Sensor Selection for Minimizing Worst-Case Prediction Error

Abhimanyu Das
University of Southern California
abhimand@usc.edu

David Kempe*
University of Southern California
dkempe@usc.edu

Abstract

We study the problem of choosing the "best" subset of k sensors to sample from among a sensor deployment of $n > k$ sensors, in order to predict aggregate functions over all the sensor values. The sensor data being measured are assumed to be spatially correlated, in the sense that the values at two sensors can differ by at most a monotonically increasing, concave function of their distance. The goal is then to select a subset of sensors so as to minimize the prediction error, assuming that the actual values at unsampled sensors are worst-case subject to the constraints imposed by their distances from sampled sensors.

Even selecting sensors for the optimal prediction of the mean, maximum or minimum is NP-hard; we present approximation algorithms to select near-optimal subsets of k sensors that minimize the worst-case prediction error. In general, we show that for any aggregate function satisfying certain concavity, symmetry and monotonicity conditions, the sensor selection problem can be modeled as a k-median clustering problem, and solved using efficient approximation algorithms designed for k-median clustering.

Our theoretical results are complemented by experiments on two real-world sensor data sets; our experiments confirm that our algorithms lead to prediction errors that are usually less than the (normalized) standard deviation of the test data, using only around 10% of the sensors.

1 Introduction

Sensor networks are becoming increasingly popular for instrumenting and observing natural and industrial environments. A major objective for such deployments is to monitor statistical information about the phenomenon being sensed over potentially large areas. For instance, temperature sensors might monitor the maximum temperature in a chemical reaction; chlorophyll sensors might estimate the average chlorophyll level for algae predictions in a marine biology setting; seismic sensors might monitor the minimum vibration level at a possibly dangerous frequency for a building.

Due to the severe energy constraints inherent to sensor networks, it is desirable to minimize the amount of sensed data needed to estimate such aggregate functions, and thereby reduce the power needed for sensors to make data measurements and transmit them through the network. The problem then is to decide which sensors' data are most valuable to estimate the aggregate function accurately. There is clearly a tradeoff between the amount of sensor data needed, which translates into increased energy expenditure in the sensor network, and the accuracy of estimation of the aggregate function.

Suppose that the sensor network has n sensors, but due to energy constraints, we are allowed to sample only up to $k < n$ of them, to estimate a particular aggregate function of interest. We then need to choose the "best" k sensors to help us predict the aggregate function with minimum error.

In order to reduce the number of measurements, one exploits the *spatial correlation* of sensors, i.e., the fact that sensors "near each other" tend to have "similar values".[1] Generally, it is believed that a good sensor set to sample should be "spread out", i.e., cover the entire area well.

In this paper, we formalize the underlying sensor selection problem, and provide precise characterizations of the objectives quantifying "spread out" for several aggregate functions. We assume that the n sensor nodes are embedded in some known metric space.[2] The distance $d(i, j)$ between two nodes i, j in this metric gives a hard upper bound on the difference between the sensor readings at i and j. That is, any two sensors are spatially correlated in the sense that their values can differ by at most their distance (or a metric-preserving function of their distance) from each other. The goal then is to devise algorithms that choose a set S of size

*Supported in part by NSF CAREER award 0545855, and NSF grant DDDAS-TMRP 0540420

[1] Indeed, spatial correlation is the reason that any discrete deployment, however dense, can give an accurate estimate of an aggregate function over what is intrinsically a continuous domain.

[2] This space may be significantly different from the actual 2- or 3-dimensional Euclidean space; for instance, walls could cause physically proximate nodes to measure very different phenomena.

k to sample, and output an estimate of the aggregate function value, so as to minimize the worst-case estimation error for the function, over all sensor values (both observed and unobserved) satisfying all the distance constraints. (For precise definitions, see Section 2.)

Remark 1.1 *This sensor selection problem can also be alternatively viewed as a sensor placement problem, where there are n potential sensor locations covering an area of interest, but only k sensors, which can be placed at any of these locations. The goal is to choose a set of k locations to place the sensors, in order to best estimate a particular aggregate function that needs to be measured over all the locations. The resulting optimization problem is of course the same as before, and we will continue to use the terminology of sensor selection here.*

Remark 1.2 *It might appear that even the existence of a metric providing such a hard bound on the sensor values cannot be guaranteed. While the metric may not coincide with a low-dimensional Euclidean one, we show in Section 4.1 how to construct such a metric based on observed data, essentially defining the distance between two sensors to be the maximum observed difference in measurements. While this metric may appear to assign unnecessarily large distances, it turns out to yield surprisingly accurate results.*

Our main technical contribution is a characterization of the worst-case sensor values for the average, maximum and minimum functions, as well as for aggregate functions with certain symmetry and concavity conditions (Section 3). In particular, we obtain the following two results:

1. If the function to be estimated is the average, and up to k sensors can be measured, the optimum solution S is the optimum solution for the k-median problem with metric d. That is, it is optimal to minimize $\sum_i d(i, S)$, where $d(i, S) = \min_{j \in S} d(i, j)$.

2. If the goal is to estimate the minimum or maximum using up to k sensors, the optimum solution S is the optimum solution for the k-center problem with metric d. That is, it minimizes $\max_i d(i, S)$.

Since both of these classic problems have good (and practical) approximation algorithms, the corresponding sensor set selection problems can be approximated efficiently.

In general, we show that for any aggregate function with certain symmetry, monotonicity and concavity properties, the optimum solution S can be computed as a solution to a k-median problem.

We note here that we only exploit spatial correlations in the data, but no temporal correlations. The latter is an interesting direction for future extensions.

In Section 4, we evaluate the performance of our algorithms on two real-world sensor data sets, collected from a 54-node network at Intel Labs Berkeley [1], and a 23-node network at the SensorScope Grand St. Bernard Deployment [2]. We use one portion of the data sets to learn an appropriate distance metric, and then observe the accuracy of prediction of the aggregate function on another portion of the data sets. The results show that our algorithms lead to very good accuracy for predicting both the maximum and the average values over the entire network, using a much smaller subset of sensors. The subsets chosen by our algorithms exhibit errors that are usually five times less than the (normalized) standard deviation in the data, using only around 10% of the sensors.

In the process, we also observe that the metrics inferred from the measurements deviate significantly from the Euclidean metric. While this is hardly surprising (various man-made and natural phenomena affect correlations independently of distance), it stresses the fact that correlations between sensor values should be learned from past measurements rather than just identified with physical distance.

1.1 Related Work

Finding "good" sets of sensor nodes to sample is one of the central problems faced by designers and users of sensor networks. The general problem has been phrased in an optimization framework based on utility functions by Byers and Nasser [8] and Bian et al. [7].

The most common approach to formalize the problem is to treat the sensor readings as *random variables* for which statistics such as mean, variance, and covariance are known. Such statistical models can then be used to estimate unobserved sensors' values from other correlated sensor readings. For example, using joint Gaussian models, Deshpande et al. [12] show that the number of sensors to sample can be decreased significantly. Under similar assumptions, Anstreicher et al. [4] and Ko et al. [21] formalize the information gained from a sensor set S as the joint and conditional entropies of S, and consider the corresponding optimization problems of maximum entropy sampling and maximum entropy remote sampling, respectively. Similarly, Guestrin et al. [14] considered the related sensor placement problem using a mutual information objective.

Staying within the framework of random variables, an alternative is to minimize the expected (squared) prediction error, using linear regression [20]. The problem is then equivalent to the well-known problem of subset selection for regression [24, 11] and to sparse approximation of signals over dictionaries [26]. Regression assumes knowledge of the variables' covariance matrix, which can be approximated using the empirically observed covariance. An essentially equivalent approach was recently proposed by Liaskovitis and Schurgers [22].

Our work differs from all these approaches in that we do

not focus on stochastic dependencies between variables, but rather on the case of adversarially chosen values subject to hard constraints. Our goal is thus to minimize the worst-case prediction error, instead of the expected error.

The idea that good sensor sets to sample should provide some notion of "coverage" of an area is ubiquitous in the community, and many different variants have been suggested to formalize the notion of coverage (see, e.g., [3, 23]).

