A Distance-Based Scheme for Reducing Bandwidth in Distributed Geometric Monitoring

Yuval Alfassi
University of Haifa
Haifa, Israel
yalfas02@campus.haifa.ac.il

Moshe Gabel
University of Toronto
Toronto, Canada
mgabel@cs.toronto.edu

Gal Yehuda
Technion – Israel Institute of Technology
Haifa, Israel
ygal@cs.technion.ac.il

Daniel Keren
University of Haifa
Haifa, Israel
dkeren@cs.haifa.ac.il

***Abstract*—Tracking the value of a function computed from a dynamic, distributed data stream is a challenging problem with many real-world applications. Continuously forwarding data updates can be costly, yet complex functions are difficult to evaluate when data is not centralized. One general approach to continuous distributed monitoring is the Geometric Monitoring (GM) family of techniques. GM reduces the functional monitoring problem to a set of local constraints that each node checks locally, and uses a simple protocol to update those constraints as needed.**

While most work on GM focuses on reducing the number of messages exchanged by the common GM protocol, with one recent notable exception, there has been little attention to reducing the size of those messages, which impacts bandwidth.

We propose the Distance Scheme: a novel bandwidth-efficient variation of the GM protocol that reduces the size of most monitoring messages in GM to a single scalar, and is compatible with the large body of prior work on GM. We apply it to monitor three different functions using three real-world datasets, and show it substantially reduces bandwidth while requiring fewer messages to be transmitted than the current state-of-the-art approach. We further describe a value-based scheme that, while typically outperformed by the Distance Scheme, is simpler to apply, matches state-of-the-art bandwidth performance with fewer messages, and is also compatible with existing work.

I. Introduction

Consider the setting of distributed monitoring [1], where one tries to approximate a function defined over the aggregation of multiple distributed data streams. Computing the function locally over each stream does not always provide insight for the approximation over the global aggregate, yet forwarding all the data to a single location can be prohibitive due to high bandwidth or power consumption.

Common approaches to this problem include crafting individual distributed protocols tailored to the specific monitored function, or *sketching*, which reduces the size of transmitted data, but still relies on data centralization.

Geometric monitoring (GM), introduced in [2] is a family of techniques that share the same distributed protocol and underlying principles [3]. GM reduces the global function's approximation bounds to a set of local constraints on each data stream, called *safe zones*; as long as the constraints are satisfied, no communication is required. Unlike the previous function-specific approaches (which were, in addition, typically restricted to certain classes of functions – monotonic, linear, or convex), GM techniques are generic: they can and have been applied to monitor a large class of non-trivial functions, including regression [4], Pearson correlation [5], spectral gap [6], cosine-similarity [5], effective dimension [5], and even sketches [7], [8]. Moreover, GM is modular: different techniques and monitored functions can be composed together to achieve dramatic reduction in communication, since they share the same underlying distributed protocol [3].

While GM is very effective in reducing the number of sent messages, current techniques typically require high bandwidth whenever the local GM constraint is violated, as the resulting *violation resolution* phase requires sending large data vectors between nodes, which can offset the bandwidth-savings effects of the other GM modules, especially for high dimensional data.

A recent insightful approach, *functional geometric monitoring* (FGM) [9], addresses this issue by replacing the geometric monitoring protocol with a distributed counting protocol, which only requires sending scalars as opposed to vectors. This approach is effective in limiting bandwidth and allows theoretically bounding the communication cost. However, it loses some of GM's modularity, since much of the existing work has been designed for the GM protocol. Also, the fact that it relies solely on the function's value results in a certain bias towards false alarms, especially when the monitored function changes quickly. We elaborate on this in Section III-D, and provide experimental evidence for this problem in Section V. Also, as we later show, while the distributed counting protocol is bandwidth efficient, it does result in more message exchanges than GM. In some settings the number of messages can be as important as bandwidth, for example when initiating or waiting for a wireless transmission requires substantial battery power [10], [11].

