfy submodularity as well [17]. For example, the *sensing region* model where $f_{REG}(\mathcal{A})$ is the total area covered by all sensors $\mathcal{A}$ is submodular. The *detection* model where $f_{DET}(\mathcal{A})$ counts the expected number of targets detected by sensors $\mathcal{A}$ is submodular as well.

A fundamental result of Nemhauser et al. [23] is that for monotone submodular functions, a simple greedy algorithm, which starts with the empty set $\mathcal{A}_0 = \emptyset$ and iteratively adds the element

$$s_k = \arg\max_{s\in V\setminus\mathcal{A}_{k-1}} f(\mathcal{A}_{k-1} \cup \{s\});\;\; \mathcal{A}_k = \mathcal{A}_{k-1} \cup \{s_k\}$$

which maximally improves the utility obtains a near-optimal solution: For the set $\mathcal{A}_k$ it holds that

$$f(\mathcal{A}_k) \geq (1 - 1/e) \max_{|\mathcal{A}|\leq k} f(\mathcal{A}),$$

[1]For Gaussian models and conditional suppressorfreeness [8]

i.e., the greedy solution obtains at least a constant fraction of $(1 - 1/e) \approx 63\%$ of the optimal value.

One fundamental problem with this offline approach is that it requires the function $f$ to be specified in advance, i.e., before running the greedy algorithm. For the function $f_{\mathrm{EMSE}}$, this means that the probabilistic model $P(\mathcal{X}_V)$ needs to be known in advance. While for some applications some prior data, e.g., from pilot deployments, may be accessible, very often no such prior data is available. This leads to a "chicken-and-egg" problem, where sensors need to be activated to collect data in order to learn a model, but also the model is required to inform the sensor selection. This is akin to the "exploration–exploitation tradeoff" in reinforcement learning [2], where an agent needs to decide whether to explore and gather information about effectiveness of an action, or to exploit, i.e., choose actions known to be effective. In the following, we devise an online monitoring scheme based on this analogy.

## 2.2 The Online Sensor Selection Problem

We now consider the more challenging problem where the objective function is not specified in advance, and needs to be learned during the monitoring task. We assume that we intend to monitor the environment for a number $T$ of time steps (rounds). In each round $t$, a set $S_t$ of sensors is selected, and these sensors transmit their measurements to a server (base station). The server then determines a sensing quality $f_t(S_t)$ quantifying the utility obtained from the resulting analysis. For example, if our goal is spatial prediction, the server would build a model based on the previously collected sensor data, pick a random sensor $s$, make prediction for the variable $\mathcal{X}_s$, and then compare the prediction $\mu_s$ with the sensor reading $x_s$. The error $f_t = \sigma_s^2 - (\mu_s - x_s)^2$ is an unbiased estimate of the reduction in EMSE. In the following analysis, we will only assume that the objective functions $f_t$ are bounded (w.l.o.g., take values in $[0, 1]$), monotone, and submodular, and that we have some way of computing $f_t(S)$ for any subset of sensors $S$. Our goal is to maximize the total reward obtained by the system over $T$ rounds, $\sum_{t=1}^{T} f_t(S_t)$.

We seek to develop a protocol for selecting the sets $S_t$ of sensors at each round, such that after a small number of rounds the average performance of our online algorithm converges to the same performance of the offline strategy (that knows the objective functions). We thus compare our protocol against all strategies that can select a fixed set of $k$ sensors for use in all of the rounds; the best such strategy obtains reward $\max_{S\subseteq V:|S|\leq k} \sum_{t=1}^{T} f_t(S)$. The difference between this quantity and what our protocol obtains is known as its *regret*, and an algorithm is said to be *no-regret* if its average regret tends to zero (or less)[2] as $T \to \infty$.

When $k = 1$, our problem is simply the well-studied *multiarmed bandit* (MAB) problem, for which many no-regret

[2]Formally, if $R_T$ is the total regret for the first $T$ rounds, no-regret means $\limsup_{T\to\infty} R_T/T \leq 0$.

algorithms are known [12]. For general $k$, because the average of several submodular functions remains submodular, we can apply the result of Nemhauser *et al.* [23] (*cf.*, Sec. 2.1) to prove that a simple greedy algorithm obtains a $(1-1/e)$ approximation to the optimal offline solution. Feige [11] showed that this is optimal in the sense that obtaining a $(1-1/e+\epsilon)$ approximation for any $\epsilon > 0$ is NP-hard. These facts suggest that we cannot expect any efficient online algorithm to converge to a solution better than $(1-1/e)\max_{S\subseteq V:|S|\le k}\sum_{t=1}^{T} f_t(S)$. We therefore define the $(1-1/e)$-regret of a sequence of (possibly random) sets $\{S_t\}_{t=1}^{T}$ as

$$R_T := (1-1/e)\cdot \max_{S\subseteq V:|S|\le k}\sum_{t=1}^{T} f_t(S) \;-\; \sum_{t=1}^{T}\mathbb{E}\left[f_t(S_t)\right]$$

where the expectation is taken over the distribution for each $S_t$. We say an online algorithm producing a sequence of sets has *no-*$(1-1/e)$*-regret* if $\limsup_{T\to\infty}\frac{R_T}{T}\le 0$.

## 3. CENTRALIZED ALGORITHM FOR ONLINE SENSOR SELECTION

Before developing the distributed algorithm for online sensor selection, we will first review a centralized algorithm which is guaranteed to achieve no $(1-1/e)$-regret. In Sec. 4 we will show how this centralized algorithm can be implemented efficiently in a distributed manner. This algorithm starts with the greedy algorithm for a known submodular function mentioned in Sec. 2.1, and adapts it to the online setting. Doing so requires an online algorithm for selecting a *single* sensor as a subroutine, and we review such an algorithm in Sec. 3.1 before discussing the centralized algorithm for selecting multiple sensors in Sec. 3.2.

### 3.1 Centralized Online Single Sensor Selection

Let us first consider the case where $k=1$, i.e., we would like to select one sensor at each round. This simpler problem can be interpreted as an instance of the multiarmed bandit problem (as introduced in Sec. 2.2), where we have one arm for each possible sensor. In this case, the EXP3 algorithm [2] is a centralized solution for no-regret single sensor selection. EXP3 works as follows: It is parameterized by a *learning rate* $\eta$, and an *exploration probability* $\gamma$. It maintains a set of weights $w_s$, one for each arm (sensor) $s$, initialized to 1. At every round $t$, it will select each arm $s$ with probability

$$p_s = (1-\gamma)\frac{w_s}{\sum_{s'} w_{s'}} + \frac{\gamma}{n},$$

i.e., with probability $\gamma$ it explores, picking an arm uniformly at random, and with probability $(1-\gamma)$ it exploits, picking an arm $s$ with probability proportional to its weight $w_s$. Once an arm $s$ has been selected, a feedback $r = f_t(\{s\})$ is obtained, and the weight $w_s$ is updated to

$$w_s \leftarrow w_s \exp(\eta r/p_s).$$

Auer et al. [2] showed that with appropriately chosen learning rate $\eta$ and exploration probability $\gamma$ it holds that the cumulative regret $R_T$ of EXP3 is $\mathcal{O}(\sqrt{Tn\ln n})$, i.e., the average regret $R_T/T$ converges to zero.