Correlations among sensor values have been exploited to reduce energy consumption in various ways. One approach, orthogonal to ours, is to compress data while forwarding them [25]. Several other papers exploit correlations to reduce the number of transmissions, while explicitly considering the network structure used for forwarding data (e.g., [15, 29]). Yet another paradigm that takes advantage of correlated sensor information is Distributed Source Coding, where all sensors are assumed to have perfect information about all correlations in the network, and can individually compress their data before transmission [28].

The idea of sampling sensors to evaluate an aggregate function has also been suggested by Bash et al. [6] and Ganeriwal et al. [13]. Their focus, however, lies more on the question of how to generate a uniformly random sensor in a decentralized way, and on how to interpolate data based on those samples.

2 Preliminaries and Notation

We first define some notation. Bold-face letters $\mathbf{x}$ always denote vectors, whose entries are x_i. We write $\mathbf{x}_{i \mapsto v}$ to denote the vector whose entries agree with $\mathbf{x}$, except the i^{th} entry, which is v instead. Whenever we write $\mathbf{x} \leq \mathbf{x}'$ for vectors $\mathbf{x}$ and $\mathbf{x}'$, we mean that the inequality holds for *all* coordinates. We will explicitly allow vectors with some undefined entries $\perp$.

We formally define our framework for optimizing the sensor selection. The n sensors[3] $\{1, \ldots, n\}$ are assumed to be embedded in some metric space. We use $d(i, j)$ to denote the *distance* between sensors i and j. These distances satisfy the usual metric constraints, i.e., symmetry ($d(i, j) = d(j, i)$), non-negativity ($d(i, j) \geq 0$), and the triangle inequality ($d(i, j) + d(j, k) \geq d(i, k)$ for all i, j, k). Extending the notation to sets, we write $d(i, S) := \min_{j \in S} d(i, j)$ for the distance of location i from set S.

The sensor reading at sensor i is denoted by x_i, and the vector of all sensor readings by $\mathbf{x}$. The distances impose hard constraints on how much the readings of two sensors i and j can differ. We define a *valid assignment* of sensor readings $\mathbf{x}$ to be one in which $|x_i - x_j| \leq d(i, j)$ for all pairs i, j. Validity of assignments is precisely the sense in which we quantify the informal notion that "nearby nodes have nearby values." We use the assignment $x_i = \perp$ to express that the reading at sensor i is unknown. A *valid S-assignment* is then a valid assignment for which $x_i \neq \perp$ for all $i \in S$. (Thus, a valid assignment is a valid $\{1, \ldots, n\}$-assignment.) In particular, we will frequently refer to the S-assignment vector with its i^{th} component equal to 0 if $i \in S$, and equal to $\perp$ for $i \notin S$, and denote it by $\mathbf{0}^{(\mathbf{S})}$.

Remark 2.1 *We stress here that the metric space need not, and usually will not, coincide with the 3-dimensional Euclidean space in which the sensors are actually placed. For instance, building walls may result in two sensors with small Euclidean distance measuring distinct phenomena; conversely, two distant rooms could be on the same air-conditioning control, and temperature readings in those two rooms would be very close to each other. Thus, in general, the metric should be inferred from past measurements or domain knowledge, rather than identified with Euclidean distance (we discuss this point in more depth in Section 4.1).*

Remark 2.2 *The constraint that readings differ by at most the distance may seem overly restrictive, and it may seem desirable to add a function $\phi : \mathbb{R}^+ \to \mathbb{R}^+$ to the problem specification, requiring that $|x_i - x_j| \leq \phi(d(i, j))$. However, so long as ϕ is monotone non-decreasing, concave, and satisfies $\phi(0) = 0$, replacing each distance $d(i, j)$ with $d'(i, j) := \phi(d(i, j))$ can be easily seen to give rise to a new metric space in which now $|x_i - x_j| \leq d'(i, j)$. Thus, the addition of a "correlation decay function" of the distance does not add expressive power to the model.*

The purpose of the sensor network is to estimate some *aggregate function* $f(\mathbf{x})$. Standard examples include the *average* $f(\mathbf{x}) := \frac{1}{n} \sum_i x_i$, or the *maximum* or *minimum* $f(\mathbf{x}) := \max_i x_i$ or $f(\mathbf{x}) := \min_i x_i$.

Due to energy constraints, the algorithm is only allowed to sample up to k of the sensor readings x_i. Thus, an algorithm will first compute, in polynomial time, a set $S \subseteq \{1, \ldots, n\}$ of k sensors to sample. By retrieving their values, the algorithm will learn a valid S-assignment vector $\mathbf{x}^{(\mathbf{S})}$, where $x_i^{(S)} = \perp$ (undefined) for $i \notin S$, and $x_i^{(S)} = x_i$ for $i \in S$. To this S-assignment, the algorithm will then apply a prediction function $\tilde{f}$ (possibly different from f) to compute its output $\tilde{f}(\mathbf{x}^{(\mathbf{S})})$. The goal of the algorithm is to minimize its *prediction error* $|\tilde{f}(\mathbf{x}^{(\mathbf{S})}) - f(\mathbf{x})|$. We call this problem *prediction error minimization*.

In this paper, we study this problem from a worst-case perspective. That is, we want to design algorithms which will minimize the prediction error under the assumption of a worst case valid assignment. Formally, the goal is to choose $\tilde{f}$ and S so as to minimize $\max_{\mathbf{x} \text{valid}} |\tilde{f}(\mathbf{x}^{(\mathbf{S})}) - f(\mathbf{x})|$.

[3] In principle, our approach can be extended to deal with an infinite metric space, e.g., a continuous space into which sensors must be placed. The necessary modifications are fairly straightforward.

In this setting, it is convenient (and common) to think of the values $\mathbf{x}$ as being chosen by an *adversary* with full knowledge of the algorithm. So long as the algorithm is deterministic (we briefly discuss the very interesting case of randomized algorithms in Section 5), the adversary can predict at the outset which set S the algorithm will choose, and hence we can think of the process as follows: First, the algorithm chooses a set S. Then, the adversary (knowing the prediction function $\tilde{f}$ used by the algorithm) reveals a valid S-assignment $\mathbf{x}$ to the sensors in S. Using this assignment, the algorithm now computes and outputs $\tilde{f}(\mathbf{x})$, after which the adversary chooses and reveals the valid assignment $\mathbf{x}'$ that includes the values for the remaining sensors (in $\overline{S}$).

The values the adversary chooses in the final step must also be valid. We say that $\mathbf{x}'$ is a *valid S-extension* of the S-assignment $\mathbf{x}$ if $\mathbf{x}'$ is a valid assignment, and $x'_i = x_i$ for all $i \in S$. Given a set S and valid S-assignment $\mathbf{x}$, we are interested in the worst prediction error the adversary can cause against the optimal prediction function $\tilde{f}$. Formally, we define $E(S, \mathbf{x}) := \max_{\mathbf{x}'} |\tilde{f}(\mathbf{x}) - f(\mathbf{x}')|$, where $\mathbf{x}'$ ranges over all valid S-extensions of $\mathbf{x}$. The worst-case error for a sample set S is then $E(S) := \max_{\mathbf{x}} E(S, \mathbf{x})$, where $\mathbf{x}$ ranges over all valid S-assignments.

In our analysis, we will frequently want to reason about the largest or smallest possible values that the adversary could choose for sensors outside S. We therefore define two specific S-extensions $\mathbf{x}^+$ and $\mathbf{x}^-$ as follows: For all i, $x_i^+ := \min_j(x_j + d(i,j))$, and $x_i^- := \max_j(x_j - d(i,j))$. (Notice that for $i \in S$ and valid $\mathbf{x}$, the definition implies $x_i^+ = x_i^- = x_i$.) Using triangle inequality, it is then easy to verify the following:

Proposition 2.3 *$\mathbf{x}^+$ and $\mathbf{x}^-$ are valid S-extensions of $\mathbf{x}$.*

We will see below that given a choice of a set S and a valid S-assignment $\mathbf{x}$ to S, the optimum prediction function $\tilde{f}$ can be computed in polynomial time. Thus, the crux of the problem is to choose the set S so as to minimize $E(S)$.