### 3.2 Centralized Selection of Multiple Sensors

In principle, we could interpret the sensor selection problem as a $\binom{n}{k}$-armed bandit problem, and apply existing no-regret algorithms such as EXP3. Unfortunately, this approach does not scale, since the number of arms grows exponentially with $k$. However, in contrast to the traditional multiarmed bandit problem, where the arms are assumed to have independent payoffs, in the sensor selection case, the utility function is submodular and thus the payoffs are correlated across different sets. Recently, Streeter and Golovin showed how this submodularity can be exploited, and developed a no-$(1-1/e)$-regret algorithm for online maximization of submodular functions [28]. The key idea behind their algorithm, OG$_{\text{unit}}$, is to turn the offline greedy algorithm into an online algorithm by replacing the greedy selection of the element $s_k$ that maximizes the benefit $s_k = \arg\max_s f(\{s_1,...,s_{k-1}\}\cup\{s\})$ by a bandit algorithm. As shown in the pseudocode below, OG$_{\text{UNIT}}$ maintains $k$ bandit algorithms, one for each sensor to be selected. At each round $t$, it selects $k$ sensors according to the choices of the $k$ bandit algorithms $\mathcal{E}_i$ [3]. Once the elements have been selected, the $i^{\text{th}}$ bandit algorithm $\mathcal{E}_i$ receives as feedback the incremental benefit $f_t(s_1,\dots,s_i) - f_t(s_1,\dots,s_{i-1})$, i.e., how much additional utility is obtained by adding sensor $s_i$ to the set of already selected sensors. Below we define $[m] := \{1,2,\dots,m\}$.

```
Algorithm OG_UNIT from [28]:
Initialize $k$ multiarmed bandit algorithms $\mathcal{E}_1, \mathcal{E}_2, \dots, \mathcal{E}_k$,
each with action set $V$.
For each round $t \in [T]$
  For each stage $i \in [k]$ in parallel
    $\mathcal{E}_i$ selects an action $v_i^t$
  For each $i \in [k]$ in parallel
    feedback $f_t(\{v_j^t : j \le i\}) - f_t(\{v_j^t : j < i\})$ to $\mathcal{E}_i$.
  Output $S_t = \{a_1^t, a_2^t, \dots, a_k^t\}$.
```

In [27] it is shown that OG$_{\text{UNIT}}$ has a $\left(1-\frac{1}{e}\right)$-regret bound of $\mathcal{O}(kR)$ in this feedback model assuming each $\mathcal{E}_i$ has expected regret at most $R$. Thus, when using EXP3 as a subroutine, OG$_{\text{UNIT}}$ has no-$(1-1/e)$-regret.

Unfortunately, EXP3 (and in fact all MAB algorithms with no-regret guarantees for non-stochastic reward functions) require sampling from some distribution with weights associated with the sensors. If $n$ is small, we could simply store these weights on the server, and run the bandit algorithms $\mathcal{E}_i$ there. However, this solution does not scale to large numbers of sensors. Thus the key problem for online sensor selection is to develop a multiarmed bandit algorithm which implements

[3] Bandits with duplicate choices are handled in Sec. 4.6.1 of [28]

*distributed sampling* across the network, with minimal overhead of communication. In addition, the algorithm needs to be able to maintain the distributions (the weights) associated with each $\mathcal{E}_i$ in a distributed fashion.

## 4. DISTRIBUTED ALGORITHM FOR ONLINE SENSOR SELECTION

We will now develop DOG, an efficient algorithm for distributed online sensor selection. For now we make the following assumptions:

1. Each sensor $v \in V$ is able to compute its contribution to the utility $f_t(S \cup \{v\}) - f_t(S)$, where $S$ are a subset of sensors that have already been selected.
2. Each sensor can broadcast to all other sensors.
3. The sensors have calibrated clocks and unique, linearly ordered identifiers.

These assumptions are reasonable in many applications: (1) In target detection, for example, the objective function $f_t(S)$ counts the number of targets detected by the sensors $S$. Once previously selected sensors have broadcasted which targets they detected, the new sensor $s$ can determine how many additional targets have been detected. Similarly, in statistical estimation, one sensor (or a small number of sensors) randomly activates each round and broadcasts its value. After sensors $S$ have been selected and announced their measurements, the new sensor $s$ can then compute the improvement in prediction accuracy over the previously collected data. (2) The assumption that broadcasts are possible may be realistic for dense deployments and fairly long range transmissions. In Sec. 5 we will show how assumptions (1) and (2) can be relaxed.

As we have seen in Sec. 3, the key insight in developing a centralized algorithm for online selection is to replace the greedy selection of the sensor which maximally improves the total utility over the set of previously selected sensors by a bandit algorithm. Thus, a natural approach for developing a distributed algorithm for sensor selection is to first consider the single sensor case.

### 4.1 Distributed Selection of a Single Sensor

The key challenge in developing a distributed version of EXP3 is to find a way to sample exactly one element from a probability distribution $p$ over sensors in a distributed manner. This problem is distinct from randomized leader election [22], where the objective is to select exactly one element but the element need not be drawn from a specified distribution. We note that under the multi-hop communication model, sampling one element from the uniform distribution given a rooted spanning tree can be done via a simple random walk [20], but that under the broadcast and star network models this approach degenerates to centralized sampling. Our algorithm, in contrast, samples from an arbitrary distribution by allowing sensors to individually decide to activate. Our bottom-up approach also has two other advantages: (1) it is amenable to modification of the activation probabilities based on local observations, as we discuss in Sec. 6, and (2) since it does not rely on any global state of the network such as a spanning tree, it can gracefully cope with significant edge or node failures.

**A naive distributed sampling scheme.** A naive distributed algorithm would be to let each sensor keep track of all activation probabilities $p$. Then, one sensor (e.g., with the lowest identifier) would broadcast a single random number $u$ uniformly distributed in $[0,1]$, and the sensor $v$ for which $\sum_{i=1}^{v-1} p_i \leq u < \sum_{i=1}^{v} p_i$ would activate. However, for large sensor network deployments, this algorithm would require each sensor to store a large amount of global information (all activation probabilities $p$). Instead, each sensor $v$ could store only their own probability mass $p_v$; the sensors would then, in order of their identifiers, broadcast their probabilities $p_v$, and stop once the sum of the probabilities exceeds $u$. This approach only requires a constant amount of local information, but requires an impractical $\Theta(n)$ messages to be sent, and sent sequentially over $\Theta(n)$ time steps.

**Distributed multinomial sampling.** In this section we present a protocol that requires only $\mathcal{O}(1)$ messages in expectation, and only a constant amount of local information.

For a sampling procedure with input distribution $p$, we let $\hat{p}$ denote the resulting distribution, where in all cases at most one sensor is selected, and nothing is selected with probability $1 - \sum_v \hat{p}_v$. A simple approach towards distributed sampling would be to activate each sensor $v \in V$ *independently* from each other with probability $p_v$. While in expectation, exactly one sensor is activated, with probability $\prod_v (1 - p_v) > 0$ no sensor is activated; also since sensors are activated independently, there is a nonzero probability that more than one sensor is activated. Using a synchronized clock, the sensors could determine if no sensor is activated. In this case, they could simply repeat the selection procedure until at least one sensor is activated. One naive approach would be to repeat the selection procedure until exactly one sensor is activated. However with two sensors and $p_1 = \varepsilon, p_2 = 1 - \varepsilon$ this algorithm yields $\hat{p}_1 = \varepsilon^2/(1 - 2\varepsilon + 2\varepsilon^2) = \mathcal{O}(\varepsilon^2)$, so the first sensor is severely underrepresented. Another simple protocol would be to select exactly one sensor uniformly at random from the set of activated sensors, which can be implemented using few messages.

```
The Simple Protocol:
 For each sensor $v$ in parallel
     Sample $X_v \sim \mathrm{Bernoulli}(p_v)$.
     If $(X_v = 1)$, $X_v$ activates.
 All active sensors $S$ coordinate to select a single sensor uniformly at random from $S$, e.g., by electing the minimum ID sensor in $S$ to do the sampling.
```

It is not hard to show that with this protocol, for all sensors $v$,

$$\hat{p}_v = p_v \cdot \mathbb{E}\left[\frac{1}{|S|} \,\middle|\, v \in S\right] \geq p_v/\mathbb{E}\left[|S| \mid v \in S\right] \geq p_v/2$$

by appealing to Jensen's inequality. Since $\hat{p}_v \leq p_v$, we find that this simple protocol maintains a ratio $r_v := \hat{p}_v/p_v \in [\frac{1}{2}, 1]$. Unfortunately, this analysis is tight, as can be seen from the example with two sensors and $p_1 = \varepsilon, p_2 = 1 - \varepsilon$.

To improve upon the simple protocol, first consider running it on an example with $p_1 = p_2 = \cdots = p_n = 1/n$. Since the protocol behaves exactly the same under permutations of sensor labels, by symmetry we have $\hat{p}_1 = \hat{p}_2 = \cdots = \hat{p}_n$, and thus $r_i = r_j$ for all $i, j$. Now consider an input distribution $p$ where there exists integers $N$ and $k_1, k_2, \ldots, k_n$ such that $p_v = k_v/N$ for all $v$. Replace each $v$ with $k_v$ fictitious sensors, each with probability mass $1/N$, and each with a label indicating $v$. Run the simple protocol with the fictitious sensors, selecting a fictitious sensor $v'$, and then actually select the sensor indicated by the label of $v'$. By symmetry this process selects each fictitious sensor with probability $(1 - \beta)/N$, where $\beta$ is the probability that nothing at all is selected, and thus the process selects sensor $v$ with probability $k_v(1-\beta)/N = (1-\beta)p_v$ (since at most one fictitious sensor is ever selected).

We may thus consider the following improved protocol which incorporates the above idea, simulating this modification to the protocol exactly when $p_v = k_v/N$ for all $v$.

```
The Improved Protocol($N$):
 For each sensor $v$ in parallel
     Sample $X_v \sim$ Binomial($\lceil N \cdot p_v \rceil$, $1/N$).
     If ($X_v \geq 1$), then activate sensor $v$.
 From the active sensors $S$, select sensor $v$ with probability $X_v / \sum_{v' \in S} X_{v'}$.
```

This protocol ensures the ratios $r_v := \hat{p}_v/p_v$ are the same for all sensors, provided each $p_v$ is a multiple of $1/N$. Assuming the probabilities are rational, there will be a sufficiently large $N$ to satisfy this condition. To reduce $\beta := \mathbf{Pr}\left[S = \emptyset\right]$ in the simple protocol, we may sample each $X_v$ from $\text{Bernoulli}(\alpha \cdot p_v)$ for any $\alpha \in [1, n]$. The symmetry argument remains unchanged. This in turn suggests sampling $X_v$ from $\text{Binomial}(\lceil N \cdot p_v \rceil, \alpha/N)$ in the improved protocol. Taking the limit as $N \to \infty$, the binomial distribution becomes Poisson, and we obtain the desired protocol.

```
The Poisson Multinomial Sampling (PMS) Protocol($\alpha$):
 Same as the improved protocol, except each
 sensor $v$ samples $X_v \sim$ Poisson($\alpha p_v$)
```

Straight-forward calculation shows that

$$\mathbf{Pr}\left[S = \emptyset\right] = \prod_v \exp\left\{-\alpha \cdot p_v\right\} = \exp\Big\{-\sum_v \alpha \cdot p_v\Big\} = e^{-\alpha}$$

Let $C$ be the number of messages. Then

$$\mathbb{E}\left[C\right] = \sum_v \mathbf{Pr}\left[X_v \geq 1\right] = \sum_v (1 - e^{-\alpha p_v}) \leq \sum_v \alpha p_v = \alpha$$

Here we have used linearity of expectation, and $1 + x \leq e^x$ for all $x \in \mathbb{R}$. In summary, we have the following result about our protocol:

PROPOSITION 1. *Fix any fixed $p$ and $\alpha > 0$. The PMS Protocol always selects at most one sensor, ensures*

$$\forall v: \ \mathbf{Pr}\left[v \textit{ selected}\right] = (1 - e^{-\alpha})p_v$$

*and requires no more than $\alpha$ messages in expectation.*

In order to ensure that exactly one sensor is selected, whenever $S = \emptyset$ we can simply rerun the protocol with fresh random seeds as many times as needed until $S$ is non-empty. Using $\alpha = 1$, this modification will require only $\mathcal{O}(1)$ messages in expectation and at most $\mathcal{O}(\log n)$ messages with high probability in the broadcast model. We can combine this protocol with EXP3 to get the following result.

THEOREM 2. *In the broadcast model, running* EXP3 *using the PMS Protocol with $\alpha = 1$, and rerunning the protocol whenever nothing is selected, yields exactly the same regret bound as standard* EXP3*, and in each round at most $e/(e-1) + 2 \approx 3.582$ messages are broadcast in expectation.*

The regret bound for EXP3 is $\mathcal{O}(\sqrt{\mathsf{OPT} n \log n})$, where $\mathsf{OPT}$ is the total reward of the best action. Our variant simulates EXP3, and thus has identical regret. Proofs of our theoretical results can be found in the full version of this paper [13].

**Remark.** Running our variant of EXP3 requires that each sensor know the number of sensors, $n$, in order to compute its activation probability. If each sensor $v$ has only a reasonable estimate of $n_v$ of $n$, however, our algorithm still performs well. For example, it is possible to prove that if all of the sensors have the same estimate $n_v = cn$ for some constant $c > 0$, then the upper bound on expected regret, $R(c)$, grows as $R(c) \approx R(1) \cdot \max\{c, 1/c\}$. The expected number of activations in this case increases by at most $\left(\frac{1}{c} - 1\right)\gamma$. In general underestimating $n$ leads to more activations, and underestimating or overestimating $n$ can lead to more regret. This graceful degradation of performance with respect to the error in estimating $n$ holds for all of our algorithms.