Our algorithms for sensor selection will be based on known algorithms for the metric k-median and metric k-center problems. We formally define those problems here.

Definition 2.4 (k-median) *Given a distance metric d and a number k, select a set S of k nodes so as to minimize $\sum_i d(i, S)$.*

Definition 2.5 (k-center) *Given a distance metric d and a number k, select a set S of k nodes so as to minimize $\max_i d(i, S)$.*

3 Characterizing Optimal Algorithms

In this section, we analyze the problem from the adversary's perspective, and show that both for the average and minimum/maximum objectives, the adversary's strategy is to reveal identical readings at all locations in S. This observation in turn can be used to rephrase the algorithm's selection as clustering objectives (metric k-median and metric k-center, respectively), for which good approximation algorithms are known.

Subsequently, we generalize the insights from the argument for the average objective, and describe sufficient conditions for an aggregate function under which the adversary will reveal identical readings at all sensors.

3.1 Predicting the Average

We first consider the case when $f(\mathbf{x}) = \frac{1}{n}\sum_i x_i$.

Theorem 3.1 *For any set S, the adversary's best strategy is to reveal identical readings of 0 at all $i \in S$. That is, $E(S) = E(S, \mathbf{0}^{(\mathbf{S})})$.*

The worst case error is exactly $E(S) = \frac{1}{n} \cdot \sum_{i \notin S} d(i, S)$.

Proof. Let $\mathbf{x}$ be any valid S-assignment. For any $i \notin S$ and $j \in S$, the distance constraint on sensor values implies $x_j - d(i,j) \le x'_i \le x_j + d(i,j)$ for all valid S-extensions $\mathbf{x}'$ of $\mathbf{x}$, i.e., $\mathbf{x}^- \le \mathbf{x}' \le \mathbf{x}^+$ for all valid S-extensions $\mathbf{x}'$ of $\mathbf{x}$.

Since an adversary having revealed $\mathbf{x}$ on S can always choose either to extend it to $\mathbf{x}^-$ or to $\mathbf{x}^+$ after the algorithm has output $\tilde{f}(\mathbf{x})$, the optimum algorithm must be the one outputting $\frac{1}{2}(f(\mathbf{x}^+) + f(\mathbf{x}^-)) = \frac{1}{n} \cdot \sum_i \frac{1}{2}(x_i^+ + x_i^-)$. Otherwise, the adversary could choose the one among $\mathbf{x}^+$ and $\mathbf{x}^-$ farther away from $\tilde{f}(\mathbf{x})$ and guarantee a worse prediction error. Notice that both $\mathbf{x}^+$ and $\mathbf{x}^-$ can be computed by an algorithm in time $O(kn)$ once S and $\mathbf{x}$ are known.

When the algorithm makes this optimal choice, the prediction error will be exactly

$$\begin{aligned} f(\mathbf{x}^+) - \tfrac{1}{2}(f(\mathbf{x}^+) + f(\mathbf{x}^-)) &= \tfrac{1}{2}(f(\mathbf{x}^+) - f(\mathbf{x}^-)) \\ &= \tfrac{1}{n} \cdot \textstyle\sum_i \tfrac{1}{2}(x_i^+ - x_i^-). \end{aligned}$$

By definition of $\mathbf{x}^+$ and $\mathbf{x}^-$, we get that

$$\begin{aligned} x_i^+ - x_i^- &= \min_{j \in S}(x_j + d(i,j)) \\ &\quad - \max_{j \in S}(x_j - d(i,j)) \\ &\le (x_j + d(i,j)) - (x_j - d(i,j)) \\ &= 2d(i,j), \end{aligned}$$

for all $j \in S$. In particular, $x_i^+ - x_i^- \le 2d(i, S)$, and the maximum error the adversary can achieve is therefore $E(S, \mathbf{x}) \le \frac{1}{n} \cdot \sum_i d(i, S)$.

On the other hand, if the given assignment is $\mathbf{0}^{(\mathbf{S})}$, then $x_i^+ = d(i, S)$ and $x_i^- = -d(i, S)$, giving a matching error of $\frac{1}{n} \cdot \sum_i d(i, S)$ against the best algorithm. Thus, choosing $\mathbf{0}^{(\mathbf{S})}$ achieves the upper bound, and must be optimal. ■

The proof explicitly gives us the optimum prediction function $\tilde{f}$ for the algorithm as a corollary:

Corollary 3.2 *Given a choice S of a sensor set and an S-assignment $\mathbf{x}$ that has been revealed, the optimum prediction function is $\tilde{f}(\mathbf{x}) := \frac{1}{2}(f(\mathbf{x}^+) + f(\mathbf{x}^-))$. $\tilde{f}$ can be computed in time $O(kn)$.*

By Theorem 3.1, the optimum set S the algorithm should choose to sample is exactly the one minimizing $\sum_i d(i,S)$. But this is precisely the objective function of the well-known k-median problem. Since the $d(i,j)$ are assumed to be a metric, the problem of selecting the optimum set S to sample is equivalent to the metric k-median problem, giving us the following corollary.

Corollary 3.3 *1. For every $\varepsilon > 0$, there is a polynomial-time $(3+\varepsilon)$-approximation algorithm for the problem of choosing the k-element set S minimizing $E(S)$.*

2. Unless $\mathrm{NP} \subseteq \mathrm{DTIME}[n^{O(\log\log n)}]$, there is no polynomial-time algorithm approximating the best set selection to within $1 + 2/\mathrm{e}$ (where e is the basis of the natural logarithm).

The corollary follows directly from the corresponding results of Arya et al. [5] and Jain et al. [18] for the metric k-median problem. Since we use it in our experimental evaluation, we will describe the local search based algorithm of Arya et al. in more detail in Section 3.4.1.

3.2 Predicting the Maximum or Minimum

We now consider the case when $f(\mathbf{x}) = \max_i x_i$. (The case of the minimum is analogous.) Similar to the case of the average, we first show that for any given subset S of sensors, the adversary achieves worst-case error $E(S)$ by showing the vector $\mathbf{0}^{(\mathbf{S})}$.

Theorem 3.4 *For any set S, $E(S) = E(S, \mathbf{0}^{(\mathbf{S})})$. That is, the adversary can maximize the prediction error by showing value 0 at all sampled locations.*

The worst-case error is exactly $E(S) = \frac{1}{2}\max_i d(i,S)$.

Proof. Let $\mathbf{x}$ be any valid S-assignment. By the same reasoning as before, the optimum algorithm must output $\frac{1}{2}(f(\mathbf{x}^+) + f(\mathbf{x}^-))$, which in the case of predicting the maximum equals $\frac{1}{2}((\max_i \min_{j\in S}(x_j + d(i,j))) + \max_i \max_{j\in S}(x_j - d(i,j)))$. The first term can be upper bounded by $\max_i(d(i,S) + \max_{j\in S} x_j) = \max_i d(i,S) + \max_{j\in S} x_j$, and the second term always attains its maximum for some $i \in S$, so it equals $\max_{j\in S} x_j$. Thus, the worst-case prediction error is

$$f(\mathbf{x}^+) - \tfrac{1}{2}(f(\mathbf{x}^+) + f(\mathbf{x}^-)) \quad \leq \quad \tfrac{1}{2}\max_i d(i,S).$$

On the other hand, if the adversary chooses $\mathbf{x} = \mathbf{0}^{(\mathbf{S})}$, the error is exactly $E(S, \mathbf{0}^{(\mathbf{S})}) = \frac{1}{2}\max_i d(i,S)$. Thus, it is optimal for the adversary to show $\mathbf{0}^{(\mathbf{S})}$. ■

Again, a corollary of the proof of Theorem 3.4 provides an optimal prediction function for the algorithm to use.

Corollary 3.5 *Given a choice S of a sensor set and an S-assignment $\mathbf{x}$ that was revealed, the optimum prediction function is $\tilde{f}(\mathbf{x}) := \frac{1}{2}((\max_i \min_{j\in S}(x_j + d(i,j))) + \max_{j\in S} x_j)$. $\tilde{f}$ can be computed in time $O(kn)$.*

By Theorem 3.4, the algorithm should choose the set S minimizing $\max_{i\notin S} d(i,S)$. This is the objective function of the (metric) k-center problem. Thus, the problem of selecting the optimum set S to sample is equivalent to the metric k-center problem, giving us the following corollary.