## 4.2 The Distributed Online Greedy Algorithm

We now use our single sensor selection algorithm to develop our main algorithm, the Distributed Online Greedy algorithm (DOG). It is based on the distributed implementation of EXP3 using the PMS Protocol. Suppose we would like to select $k$ sensors at each round $t$. Each sensor $v$ maintains $k$ weights $w_{v,1}, \ldots, w_{v,k}$ and normalizing constants $Z_{v,1}, \ldots, Z_{v,k}$. The algorithm proceeds in $k$ stages, synchronized using the common clock. In stage $i$, a single sensor is selected using the PMS Protocol applied to the distribution $(1-\gamma)w_{v,i}/Z_{v,i}+\gamma/n$. Suppose sensors $S = \{v_1, \ldots, v_{i-1}\}$ have been selected in stages 1 through $i-1$. The sensor $v$ selected at stage $i$ then computes its local rewards $\pi_{v,i}$ using the utility function $f_t(S \cup \{v_i\}) - f_t(S)$. It then computes its new weight

$$w'_{v,i} = w_{v,i} \exp(\eta \pi_{v,i}/p_{v,i}),$$

and broadcasts the difference between its new and old weights $\Delta_{v,i} = w'_{v,i} - w_{v,i}$. All sensors then update their $i^{\text{th}}$ normalizers using $Z_{v,i} \leftarrow Z_{v,i} + \Delta_{v,i}$. Fig. 1 presents the pseudo-code of the DOG algorithm. Thus given Theorem 12 of [27] we have the following result about the DOG algorithm:

THEOREM 3. *The* DOG *algorithm selects, at each round $t$ a set $S_t \subseteq V$ of $k$ sensors such that*

$$\frac{1}{T}\mathbb{E}\left[\sum_{t=1}^{T} f_t(S_t)\right] \geq \frac{1-\frac{1}{e}}{T} \max_{|S|\leq k} \sum_{t=1}^{T} f_t(S) - O\left(k\sqrt{\frac{n \log n}{T}}\right).$$

*In expectation, only $\mathcal{O}(k)$ messages are exchanged each round.*

## 5. THE STAR NETWORK MODEL

In some applications, the assumption that sensors can broadcast messages to all sensors may be unrealistic. Furthermore, in some applications sensors may not be able to compute the marginal benefits $f_t(S \cup \{s\}) - f_t(S)$ (since this calculation may be computationally complex). In this section, we analyze LAZYDOG, a variant of our DOG algorithm, which replace the above assumptions by the assumption that there is a dedicated base station[4] available which computes utilities and which can send non-broadcast messages to individual sensors.

We make the following assumptions:

1. Every sensor stores its probability mass $p_v$ with it, and can only send messages to and receive messages from the base station.
2. The base station is able, after receiving messages from a set $S$ of sensors, to compute the utility $f_t(S)$ and send this utility back to the active sensors.

These conditions arise, for example, when cell phones in participatory sensor networks can contact the base station, but due to privacy constraints cannot directly call other phones. We do not assume that the base station has access to all weights of the sensors – we will only require the base station to have $\mathcal{O}(k + \log n)$ memory. In the fully distributed algorithm DOG that relies on broadcasts, it is easy for the sensors to maintain their normalizers $Z_{v,i}$, since they receive information about rewards from all selected sensors. The key challenge when removing the broadcast assumption is to maintain the normalizers in an appropriate manner.

### 5.1 Lazy renormalization & Distributed EXP3

EXP3 (and all MAB with no-regret guarantees against arbitrary reward functions) must maintain a distribution over actions, and update this distribution in response to feedback about the environment. In EXP3, each sensor $v$ requires only $w_v(t)$ and a normalizer $Z(t) := \sum_{v'} w_{v'}(t)$ to compute $p_v(t)$[5]. The former changes only when $v$ is selected. In the broadcast model the latter can simply be broadcast at the end of each round. In the star network model (or, more generally in multi-hop models), standard flooding echo aggregation techniques could be used to compute and distribute the new normalizer, though with high communication cost. We show that a lazy renormalization scheme can significantly reduce the amount of communication needed by a distributed bandit algorithm *without altering its regret bounds whatsoever*. Thus our lazy scheme is complementary to standard aggregation techniques.

Our lazy renormalization scheme for EXP3 works as follows. Each sensor $v$ maintains its weight $w_v(t)$ and an estimate $Z_v(t)$ for $Z(t) := \sum_{v'} w_{v'}(t)$, Initially, $w_v(0) = 1$ and $Z_v(0) = n$ for all $v$. The central server stores $Z(t)$. Let

$$\rho(x, y) := (1-\gamma)\frac{x}{y} + \frac{\gamma}{n}.$$

Each sensor then proceeds to activate as in the sampling procedure of Sec. 4.1 as if its probability mass in round $t$ were $q_v = \rho(w_v(t), Z_v(t))$ instead of its true value of $\rho(w_v(t), Z(t))$. A single sensor is selected by the server with respect to the true value $Z(t)$, resulting in a selection from the desired distribution. Moreover, $v$'s estimate $Z_v(t)$ is only updated on rounds when it communicates with the server under these circumstances. This allows the estimated probabilities of all of the sensors to sum to more than one, but has the benefit of significantly reducing the communication cost in the star network model under certain assumptions. We call the result *Distributed EXP3*, give its pseudocode for round $t$ in Fig. 2.

Since the sensors underestimate their normalizers, they may activate more frequently than in the broadcast model. Fortunately, the amount of "overactivation" remains bounded. We prove Theorem 4 and Corollary 5 in the full version of this paper [13].

THEOREM 4. *The number of sensor activations in any round of the Distributed* EXP3 *algorithm is at most $\alpha + (e-1)$ in expectation and $\mathcal{O}(\alpha + \log n)$ with high probability, and the number of messages is at most twice the number of activations.*

Unfortunately, there is still an $e^{-\alpha}$ probability of nothing

[4]Though the existence of such a base station means the protocol is not completely distributed, it is realistic in sensor network applications where the sensor data needs to be accumulated somewhere for analysis.