Corollary 3.6 *1. There is a 2-approximation algorithm for the problem of choosing the k-element set S minimizing $E(S)$.*

2. Unless $\mathrm{P} = \mathrm{NP}$, there is no polynomial-time algorithm approximating the best set selection to within $2 - \varepsilon$, for any $\varepsilon > 0$.

The corollary follows from the results for metric k-center due to Hochbaum and Shmoys [16] and Hsu and Nemhauser [17]. The greedy 2-approximation algorithm that we use is described in Section 3.4.2.

3.3 A general result

We now derive a more general condition under which the adversary's best strategy is to reveal the vector $\mathbf{0}^{(\mathbf{S})}$.

Definition 3.7 (0-centered) *We say that an aggregate function f is* 0-centered *if it satisfies the following four conditions:*

1. *f is monotonically non-decreasing in all its variables and has well-defined first and second partial derivatives.*
2. *The partial derivatives are symmetric around 0 in each variable. In other words, for each i and j, we have $\frac{\partial}{\partial x_i} f(\mathbf{x}) = \frac{\partial}{\partial x_i} f(\mathbf{x}_{j\mapsto -x_j})$.*
3. *For all i, $\frac{\partial^2}{\partial x_i^2} f(\mathbf{x}) \leq 0$ whenever $x_i > 0$, and $\frac{\partial^2}{\partial x_i^2} f(\mathbf{x}) \geq 0$ whenever $x_i < 0$.*
4. *Partial derivatives are maximized near 0, in the sense that $\frac{\partial}{\partial x_i} f(\mathbf{0}_{i\mapsto x_i}) \geq \frac{\partial}{\partial x_i} f(\mathbf{x})$ for all i.*

A sufficient condition for being 0-centered is if f is of the form $f(\mathbf{x}) = \sum_i f_i(x_i)$, where each $f_i : \mathbb{R} \to \mathbb{R}$ is a monotone function which is concave on $\mathbb{R}^+$ and convex on $\mathbb{R}^-$. In particular, the condition thus applies to the sum and average functions.

We first prove a lemma showing that the largest differences in a 0-centered function are achieved around 0.

Lemma 3.8 *Let f be 0-centered. For any vectors $\mathbf{x}, \mathbf{b}$ with $x_i \geq 0$ for all i, $f(\mathbf{x}) - f(-\mathbf{x}) \geq f(\mathbf{b}+\mathbf{x}) - f(\mathbf{b}-\mathbf{x})$.*

Proof. Define $\mathbf{x^{(j)}}$ to be the vector with $x_i^{(j)} = -x_i$ for $i \leq j$, and $x_i^{(j)} = x_i$ for $i > j$. Thus, $\mathbf{x^{(0)}} = \mathbf{x}$, and $\mathbf{x^{(n)}} = -\mathbf{x}$, and we can write a telescoping series

$$f(\mathbf{x}) - f(-\mathbf{x}) \quad = \quad \sum_{i=1}^{n} (f(\mathbf{x^{(i-1)}}) - f(\mathbf{x^{(i)}})).$$

We next show that $f(\mathbf{x^{(i-1)}}) - f(\mathbf{x^{(i)}}) = f(\mathbf{z}^{(i)}) - f(-\mathbf{z}^{(i)})$ for all i, where $\mathbf{z}^{(i)} = \mathbf{0}_{i \mapsto x_i}$ is the vector with entries $z_j^{(i)} = 0$ for $j \neq i$, and $z_i^{(i)} = x_i$. For this purpose, fix an i, and for each j define $\mathbf{y}^{(i,j)}$ as the vector with $y_r^{(i,j)} = -x_r$ for $r \leq \min(i,j)$ or $r = i$, $y_r^{(i,j)} = x_r$ for $i < r \leq j$, and $y_r^{(i,j)} = 0$ for $r > j, r \neq i$. In addition, define $\hat{\mathbf{y}}^{(i,j)} := \mathbf{y}^{(i,j)}_{i \mapsto x_i}$, i.e. $\hat{\mathbf{y}}^{(i,j)}$ agrees with $\mathbf{y}^{(i,j)}$, except for the sign in coordinate i.

We claim that $f(\hat{\mathbf{y}}^{(i,j-1)}) - f(\mathbf{y}^{(i,j-1)}) = f(\hat{\mathbf{y}}^{(i,j)}) - f(\mathbf{y}^{(i,j)})$ for all j. We can equivalently rewrite this claim as $f(\mathbf{y}^{(i,j)}) - f(\mathbf{y}^{(i,j-1)}) = f(\hat{\mathbf{y}}^{(i,j)}) - f(\hat{\mathbf{y}}^{(i,j-1)})$. For $j = i$, this claim is trivial. For other j, we will express the differences as integrals of differentials, and apply condition 2. Specifically, by substituting the definitions, we rewrite the left-hand side as $\int_0^{y_j^{(i,j)}} \frac{\partial}{\partial x_j} f(\mathbf{y}^{(i,j)}_{j \mapsto t}) dt$. Similarly, the right-hand side is $\int_0^{\hat{y}_j^{(i,j)}} \frac{\partial}{\partial x_j} f(\hat{\mathbf{y}}^{(i,j)}_{j \mapsto t}) dt$. But by definition of $\mathbf{y}^{(i,j)}$ and $\hat{\mathbf{y}}^{(i,j)}$, the two only differ in the sign of the i^{th} coordinate, so the second property in the definition of being 0-centered implies that $\frac{\partial}{\partial x_j} f(\mathbf{y}^{(i,j)}_{j \mapsto t}) = \frac{\partial}{\partial x_j} f(\hat{\mathbf{y}}^{(i,j)}_{j \mapsto t})$, and the two integrals are equal, proving the claim.

By induction over j, we therefore also obtain that $f(\hat{\mathbf{y}}^{(i,0)}) - f(\mathbf{y}^{(i,0)}) = f(\hat{\mathbf{y}}^{(i,n)}) - f(\mathbf{y}^{(i,n)})$. But notice that $\hat{\mathbf{y}}^{(i,0)} = \mathbf{z}^{(i)}, \mathbf{y}^{(i,0)} = -\mathbf{z}^{(i)}$, and $\hat{\mathbf{y}}^{(i,n)} = \mathbf{x^{(i-1)}}$ and $\mathbf{y}^{(i,n)} = \mathbf{x^{(i)}}$, proving that $f(\mathbf{x^{(i-1)}}) - f(\mathbf{x^{(i)}}) = f(\mathbf{z}^{(i)}) - f(-\mathbf{z}^{(i)})$. Thus, we obtain that

$$f(\mathbf{x}) - f(-\mathbf{x}) \quad = \quad \sum_{i=1}^{n} (f(\mathbf{z}^{(i)}) - f(-\mathbf{z}^{(i)})).$$

Similarly, we can write

$$\begin{array}{l} f(\mathbf{b}+\mathbf{x}) - f(\mathbf{b}-\mathbf{x}) \\ \quad = \quad \sum_{i=1}^{n} (f(\mathbf{b}+\mathbf{x^{(i-1)}}) - f(\mathbf{b}+\mathbf{x^{(i)}})). \end{array} \tag{1}$$

We will use conditions 3 and 4 of the definition of being 0-centered to upper bound each term of the telescoping sum $f(\mathbf{b}+\mathbf{x^{(i-1)}}) - f(\mathbf{b}+\mathbf{x^{(i)}}) \leq f(\mathbf{z}^{(i)}) - f(-\mathbf{z}^{(i)})$. The idea is again to write a difference as an integral of differentials:

$$\begin{array}{rcl} f(\mathbf{b}+\mathbf{x^{(i-1)}}) - f(\mathbf{b}+\mathbf{x^{(i)}}) & & \\ = & \int_{b_i - x_i}^{b_i + x_i} \frac{\partial}{\partial x_i} f((\mathbf{b}+\mathbf{x^{(i)}})_{i \mapsto t}) dt & \\ \overset{(4)}{\leq} & \int_{b_i - x_i}^{b_i + x_i} \frac{\partial}{\partial x_i} f(\mathbf{0}_{i \mapsto t}) dt. & \end{array} \tag{2}$$