[5]We let $x(t)$ denote the value of variable $x$ at the start of round $t$, to ease analysis. We do not actually need to store the historical values of the variables over multiple time steps.

```
**Algorithm:** Distributed Online Greedy (DOG)   (described in the broadcast model)
**Input**: $k \in \mathbb{N}$, a set $V$, and $\alpha, \gamma, \eta \in \mathbb{R}_{>0}$. Reasonable defaults are any $\alpha \in [1, \ln |V|]$, and $\gamma = \eta =$
    $\min\left(1, (|V| \ln |V| / g)^{1/2}\right)$, where $g$ is a guess for the maximum cumulative reward of any single sensor [2].
Initialize $w_{v,i} \leftarrow 1$ and $Z_{v,i} \leftarrow |V|$ for all $v \in V$, $i \in [k]$. Let $\rho(x, y) := (1-\gamma)\frac{x}{y} + \frac{\gamma}{|V|}$.
**for each** round $t = 1, 2, 3, \ldots$ **do**
    Initialize $S_{v,t} \leftarrow \emptyset$ for each $v$ in parallel.
    **for each** stage $i \in [k]$ **do**
        **for each** sensor $v \in V$ in parallel **do**
            **repeat**
                Sample $X_v \sim \mathrm{Poisson}(\alpha \cdot \rho(w_{v,i}, Z_{v,i}))$.
                **if** $(X_v \geq 1)$ **then**
                    Broadcast $\langle$sampled $X_v, \mathrm{id}(v)\rangle$; Receive messages from sensors $S$. (Include $v \in S$ for convenience).
                    **if** $\mathrm{id}(v) = \min_{v' \in S} \mathrm{id}(v')$ **then**
                        Select exactly one element $v_{it}$ from $S$ such that each $v'$ is selected with probability $X_{v'} / \sum_{u \in S} X_u$.
                        Broadcast $\langle$select $\mathrm{id}(v_{it})\rangle$.
                    Receive message $\langle$select $\mathrm{id}(v_{it})\rangle$.
                    **if** $\mathrm{id}(v) = \mathrm{id}(v_{it})$ **then**
                        Observe $f_t(S_{v,t} + v)$; $\pi \leftarrow f_t(S_{v,t} + v) - f_t(S_{v,t})$; $\Delta_v \leftarrow w_{v,i}(\exp\{\eta \cdot \pi / \rho(w_{v,i}, Z_{v,i})\} - 1)$;
                        $Z_{v,i} \leftarrow Z_{v,i} + \Delta_v$; $w_v \leftarrow w_v + \Delta_v$; Broadcast $\langle$weight update $\Delta_v, \mathrm{id}(v)\rangle$.
                **if** receive message $\langle$weight update $\Delta, \mathrm{id}(v_{it})\rangle$ **then** $S_{v,t} \leftarrow S_{v,t} \cup \{v_{it}\}$; $Z_{v,i} \leftarrow Z_{v,i} + \Delta$;
            **until** $v$ receives a message of type $\langle$select id$\rangle$ ;
**Output**: At the end of each round $t$ each sensor has an identical local copy $S_{v,t}$ of the selected set $S_t$.
```

**Figure 1: The Distributed Online Greedy Algorithm**

being selected. To address this, we can set $\alpha = c \ln n$ for some $c \geq 1$, and if nothing is selected, transmit a message to each of the $n$ sensors to rerun the protocol.

COROLLARY 5. *There is a distributed implementation of* EXP3 *that always selects a sensor in each round, has the same regret bounds as standard* EXP3*, ensures that the number of sensor activations in any round is at most* $\ln n + \mathcal{O}(1)$ *in expectation or* $\mathcal{O}(\log n)$ *with high probability, and in which the number of messages is at most twice the number of activations.*

### 5.2 LazyDOG

Once we have the distributed EXP3 variant described above, we can use it for the bandit subroutines in the OG$_{\text{UNIT}}$ algorithm (*cf.* Sec. 3.2). We call the result the LAZYDOG algorithm, due to its use of lazy renormalization. The lazy distributed EXP3 still samples sensors from the same distribution as the regular distributed EXP3, so LAZYDOG has precisely the same performance guarantees with respect to $\sum_t f_t(S_t)$ as DOG. It works in the star network communication model, and requires few messages or sensor activations. Corollary 5 immediately implies the following result.

COROLLARY 6. *The number of sensors that activate each round in* LAZYDOG *is at most* $k \ln n + \mathcal{O}(k)$ *in expectation and* $\mathcal{O}(k \log n)$ *with high probability, the number of messages is at most twice the number of activations, and the* $(1 - 1/e)$*-regret of* LAZYDOG *is the same as* DOG.

## 6. OBSERVATION-DEPENDENT SAMPLING

Theorem 3 states that DOG is guaranteed to do nearly as well as the offline greedy algorithm run on an instance with objective function $f_\Sigma := \sum_t f_t$. Thus the reward of DOG is asymptotically near-optimal on average. In many applications, however, we would like to perform well on rounds with "atypical" objective functions. For example, in an outbreak detection application as we discuss in Sec. 7, we would like to get very good data on rounds with significant events, even if the nearest sensors typically report "boring" readings that contribute very little to the objective function. For now, suppose that we are only running a single MAB instance to select a single sensor in each round. If we have access to a black-box for evaluating $f_t$ on round $t$, then we can perform well on atypical rounds at the cost of some additional communication by having each sensor $v$ take a local reading of its environment and estimate its payoff $\bar{\pi} = f_t(\{v\})$ if selected. This value, which serves as a measure of how interesting its input is, can then be used to decide whether to boost $v$'s probability for reporting its sensor reading to the server. In the simplest case, we can imagine that each $v$ has a threshold $\tau_v$ such that $v$ activates with probability 1 if $\bar{\pi} \geq \tau_v$, and with its normal probability otherwise. In the case where we select $k > 1$ sensors in each round, each sensor can have a threshold for each of the $k$ stages, where in each stage it computes $\bar{\pi} = f_t(S \cup \{v\}) - f_t(S)$ where $S$ is the set of currently selected sensors. Since the activation probability only goes up, we can retain the performance guarantees of

```
Algorithm: Distributed EXP3 (executing on round $t$)
Input: Parameters $\alpha, \eta, \gamma \in \mathbb{R}_{>0}$, sensor set $V$.
Let $\rho(x, y) := (1-\gamma)\frac{x}{y} + \frac{\gamma}{|V|}$.
Sensors:
foreach sensor $v$ in parallel do
    Sample $r_v$ uniformly at random from $[0, 1]$.
    if ($r_v \geq 1 - \alpha \cdot \rho(w_v(t), Z_v(t))$) then
        Send $\langle r_v, w_v(t) \rangle$ to the server.
        Receive message $\langle Z, w \rangle$ from server.
        $Z_v(t+1) \leftarrow Z$; $w_v(t+1) \leftarrow w$.
    else $Z_v(t+1) \leftarrow Z_v(t)$; $w_v(t+1) \leftarrow w_v(t)$.
Server:
Receive messages from a set $S$ of sensors.
if $S = \emptyset$ then Select nothing and wait for next round.
else foreach sensor $v \in S$ do
    $Y_v \leftarrow \min\{x : \mathbf{Pr}[X \leq x] \geq r_v\}$, where
    $X \sim \text{Poisson}(\alpha \cdot \rho(w_v(t), Z(t)))$.
    Select $v$ with probability $Y_v / \sum_{v' \in S} Y_{v'}$.
    Observe the payoff $\pi$ for the selected sensor $v^*$;
    $w_{v^*}(t+1) \leftarrow w_{v^*}(t) \cdot \exp\{\eta\pi/\rho(w_{v^*}(t), Z(t))\}$;
    $Z(t+1) \leftarrow Z(t) + w_{v^*}(t+1) - w_{v^*}(t)$;
    for each $v \in S \setminus v^*$ do $w_v(t+1) \leftarrow w_v(t)$;
    for each $v \in S$ do Send $\langle Z(t+1), w_v(t+1) \rangle$ to $v$.
```

**Figure 2: Distributed EXP3: the PMS Protocol($\alpha$) with lazy renormalization, applied to EXP3**

DOG if we are careful to adjust the feedback properly.

Ideally, we wish that the sensors learn what their thresholds $\tau_v$ should be. We treat the selection of $\tau_v$ in each round as an online decision problem that each $v$ must play. We construct a particular game that the sensors play, where the strategies are the thresholds (suitably discretized), there is an *activation cost* $c_v$ that $v$ pays if $\bar{\pi}_v \geq \tau_v$, and the payoffs are defined as follows: Let $\pi_v = f_t(S \cup \{v\}) - f_t(S)$ be the marginal benefit of selecting $v$ given that sensor set $S$ has already been selected. Let $A$ be the set of sensors that activate in the current iteration of the game, and let $\max\left(\pi_{(A \setminus v)}\right) := \max\left(\pi_{v'} : v' \in A \setminus \{v\}\right)$. The particular reward function $\psi_v$ we choose for each sensor $v$ for each iteration of the game is

$$\psi_v(\tau) = \begin{cases} c_v - \max\left(\pi_v - \max\left(\pi_{(A \setminus v)}\right), 0\right) & \text{if } \bar{\pi} < \tau \\ \max\left(\pi_v - \max\left(\pi_{(A \setminus v)}\right), 0\right) - c_v & \text{if } \bar{\pi} \geq \tau \end{cases}$$

based on empirical performance. Thus, if a sensor activates ($\bar{\pi} \geq \tau$), its payoff is the improvement over the best payoff $\pi_{v'}$ among all sensors $v' \in A$ minus its activation cost. In case multiple sensors activate, the highest reward is retained.

In the broadcast model where each sensor can compute its marginal benefit, we can use any standard no-regret algorithm for combining expert advice, such as *Randomized Weighted Majority* (WMR) [21], to play this game and obtain no regret guarantees[6] for selecting $\tau_v$. In our context a sensor using WMR simply maintains weights $w(\tau_i) = \exp\left(\eta \cdot \psi_{\text{total}}(\tau_i)\right)$ for each possible threshold $\tau_i$, where $\eta > 0$ is a learning parameter, and $\psi_{\text{total}}(\tau_i)$ is the total cumulative reward for playing $\tau_i$ in every round so far. On each step each threshold is picked with probability proportional to its weight. In the more restricted star network model, we can use a modification of WMR that feeds back unbiased estimates for $\psi_t(\tau_i)$, the payoff to the sensor for using a threshold of $\tau_i$ in round $t$, and thus obtains reasonably good estimates of $\psi_{\text{total}}(\tau_i)$ after many rounds. We give pseudocode in Fig. 3. In it, we assume that an activated sensor can compute the reward of playing any threshold.

[6]We leave it as an open problem to determine if the outcome is close to optimal when all sensors play low regret strategies (i.e., is the *price of total anarchy* [4] small in any variant of this game with a reasonable way of splitting the value from the information?)

```
Algorithm: Modified WMR (star network setting)
Input: parameter $\eta > 0$, threshold set $\{\tau_i : i \in [m]\}$
Initialize $w(\tau_i) \leftarrow 1$ for all $i \in [m]$.
for each round $t = 1, 2, \ldots$ do
    Select $\tau_i$ with probability $w(\tau_i) / \sum_{j=1}^{m} w(\tau_j)$.
    if sensor activates then
        Let $\psi(\tau_i)$ be the reward for playing $\tau_i$ in this
        round of the game. Let $q(\tau_i)$ be the total
        probability of activation conditioned on $\tau_i$ being
        selected (including the activation probability that
        does not depend on local observations.)
        for each threshold $\tau_i$ do
            $w(\tau_i) \leftarrow w(\tau_i) \exp\left(\eta\psi(\tau_i)/q(\tau_i)\right)$.
```

**Figure 3: Selecting activation thresholds for a sensor**

We incorporate these ideas into the DOG algorithm, to obtain what we call the *Observation-Dependent Distributed Online Greedy* algorithm (OD-DOG). In the extreme case that $c_v = 0$ for all $v$ the sensors will soon set their thresholds so low that each sensor activates in each round. In this case OD-DOG will exactly simulate the offline greedy algorithm run on each round. In other words, if we let $G(f)$ be the result of running the offline greedy algorithm on the problem

$$\arg\max\{f(S) : S \subset V,\ |S| \leq k\}$$

then OD-DOG will obtain a value of $\sum_t f_t(G(f_t))$; in contrast, DOG gets roughly $\sum_t f_t(G(\sum_t f_t))$, which may be significantly smaller. Note that Feige's result [11] implies that the former value is the best we can hope for from efficient algorithms (assuming P $\neq$ NP). Of course, querying each sensor in each round is impractical when querying sensors is expensive. In the other extreme case where $c_v = \infty$ for all $v$, OD-DOG will simulate DOG after a brief learning phase. In general, by adjusting the activation costs $c_v$ we can smoothly trade off the cost of sensor communication with the value of the resulting data.

## 7. EXPERIMENTS

In this section, we evaluate our DOG algorithm on several real-world sensing problems.

### 7.1 Data sets

**Temperature data.** In our first data set, we analyze temperature measurements from the network of 46 sensors deployed at Intel Research Berkeley. Our training data consisted of samples collected at 30 second intervals on 3 consecutive days (starting Feb. 28th 2004), the testing data consisted of the corresponding samples on the two following days. The objective functions used for this application are based on the expected reduction in mean squared prediction error $f_{\mathrm{EMSE}}$, as introduced in Sec. 2.

**Precipitation data.** Our second data set consists of precipitation data collected during the years 1949 - 1994 in the states of Washington and Oregon [30]. Overall 167 regions of equal area, approximately 50 km apart, reported the daily precipitation. To ensure the data could be reasonably modeled using a Gaussian process we applied preprocessing as described in [19]. As objective functions we again use the expected reduction in mean squared prediction error $f_{\mathrm{EMSE}}$.

**Water network monitoring.** Our third data set is based on the application of monitoring for outbreak detection. Consider a city water distribution network for delivering drinking water to households. Accidental or malicious intrusions can cause contaminants to spread over the network, and we want to install sensors to detect these contaminations as quickly as possible. In August 2006, the Battle of Water Sensor Networks (BWSN) [10] was organized as an international challenge to find the best sensor placements for a real metropolitan water distribution network, consisting of 12,527 nodes. In this challenge, a set of intrusion scenarios is specified, and for each scenario a realistic simulator provided by the EPA is used to simulate the spread of the contaminant for a 48 hour period. An intrusion is considered detected when one selected node shows positive contaminant concentration. The goal of BWSN was to minimize impact measures, such as the expected population affected, which is calculated using a realistic disease model. For a security-critical sensing task such as protecting drinking water from contamination, it is important to develop sensor selection schemes that maximize detection performance even in adversarial environments (i.e., where an adversary picks the contamination strategy knowing our network deployment and selection algorithm). The algorithms developed in this paper apply to such adversarial settings. We reproduce the experimental setup detailed in [18]. For each contamination event $i$, we define a separate submodular objective function $f_i(S)$ that measures the expected population protected when detecting the contamination from sensors $S$. In [18], Krause et al. showed that the functions $f_i(A)$ are monotone submodular functions.