For simplicity, we write $h(t) := \frac{\partial}{\partial x_i} f(\mathbf{0}_{i \mapsto t})$, as well as $\ell := b_i - x_i$ and $u := b_i + x_i$. Condition 3 in the definition of being 0-centered implies that h is monotone decreasing in t for $t \geq 0$, and monotone increasing for $t \leq 0$, and condition 2 implies that $h(t) = h(-t)$ for all t. If $\ell \geq 0$, monotonicity implies that $\int_\ell^u h(t)dt \leq \int_0^{u-\ell} h(t)dt$. Similarly for $u \leq 0$. Next, once we have $\ell \leq 0 \leq u$, we assume w.l.o.g. that $u + \ell > 0$ (the case $u + \ell < 0$ is symmetric). Because $\int_{(u-\ell)/2}^{u} h(t)dt \leq \int_{-\ell}^{(u-\ell)/2} h(t)dt = \int_{(\ell-u)/2}^{\ell} h(t)dt$ by monotonicity,

$$\begin{array}{rcl} \int_\ell^u h(t)dt & = & \int_\ell^{(u-\ell)/2} h(t)dt + \int_{(u-\ell)/2}^u h(t)dt \\ & \leq & \int_{(\ell-u)/2}^{(u-\ell)/2} h(t)dt. \end{array}$$

But $(\ell - u)/2 = -x_i$ and $(u - \ell)/2 = x_i$, so we obtain that

$$\int_{b_i - x_i}^{b_i + x_i} \frac{\partial}{\partial x_i} f(\mathbf{0}_{i \mapsto t}) dt \quad \leq \quad \int_{-x_i}^{x_i} \frac{\partial}{\partial x_i} f(\mathbf{0}_{i \mapsto t}) dt. \tag{3}$$

Combining Equations (1), (2) and (3), we obtain that

$$\begin{array}{rcl} f(\mathbf{b}+\mathbf{x}) - f(\mathbf{b}-\mathbf{x}) & \leq & \sum_i f(\mathbf{z}^{(i)}) - f(-\mathbf{z}^{(i)}) \\ & = & f(\mathbf{x}) - f(-\mathbf{x}). \end{array}$$ ■

Using the above lemma, we can now prove that the worst-case prediction error for f with set S sampled is maximized when the adversary reveals the vector $\mathbf{0^{(S)}}$.

Theorem 3.9 *If f is 0-centered, then for any subset S of sensors, $E(S) = E(S, \mathbf{0^{(S)}})$. That is, the adversary can maximize the prediction error by showing the value 0 at all sampled locations.*

Furthermore, the worst-case error is then exactly $E(S) = \frac{1}{2}(f(\mathbf{\Delta}) - f(-\mathbf{\Delta}))$, where $\mathbf{\Delta}$ is the vector of distances from S, i.e., $\Delta_i = d(i, S)$.

Proof. Let $\mathbf{x}$ be any valid S-assignment. By the monotonicity of f (condition 1), the maximum of f is achieved by $\mathbf{x}^+$, and the minimum by $\mathbf{x}^-$ (both over valid S-extensions). Thus, the optimum algorithm outputs $\frac{1}{2}(f(\mathbf{x}^+) + f(\mathbf{x}^-))$, leading to a worst-case error of $\frac{1}{2}(f(\mathbf{x}^+) - f(\mathbf{x}^-))$.

For each i, let $j_i \in S$ be the closest sensor to i, so that $d(i, j_i) = d(i, S)$. Then, the definitions of $\mathbf{x}^+$ and $\mathbf{x}^-$ imply that $x_i^+ \leq x_{j_i} + d(i, S)$ and $x_i^- \geq x_{j_i} - d(i, S)$. Writing $\mathbf{b}$ for the vector $b_i = x_{j_i}$ for all i, the previous argument then gives that $\mathbf{x}^+ \leq \mathbf{b} + \mathbf{\Delta}$ and $\mathbf{x}^- \geq \mathbf{b} - \mathbf{\Delta}$. Monotonicity of f implies that $f(\mathbf{x}^+) - f(\mathbf{x}^-) \leq f(\mathbf{b}+\mathbf{\Delta}) - f(\mathbf{b}-\mathbf{\Delta})$.

The latter, in turn, is at most $f(\mathbf{\Delta}) - f(-\mathbf{\Delta})$ by Lemma 3.8, so $E(S, \mathbf{x}) \leq \frac{1}{2}(f(\mathbf{\Delta}) - f(-\mathbf{\Delta}))$ for all $\mathbf{x}$, and thus $E(S) \leq \frac{1}{2}(f(\mathbf{\Delta}) - f(-\mathbf{\Delta}))$.

On the other hand, for the vector $\mathbf{0^{(S)}}$, we obtain that $\mathbf{x}^+ = \mathbf{\Delta}$ and $\mathbf{x}^- = -\mathbf{\Delta}$, giving $E(S, \mathbf{0^{(S)}}) = E(S)$. ■

As before, the proof also gives us the optimum prediction function $\tilde{f}$ for the algorithm:

Corollary 3.10 *Given a choice S of a sensor set and an S-assignment $\mathbf{x}$ that was revealed, the optimum prediction function is $\tilde{f}(\mathbf{x}) := \frac{1}{2}(f(\mathbf{x}^+) + f(\mathbf{x}^-))$. Assuming that f can be evaluated efficiently, $\tilde{f}$ can be computed in time $O(kn)$.*

Using the observation made in the proof of Lemma 3.8, together with the error bound proved in Theorem 3.9, we observe that the worst-case error incurred when sampling set S is $E(S) = \frac{1}{2}\sum_i (f(\mathbf{0}_{i \mapsto d(i,S)}) - f(\mathbf{0}_{i \mapsto -d(i,S)}))$. Writing $g_i(x) := f(\mathbf{0}_{i \mapsto x}) - f(\mathbf{0}_{i \mapsto -x})$, the error is thus of the form $E(S) = \frac{1}{2}\sum_i g_i(d(i, S))$, where each g_i is a monotone increasing concave function with $g_i(0) = 0$.

In general, it is not clear how well clustering problems with such objective functions can be approximated. However, if we make the additional assumption that f is symmetric in its arguments, i.e., that $f(\mathbf{x}) = f(x_{\pi(1)}, \ldots, x_{\pi(n)})$ for every permutation π of $\{1, \ldots, n\}$, then all functions g_i are equal, so $E(S) = \frac{1}{2}\sum_i g(d(i, S))$. It can then be seen that by replacing the metric with $d'(i, j) := \frac{1}{2}g(d(i, j))$ (which is a metric by the properties of g), we obtain the equivalent problem of choosing S to minimize $\sum_i d'(i, S)$. That is, for symmetric 0-centered functions f, the problem reduces to k-median again.

3.4 Approximation Algorithms

Here, we describe the approximation algorithms for k-median and k-center used in our experiments. Both are well-studied NP-hard optimization problems [27].

3.4.1 The k-median algorithm

For the metric k-median problem, since the discovery of the first constant-factor approximation by Charikar et al. [10] (based on LP-rounding), a series of papers ([19, 9, 5], among others) has led to improved approximation bounds and more practical algorithms, using a variety of techniques. On the other hand, Jain et al. [18] have shown an approximation hardness result of $1 + 2/\mathrm{e}$ for metric k-median unless $\mathrm{NP} \subseteq \mathrm{DTIME}[n^{O(\log \log n)}]$.

We use the approximation algorithm for k-median due to Arya et al. [5]. It is based on a simple local search procedure, and can be stated as follows.

The algorithm, with a parameter p, starts with an arbitrary subset S of k sensors, and keeps improving it by swapping p sensors at a time, until the maximum possible improvement is "small". Specifically, for some polynomial $q(n)$ and a parameter ε, the algorithm tries all sets $S' \subset S$ of size $|S'| \leq p$, as well as all $T \subseteq \overline{S}$ of size $|T| = |S'|$, and updates S to $S \cup T \setminus S'$ if the objective function $\sum_i d(i, S)$ decreases by at least a factor $1 - \frac{\varepsilon}{q(n)}$. Arya et al. [5] show that this algorithm gives a $3 + \frac{2}{p} + \varepsilon$ approximation, i.e., the algorithm outputs a set S such that the total distance of all nodes from S is within a $3 + \frac{2}{p} + \varepsilon$ factor of the total distance from the best set S^*. We call this algorithm the *local search algorithm for k-median*. In our implementation of this algorithm, we chose the value of the parameter $p = 2$, guaranteeing a $4 + \varepsilon$ approximation. (We noticed that in our experiments, the performance with $p = 1$ was not significantly different from the performance with $p = 2$.)

3.4.2 The k-center algorithm

For k-center, Hochbaum and Shmoys [16] give a simple 2-approximation algorithm based on parametric pruning and greedy selection. The 2-approximation is also the best possible unless $\mathrm{P} = \mathrm{NP}$ [17].

We use a simple greedy algorithm with the same approximation guarantee, and based on similar ideas: Starting from one arbitrary sensor $S = \{i\}$, the algorithm repeatedly adds to S the location j maximizing $d(j, S)$, and terminates once $|S| = k$.

Let j be a point farthest from S, i.e., maximizing $d(j, S)$, and let $\delta = d(j, S)$. Then, by the choice of the algorithm, $S \cup \{j\}$ is a set at pairwise distance at least δ; hence, any set S' of at most k points must have distance at least $\delta/2$ to at least one of them. Therefore, the algorithm is a 2-approximation. (The analysis sketch is analogous to the parametric pruning analysis of Hochbaum and Shmoys [16].) We refer to this algorithm as the *greedy algorithm for k-center*.

4 Experimental Results

We validate our methods on two real-world data sets. The first data set consists of a 54-node indoor sensor deployment at Intel Labs Berkeley [1]. The measurements were collected over a one-month period in March 2004. They comprised light, temperature, and humidity measurements, measured at each of the sensors every few seconds.

The second data set consists of measurements from 23 sensors from the SensorScope Deployment [2], deployed outdoors at the Grand St. Bernard Pass. These measurements were collected over a one-week period in October 2007, and comprise temperature, humidity, wind speed, radiation and soil moisture values, measured at each sensor

every few minutes.

While both data sets contain measurements for multiple physical attributes, our experiments focus only on the light data from the Intel data set (measured in Lux), and the relative humidity data from the SensorScope data (measured in percentage). We restrict our attention to these two attributes because they exhibit larger standard deviations than the others. The other attributes (in particular, temperature) show less spatial variation, so high prediction accuracy for them would be statistically less significant.

4.1 Estimating the Distance Metric

Our algorithms require knowledge of the distance metric $d(i, j)$ constraining the difference in sensor readings. Our experiments have shown that the best such metric tends to be rather independent of the Euclidean distance between sensors. When trying to fit a mathematical function of the Euclidean inter-sensor distances to the difference in values at the sensor nodes, the result has high sum-of-squares fitting errors. This is not particularly surprising, given that the Euclidean distance ignores important physical parameters such as walls or presence or absence of shade.

We therefore use an empirical method to directly estimate a distance metric, using a training data set. We define the "distance" between two sensors to be the worst case absolute difference in readings at the two sensors over all instances of the training data.

Specifically, we partition our data sets into *training instances* and the *test instances*. The training instances are used by the algorithm to estimate the distances, and the test instances are used to evaluate the subsequent prediction accuracy. Thus, the sensor network can adapt its sampling strategy based on the correlations estimated from past measurements.

Each training instance consists of snapshots in time of all the sensor values. For each sensor pair (i, j), we compute the largest difference in values between i and j over all training instances, and use the difference as an estimate of $d(i, j)$ for the future. It is straightforward to prove that the function estimated in this way forms a metric. In effect, we are embedding the sensors from the Euclidean space into a new metric space d. This eliminates the dependency on the physical coordinates of the sensors, while allowing significantly better performance in our experiments.

4.2 Experimental Methodology

We consider one attribute each from the two data sets: light from the first data set, and humidity from the second data set. For each data set, we fix the value of k (the number of sensors to sample) to be around 10% of the total number of sensors.

We report on three sets of experiments: estimating the maximum and average functions for the light measurements in the Intel Labs deployments, and estimating the average function for the humidity measurements in the SensorScope deployment. (An estimate of the maximum humidity in the SensorScope deployment performed similarly.) For each experiment, we use the following methodology:

1. Sample the entire set of sensors multiple times over a time window to generate a set of training instances. Each training instance corresponds to a snapshot of all the sensor readings in the deployment at a particular time. Similarly, sample the set of sensors multiple times over another time window to generate a set of test instances.

2. Clean the training and test instances to remove outliers.

3. Estimate the distance metric $d(i, j)$ based on the training instances, and run the sensor selection algorithm using this distance metric to get a set S of k sensors.

4. For each test instance, use the readings at the k sensors and the prediction function $\tilde{f}(\mathbf{x}^{(\mathbf{S})})$ to estimate the objective function $f(\mathbf{x})$, and calculate the prediction error $|\tilde{f}(\mathbf{x}^{(\mathbf{S})}) - f(\mathbf{x})|$.

We evaluate our prediction errors in two ways. The first involves estimating the standard deviation of the sensors in the training data set, and comparing our prediction error with the standard deviation. The second involves comparing our prediction errors with the average and minimum prediction error obtained by using multiple randomly generated sets of sensors, using the same test instances.

4.3 Light Measurements

In this set of experiments, we use our algorithms on light measurements from the Intel Labs sensor deployment. We perform our analysis using data for ambient light (measured in Lux) detected at each sensor, taken over a period of 9 days from March 1st to March 9th. We cleaned the data by using a median filter to eliminate outliers.

We do not consider measurements after 9PM and before 9AM, since we observe that the mean and standard deviation during these times are usually very close to zero. In other words, there is no challenge in providing accurate predictions, and good performance on these data would not be indicative of the quality of our or other algorithms.

We start by calculating the standard deviation of the test data to estimate the dispersion of the data. This helps us gauge the statistical significance of results for any prediction algorithm on this data set. Figure 1 plots the coefficient of variation (the ratio of standard deviation to the mean) for the light measurements over different days and times. The

standard deviation varies from 50% to as much as 130% of the mean, with an average standard deviation of 71% of the mean.

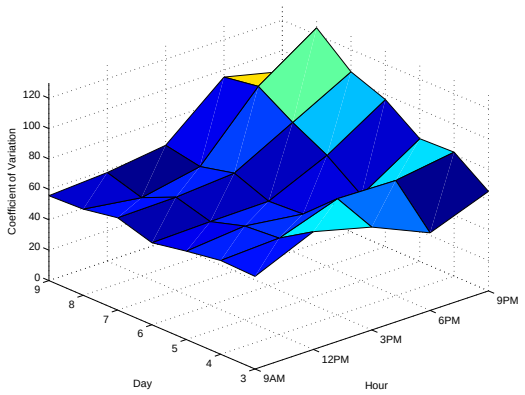

Figure 1. Coefficient of variation over light measurements.

Our experiments on this data set involve predicting the average value and the maximum value in the sensor deployment using measurements from just 5 sensors ($k = 5$).

The data from March 1st and 2nd were used for training instances, and the remaining seven days (March 3rd to 9th) provided the test instances. Based on the data from the first two days, a distance metric was learned which was then used for estimating both the maximum and the average.

4.3.1 Predicting the Average

We first present results for prediction of the *average* value. We run our k-median approximation algorithm on this distance metric to choose a subset S of 5 sensors that will subsequently be used for prediction, i.e., we compute an approximate 5-median solution.

For each day, our algorithms use only the readings from the sensors in S to estimate the average light value at all the sensors. We evaluate the performance at 3-hour intervals from 9AM until 9PM. Since sensor readings were not always available at the precise same time for all locations, and to reduce the impact of noise, we perform comparisons against the average over a 10-minute window centered around the time of prediction.