### 7.2 Convergence experiments

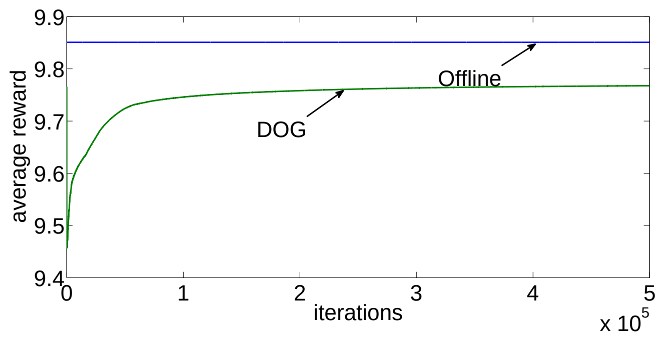
In our first set of experiments, we analyzed the convergence of our DOG algorithm. For both the temperature [T] and precipitation [R] data sets, we first run the offline greedy algorithm using the $f_{\mathrm{EMSE}}$ objective function to pick $k = 5$ sensors. We compare its performance to the DOG algorithm, where we feed back the same objective function at every round. We use an exploration probability $\gamma = 0.01$ and a learning rate inversely proportional to the maximum achievable reward $f_{\mathrm{EMSE}}(V)$. Fig. 4(a) presents the results for the temperature data set. Note that even after only a small number of rounds ($\approx 100$), the algorithm obtains 95% of the performance of the offline algorithm. After about 13,000 iterations, the algorithm obtains 99% of the offline performance, which is the best that can be expected with a .01 exploration probability. Fig. 4(b) show the same experiment on the precipitation data set. In this more complex problem, after 100 iterations, 76% of the offline performance is obtained, which increases to 87% after 500,000 iterations.

### 7.3 Observation dependent activation

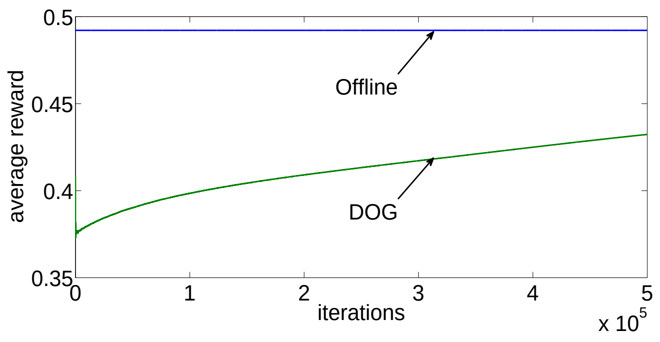
We also experimentally evaluate our OD-DOG algorithm with observation specific sensor activations. We choose different values for the activation cost $c_v$, which we vary as multiples of the total achievable reward. The activation cost $c_v$ lets us smoothly trade off the average number of sensors activating each round and the average obtained reward. The resulting activation strategies are used to select a subset of size $k = 10$ from a collection of 12,527 sensors. Fig. 4(c) presents rates of convergence using the OD-DOG algorithm under a fixed objective function which considers all contamination events. In Fig. 4(d), convergence rates are presented under a varying objective function, which selects a different contamination event on each round. For low activation costs, the performance quickly converges to or exceeds the performance of the offline solution. Even under the lowest activation costs in our experiments, the average number of extra activations per stage in the OD-DOG algorithm is at most 5. These results indicate that observation specific activation can lead to drastically improved performance at small additional activation cost.

## 8. RELATED WORK

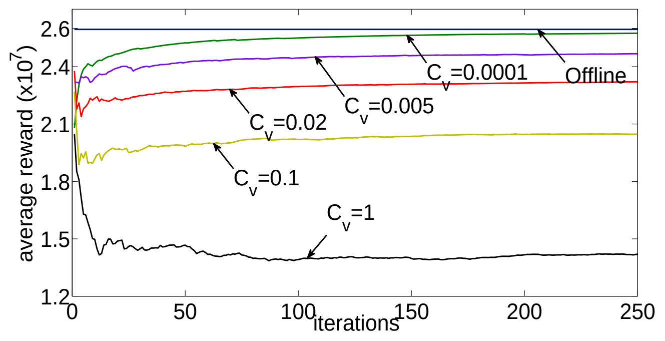
**Sensor Selection.** The problem of deciding when to selectively turn on sensors in sensor networks in order to conserve power was first discussed by [26] and [32]. Many approaches for optimizing sensor placements and selection assume that sensors have a fixed region [15, 14, 3]. These regions are usually convex or even circular. Further, it is assumed that everything within a region can be perfectly observed, and everything outside cannot be measured by the sensors. For complex applications such as environmental monitoring, these assumptions are unrealistic, and the direct optimization of prediction



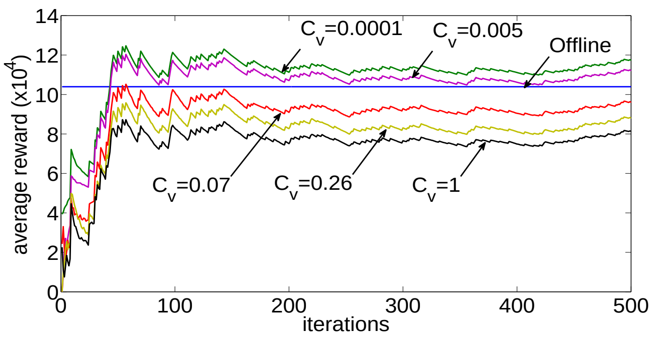
(a) *[T] Convergence*



(b) *[R] Convergence*



(c) *[W] Constant Objective*



(d) *[W] Varying Objective*

**Figure 4: Experimental results on [T] Temperature data, [R] precipitation data and [W] water distribution network data.**

accuracy is desired. The problem of selecting observations for monitoring spatial phenomena has been investigated extensively in geostatistics [7], and more generally (Bayesian) experimental design [6]. Several approaches have been proposed to activate sensors in order to minimize uncertainty [32] or prediction error [9]. However, these approaches do not have performance guarantees. Submodularity has been used to analyze algorithms for placing [19] or selecting [31] a fixed set of sensors. These approaches however assume that the model is known in advance.

**Submodular optimization.** The problem of centralized maximization of a submodular function has been studied by [23], who proved that the greedy algorithm gives a factor $(1-1/e)$ approximation. Several algorithms have since been developed for maximizing submodular functions under more complex constraints (see [29] for an overview). Streeter and Golovin developed an algorithm for online optimization of submodular functions, which we build on in this paper [28].