In Figure 2, we plot the percentage error in prediction of the average light measurement using our algorithm, for all 7 days and times from 9AM until 9PM. Our prediction error varies from 0.3% to 31%, with an average prediction error of 10.1%. Notice that this constitutes a seven-fold reduction compared to the standard deviation value.

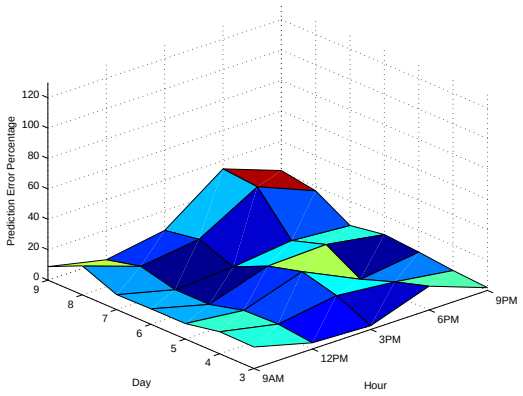

Figure 2. Predicting the Average: Prediction error using the k-median algorithm.

We next compare the prediction error against that obtained by using 20 different random subsets of size 5 on each day of the test instances. We report the minimum and the mean of the errors obtained by the random subsets. Figure 3 plots the prediction error using our k-median based algorithm versus those obtained by the "best" random selection (corresponding to the minimum error over all the trials) and a "mean" random selection (corresponding to the mean error). Over the seven days of test data, the average prediction error using the subset S (10.1%) significantly outperforms the "mean" random selection and is even better than the "best" random selection (18.8%) using 20 trials in each day. Note that for the random selection comparisons, the randomly chosen subsets are allowed to change from day to day (as opposed to our algorithm's use of the same subset S for all the days).

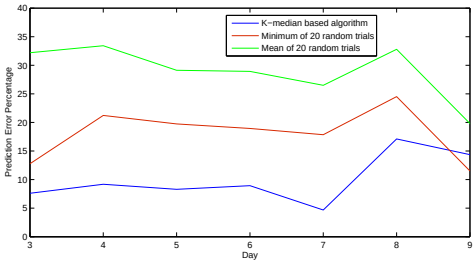

Figure 3. Predicting the Average: Comparison of the k-median algorithm against the minimum and average of 20 random trials.

In separate experiments, we pick random subsets of 5, 10 or 15 nodes and use them to predict the average over all

the days of the test data. We report the minimum and mean prediction error over 50 repetitions of this experiment. For sets of 5 nodes, the mean prediction error over 50 random iterations is 18.1%, while the minimum prediction error is 12.7%. Notice that the best such set can only be determined in retrospect, yet the k-median algorithm outperforms the best fixed set over 50 random trials. For 10 nodes, the mean prediction error is around 13.5%, while the minimum error is 10.1%. It took 15 sensors for the average prediction error over 50 iterations to exceed our algorithm's performance. Our results show that the use of the k-median algorithm gives almost a threefold saving in the number of sensors that need to be sampled, when compared against the results of random sampling. Table 1 summarizes these results.

Scheme	Avg Error	Minimum Error
5-median	10.1	-
5 random samples	18.1	12.7
10 random samples	13.5	10.1
15 random samples	9.7	7.7

Table 1. Predicting the Average: Comparison of the k-median algorithm against random trials with larger subsets.

4.3.2 Predicting the Maximum

Next, we look at results for predicting the *maximum* value among all sensors, using a subset of just 5 sensors. Using the greedy k-center approximation algorithm with $k = 5$, we choose a subset S of 5 sensors to be subsequently used for prediction, i.e., a 5-center solution.

The prediction error over the seven days of test data varies from 0% to 59%, with an average prediction error of 14.9%. As before, we compare this performance against the minimum and mean error from 20 random sets of 5 nodes chosen each day. Figure 4 plots the prediction errors for our k-center algorithm versus the random trials for all 7 days. Our algorithm significantly outperforms the error corresponding to the mean of the random trials, and most of the time also does better than the minimum of the random trials for each day.

As in the case of estimating the average, we also compare our algorithm against 50 trials of a randomly chosen set of 5, 10, or 15 sensors that is kept constant over all days. As earlier, the mean and minimum prediction error over 50 random trials of 5 sensors is worse than that of our algorithm, and it takes 15 randomly chosen sensors to improve on the performance of the 5-center solution. Table 2 summarizes these results.

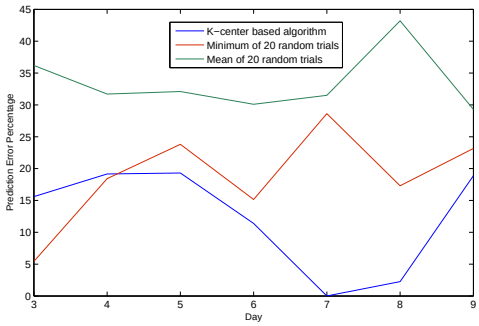

Figure 4. Predicting the Maximum: Comparison of the k-center algorithm against the minimum and average of 20 random trials.

Scheme	Avg Error	Minimum Error
5-center	14.9	-
5 random samples	23.7	17.2
10 random samples	19.8	15.1
15 random samples	14.7	10.7

Table 2. Predicting the Maximum: Comparison of the k-center algorithm against random trials with larger sensor sets.

4.4 Humidity Measurements

In a third set of experiments, we use our algorithms on humidity measurements (measured in percentage) taken from the SensorScope sensor deployment, an outdoor environment. The data were recorded over a period of 4 days from October 1st to October 4th, 2007, at intervals of a few minutes. We cleaned the data by using a median filter to remove outliers.

Figure 1 plots the coefficient of variation (the ratio of standard deviation to the mean) for the humidity measurements over different days and times. The standard deviation varies between 4.2% and 18% of the mean, with an average standard deviation of 10.4% of the mean. Thus, the outdoor humidity data have intrinsically less spatial variance than the indoor light data. (The other measured quantities in this data set varied even less.)

Our algorithm uses data from October 1st as training data to construct the empirical distance metric. Measurements over the remaining three days were used as test cases.

Based on the estimated metric, we run the k-median approximation algorithm with $k = 3$ to choose a subset S of 3 sensors. That subset is then used to predict the average humidity for the remaining three days. As before, we evaluate the prediction at 3-hour intervals, and smoothen out the

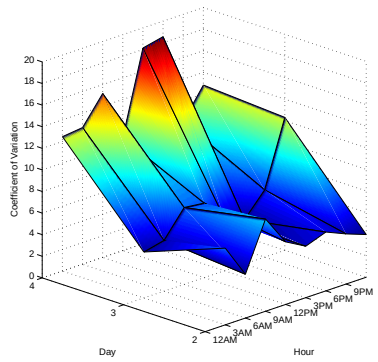

Figure 5. Coefficient of variation over humidity measurements.

measured values over 10-minute windows.

Figure 6 shows the prediction error (in percentage) in estimating the average over the three days at 3-hour intervals. The error varies from 0.34% to 7%, with an average prediction error of 2.2%, roughly a five-fold improvement over the standard deviation.

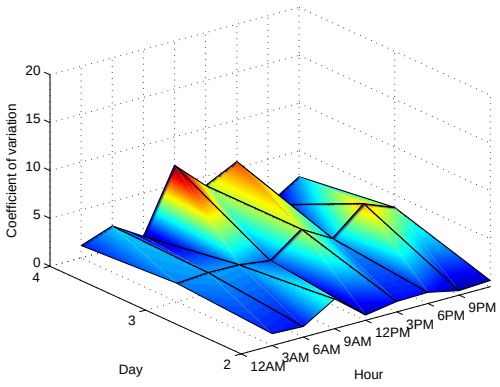

Figure 6. Predicting the Average: Prediction error using the k-median algorithm.

As before, we compare the prediction error against the best and average error over 50 trials of randomly chosen sets of 3, 6, or 9 sensors. Once chosen, the set is used to predict the average humidity over all three days of test instances. The mean prediction error of 50 random iterations choosing 3 sensors is 2.6%, and the minimum is 1.7%. For 6 sensors, these values improve to 2.0% and 1.3%, respectively, and for 9 random samples to 1.1% and 0.47%. The results are summarized in Table 3.