## 9. CONCLUSIONS

In this paper, we considered the problem of repeatedly selecting subsets $S_t$ from a large set of deployed sensors, in order to maximize a sequence of submodular utility functions $f_1, \dots, f_T$. We developed an efficient Distributed Online Greedy algorithm DOG, and proved it suffers no $(1-1/e)$-regret, essentially the best possible performance obtainable unless P = NP. Our algorithm is fully distributed, requiring only a small number of messages to be exchanged at each round with high probability. We analyze our algorithm both in the broadcast model, and in the star network model, where a separate base station is responsible for computing utilities of selected sets of sensors. Our LAZYDOG algorithm for the latter model uses lazy renormalization in order to reduce the number of messages required from $\Theta(n)$ to $\mathcal{O}(k \log n)$, and the server memory required from $\Theta(n)$ to $\mathcal{O}(k + \log n)$, where $k$ is the desired number of sensors to be selected. In addition, we developed OD-DOG, an extension of DOG that allows observation-dependent sensor selection. We empirically demonstrate the effectiveness of our algorithms on three real-world sensing tasks, demonstrating how our DOG algorithm's performance converges towards the performance of a clairvoyant offline greedy algorithm. In addition, our results with the OD-DOG algorithm indicate that a small number of extra sensor activations can lead to drastically improved convergence. We believe that our results provide an interesting step towards a principled study of distributed active learning and information gathering.

**Acknowledgments.** The authors wish to thank Phillip Gibbons and the anonymous referees for their valuable help and suggestions. This research was partially supported by ONR grant N00014-09-1-1044, NSF grant CNS-0932392, a gift from Microsoft Corporation and the Caltech Center for the Mathematics of Information.

## 10. REFERENCES

[1] Z. Abrams, A. Goel, and S. Plotkin. Set $k$-cover algorithms for energy efficient monitoring in wireless sensor networks. In *IPSN*, pages 424–432, 2004.

[2] P. Auer, N. Cesa-Bianchi, Y. Freund, and R. E. Schapire. The nonstochastic multiarmed bandit problem. *SIAM Journal on Computing*, 32(1):48–77, 2002.

[3] X. Bai, S. Kumar, Z. Yun, D. Xuan, and T. H. Lai. Deploying wireless sensors to achieve both coverage and connectivity. In *ACM MobiHoc*, 2006.

[4] A. Blum, M. Hajiaghayi, K. Ligett, and A. Roth. Regret minimization and the price of total anarchy. In *STOC*, pages 373–382, 2008.

[5] J. Burke, D. Estrin, M. Hansen, A. Parker, N. Ramanathan, S. Reddy, and M. B. Srivastava. Participatory sensing. In *World Sensor Web Workshop, ACM Sensys*, 2006.

[6] K. Chaloner and I. Verdinelli. Bayesian experimental design: A review. *Stat. Sci.*, 10(3):273–304, Aug. 1995.

[7] N. A. C. Cressie. *Statistics for Spatial Data*. Wiley, 1991.

[8] A. Das and D. Kempe. Algorithms for subset selection in linear regression. In *STOC*, pages 45–54, 2008.

[9] A. Deshpande, C. Guestrin, S. Madden, J. Hellerstein, and W. Hong. Model-driven data acquisition in sensor networks. In *VLDB*, pages 588–599, 2004.

[10] A. Ostfeld et al. The battle of the water sensor networks (bwsn): A design challenge for engineers and algorithms. *Journal of Water Resources Planning and Management*, 134(6):556–568, 2008.

[11] U. Feige. A threshold of $\ln n$ for approximating set cover. *Journal of the ACM*, 45(4):634–652, 1998.

[12] D. P. Foster and R. Vohra. Regret in the on-line decision problem. *Games and Economic Behavior*, 29(1-2):7–35, October 1999.

[13] D. Golovin, M. Faulkner, and A. Krause. Distributed online sensor selection. *arXiv.org*, arXiv:1002.1782 [cs.LG], Feb. 2010.

[14] H. H. Gonzalez-Banos and J. Latombe. A randomized art-gallery algorithm for sensor placement. In *Proc. 17th ACM Symp. Comp. Geom.*, pages 232–240, 2001.

[15] D. S. Hochbaum and W. Maas. Approximation schemes for covering and packing problems in image processing and VLSI. *Journal of the ACM*, 32:130–136, 1985.

[16] A. Krause and C. Guestrin. Near-optimal nonmyopic value of information in graphical models. In *Proc. of Uncertainty in Artificial Intelligence (UAI)*, 2005.

[17] A. Krause and C. Guestrin. Near-optimal observation selection using submodular functions. In *AAAI Nectar track*, pages 1650–1654, 2007.

[18] A. Krause, J. Leskovec, C. Guestrin, J. VanBriesen, and C. Faloutsos. Efficient sensor placement optimization for securing large water distribution networks. *J. Wat. Res. Plan. Mgmt.*, 136(6), 2008.

[19] A. Krause, A. Singh, and C. Guestrin. Near-optimal sensor placements in Gaussian processes: Theory, efficient algorithms and empirical studies. In *JMLR*, volume 9, pages 235–284, 2008.

[20] F. Kuhn, T. Locher, and R. Wattenhofer. Distributed selection: a missing piece of data aggregation. *Commun. ACM*, 51(9):93–99, 2008.

[21] N. Littlestone and M. K. Warmuth. The weighted majority algorithm. *Information and Computation*, 108(2):212–261, 1994.

[22] K. Nakano and S. Olariu. A survey on leader election protocols for radio networks. In *Parallel Architectures, Algorithms and Networks, 2002. I-SPAN '02.*, pages 63–68, 2002.

[23] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher. An analysis of approximations for maximizing submodular set functions - I. *Mathematical Programming*, 14(1):265–294, 1978.

[24] H. Robbins. Some aspects of the sequential design of experiments. *Bulletin of the American Mathematical Society*, 58:527–535, 1952.

[25] V. Singhvi, A. Krause, C. Guestrin, J. Garrett, and H.S. Matthews. Intelligent light control using sensor networks. In *SenSys*, pages 218–229, 2005.

[26] S. Slijepcevic and M. Potkonjak. Power efficient organization of wireless sensor networks. In *ICC*, pages 472–476, 2001.

[27] M. Streeter and D. Golovin. An online algorithm for maximizing submodular functions. Technical Report CMU-CS-07-171, Carnegie Mellon University, 2007.

[28] M. Streeter and D. Golovin. An online algorithm for maximizing submodular functions. In *NIPS*, pages 1577–1584, 2008.

[29] J. Vondrák. *Submodularity in Combinatorial Optimization*. PhD thesis, Charles University, Prague, Czech Republic, 2007.

[30] M. Widmann and C. S. Bretherton. 50 km resolution daily precipitation for the pacific northwest. http://www.jisao.washington.edu/data_sets/widmann/, May 1999.

[31] J.L. Williams, J.W. Fisher III, and A.S. Willsky. Performance guarantees for information theoretic active inference. In *AISTATS*, 2007.

[32] F. Zhao, J. Shin, and J. Reich. Information-driven dynamic sensor collaboration for tracking applications. *IEEE Signal Processing*, 19(2):61–72, 2002.