In this case, the performance of the k-median algorithm is not significantly better than that of a randomly chosen set of the same size. The reason partly lies in the low spatial variance of the data, and partly in the high temporal variance. Simply by the law of large numbers, after sufficiently many trials, the error of random sets will likely be small; due to the small spatial variance, the number of such trials needed is not very high. More importantly, the high temporal variance evident in Figure 5 suggests that the distance learned from the first day of data may not be most appropriate for later days.

Scheme	Avg Error	Minimum Error
3-median	2.2	-
3 random samples	2.6	1.7
6 random samples	2.0	1.3
9 random samples	1.1	0.47

Table 3. Predicting the Average: Comparison of the k-median algorithm against random trials with larger sensor sets

5 Conclusions

We have presented an optimization formulation for finding the best set of sensors to predict an aggregate function. We assumed worst-case behavior of the actual values subject to hard, metric-induced constraints.This led to proving an equivalence with selecting a k-median resp. k-center solution. Our experiments on real-world data showed that these approaches can be expected to choose good sets for prediction in practice.

Many interesting questions remain for future work. For instance, the behavior of sensors at distance d need not be symmetric. It may be known that sensor i's value is always at least as large as sensor j's, and they differ by at most $d(i, j)$. This leads to an asymmetric notion of distance. In future work, we intend to investigate whether good approximation algorithms exist for this asymmetric distance notion.

The general framework we provide in Section 3.3 gives a characterization of the adversary's behavior. However, it does not guarantee the existence of a good approximation algorithm to optimize the particular objective function at hand. Extending the known algorithms for k-median and k-center to different coverage problems would be very interesting. Indeed, this question is meaningful even when the adversary will not reveal identical values at all locations. In particular, it would be intriguing how well one can approximate the *median* of all sensor values.

Much like standard adversarial analysis (e.g., in online algorithms), the adversary in our model is very powerful, in that he knows all choices that will be made by the algorithm. A standard way to guard against such worst-case behavior

is via randomness, and the assumption that the adversary knows the randomized algorithm, but not the outcome of the algorithm's coin flips. Devising a randomized algorithm for selecting a good subset S to sample in this framework is quite intriguing.

For example, if the sensors are evenly (and very densely) spaced on a line, and we want to estimate the average by sampling just one sensor, then the best deterministic algorithm should choose the midpoint of the line. However, choosing the point $\frac{1}{3}$ half the time, and $\frac{2}{3}$ the other half results in an expected error only $2/3$ of the deterministic strategy. In ongoing work, we are investigating the best possible distribution to use for sampling sets, on a line or in more general settings. Even for a line, the problem is non-trivial; the maximum improvement possible over the deterministic strategy appears to lie strictly between $\frac{4}{7}$ and $\frac{1}{2}$.

Acknowledgments

We would like to thank Debojyoti Dutta, Alon Efrat and Bhaskar Krishnamachari for useful discussions, and anonymous reviewers for helpful feedback.

References

[1] Intel Berkeley Lab Data. http://db.csail.mit.edu/labdata/labdata.html.

[2] The SensorScope Project. http://sensorscope.epfl.ch.

[3] Z. Abrams, A. Goel, and S. Plotkin. Set k-cover algorithms for energy efficient monitoring in wireless sensor networks. In *Proc. 3rd Intl. Symp. on Information Processing in Sensor Networks*, 2004.

[4] K. Anstreicher, M. Fampa, J. Lee, and J. Williams. Maximum-entropy remote sampling. *Discrete Applied Mathematics*, 108(3):211–226, 2001.

[5] V. Arya, N. Garg, R. Khandekar, A. Meyerson, K. Munagala, and V. Pandit. Local search heuristics for k-median and facility location problems. In *Proc. 33rd ACM Symp. on Theory of Computing*, 2001.

[6] B. Bash, J. Byers, and J. Considine. Approximately uniform random sampling in sensor networks. In *Proc. 1st Intl. Workshop on Data Management for Sensor Networks*, pages 32–39, 2004.

[7] F. Bian, D. Kempe, and R. Govindan. Utility-based sensor selection. In *Proc. 5th Intl. Symp. on Information Processing in Sensor Networks*, pages 11–18, 2006.

[8] J. Byers and G. Nasser. Utility-based decision-making in wireless sensor networks. In *Proc. 1st ACM Symp. on Mobile Ad Hoc Networking and Computing*, pages 143–144, 2000.

[9] M. Charikar and S. Guha. Improved combinatorial algorithms for facility location problems. *SIAM J. on Computing*, 34:803–824, 2005.

[10] M. Charikar, S. Guha, D. Shmoys, and E. Tardos. A constant-factor approximation algorithm for the k-median problem. *Journal of Computer and System Sciences*, 65:129–149, 2002.

[11] A. Das and D. Kempe. Algorithms for subset selection in linear regression. In *ACM Symposium on Theory of Computing*, 2008.

[12] A. Deshpande, C. Guestrin, S. Madden, J. Hellerstein, and W. Hong. Model driven data acquisition in sensor networks. In *Proc. 30th Intl. Conf. on Very Large Data Bases*, 2004.

[13] S. Ganeriwal, C. Han, and M. Srivastava. Going beyond spatial aggregates in sensor networks. Technical report, NESL, 2004.

[14] C. Guestrin, A. Krause, and A. Singh. Near-optimal sensor placements in gaussian processes. In *Proc. 22nd Intl. Conf. on Machine Learning*, pages 265–272, 2005.

[15] H. Gupta, V. Navda, S. Das, and V. Chowdhary. Efficient gathering of correlated data in sensor networks. In *Proc. 6th ACM Symp. on Mobile Ad Hoc Networking and Computing*, 2005.

[16] D. Hochbaum and D. Shmoys. A best possible heuristic for the k-center problem. *Mathematics of Operations Research*, 10:180–184, 1985.

[17] W. Hsu and G. Nemhauser. Easy and hard bottleneck location problems. *Discrete Applied Mathematics*, 1:209–216, 1979.

[18] K. Jain, M. Mahdian, and A. Saberi. A new greedy approach for facility location problem. In *Proc. 34th ACM Symp. on Theory of Computing*, pages 731–740, 2002.

[19] K. Jain and V. Vazirani. Primal-dual approximation algorithms for metric facility location and k-median problems. In *Proc. 40th IEEE Symp. on Foundations of Computer Science*, pages 2–13, 1999.

[20] R. Johnson and D. Wichern. *Applied Multivariate Statistical Analysis*. Prentice Hall, 2002.

[21] C.-W. Ko, J. Lee, and M. Queyranne. An exact algorithm for maximum entropy sampling. *Operations Research*, 43(4):684–691, 1995.

[22] P. Liaskovitis and C. Schurgers. Leveraging redundancy in sampling-interpolation applications for sensor networks. In *Proc. 3rd Intl. Conf. on Distributed Computing in Sensor Systems*, 2007.

[23] S. Meguerdichian, F. Koushanfar, M. Potkonjak, and M. Srivastava. Coverage problems in wireless ad-hoc sensor networks. In *Proc. 20th IEEE INFOCOM Conference*, pages 1380–1387, 2001.

[24] A. Miller. *Subset Selection in Regression*. Chapman and Hall, second edition, 2002.

[25] S. Pattem, B. Krishnamachari, and R. Govindan. The impact of spatial correlation on routing with compression in wireless sensor networks. In *Proc. 3rd Intl. Symp. on Information Processing in Sensor Networks*, 2004.

[26] J. Tropp. *Topics in Sparse Approximation*. PhD thesis, University of Texas, Austin, 2004.

[27] V. Vazirani. *Approximation Algorithms*. Springer, 2001.

[28] Z. Xiong, A. Liveris, and S. Cheng. Distributed source coding for sensor networks. In *IEEE Signal Processing Magazine*, 1998.

[29] S. Yoon and C. Shahabi. The clustered aggregation (cag) technique leveraging spatial and temporal correlations in wireless sensor networks. *ACM Transactions on Sensor Networks*, 3(1), 2007.