Our Contributions: We propose an adaptation of the GM protocol that can resolve local constraint violations while sending single scalars, thus dramatically reducing bandwidth consumption during the violation resolution phase. Further, it is compatible with existing GM extensions, such as reference point prediction [12] and slack distribution protocols [7], [13]. The main contributions of this work:

- We describe a novel geometric result (the Distance Lemma) that relates the average of vectors to the sum of their (generalized) distance from the boundary of a convex set.
- We propose two low-bandwidth schemes for distributed functional monitoring that are compatible with existing GM approaches. The *Distance Scheme*, our main contribution,

improves the state-of-the-art in bandwidth reduction, and scales well with the number of nodes. The other approach is the *Value Scheme*, a low-bandwidth scheme based on [5]. The Value Scheme is easier to use than the Distance Scheme and achieves similar performance to current SotA, FGM [9], but is outperformed by the Distance Scheme for most nonlinear functions.

- We provide an empirical evaluation for three functions on three real-world datasets, and show that the Distance Scheme requires up to $\times 2.5$ less bandwidth than the current SotA [9], while transmitting up to $\times 5$ fewer messages.

II. Background

In this section we review the background and important technical details of prior work.

The *continuous distributed monitoring model* [1] deals with the following setup: there are k distributed nodes (sensors, computing nodes etc.), denoted $S_1 \dots S_k$, and a coordinator node C with which all nodes communicate. Each node S_i receives an unbounded stream of data, from which it locally computes at time t a *local vector* $v_i(t) \in \mathbb{R}^d$. Let the *global vector* $v(t)$ be the (possibly weighted) average of all $v_i(t)$. Note that the vectors $v(t), v_i(t)$ are dynamic: they change over time. However, for clarity, we suppress the time variable (t) except when required.

We study the problem of *distributed functional monitoring* w.r.t the value of a function f over the aggregate of all local vectors v_i. In other words, we wish to maintain an approximation of $f(v)$, while minimizing communication[1].

Geometric Monitoring (GM) is a family of related techniques for communication-efficient distributed monitoring of general functions[2]. These techniques were developed in a series of papers, including [2], [3], [14], [15], and applied to many problems, such as least squares models [4], entropy [7], skyline computations [16], inner products, effective dimension and anomaly detection [5], processing of evolving distributed graphs [6], and monitoring machine learning models [17].

A. Geometric Monitoring Framework

We briefly survey the principles and definitions behind GM. See [5] for a more complete survey of GM techniques.

As its name suggests, GM casts the monitoring problem in a geometric setting, relying heavily on the notion of convexity. Given the task of approximating a function f, the GM framework defines a distributed protocol that reduces it to monitoring threshold crossings. We next describe the GM protocol, which is summarized in Algorithm 1.

1) Fundamentals: Let $t = 0$ be some initial time, referred to as *sync time*, and let $v_i(0)$ and $v(0)$ be the values of the local and global vectors at that time; $v(0)$ is assumed to be known to all S_i's and will be referred to as the *public vector*. Given the initial value $f_0 = f(v(0))$ and the desired approximation (e.g., $1 \pm \epsilon$), we can maintain an approximation of f by monitoring the constraints $f(v) \leq (1+\epsilon) f_0$ and $f(v) \geq (1-\epsilon) f_0$. These two constraints are continuously monitored, and if one is violated, C must be alerted. When this happens, we must resolve the violation, for example by centralizing all data and updating f_0 and the constraints.

This allows us to monitor the value of f to arbitrary precision. Since we have complete freedom in defining v_i and f, it turns out that many important monitoring problems can be expressed using this formulation. We refer the reader to [5]–[7], [17] for some recent examples.

Given a *threshold constraint* $f(v) \leq T$ for some threshold T (for the reverse condition just reverse the signs), GM defines the *admissible region* $A = \{u \in \mathbb{R}^d \mid f(u) \leq T\}$, the set of vectors that satisfy the constraints. We also define a *safe zone* $Z \subseteq A$, a convex sub-set of A which contains $v(0)$. It may be defined either directly [15], or via functional inequalities [5]. Deriving the safe zone Z for a function f is a critical component of GM. Several techniques have been developed for this problem [5], [6], [14]; here we assume that Z was already computed by one of these approaches, and instead focus on reducing monitoring bandwidth. Following the previous work, we assume that Z is defined either directly as a convex subset of A, or as $\{x \mid g(x) \leq T\}$ for a suitable convex function g such that $g(x) \geq f(x)$.

Let $d_i = v_i - v_i(0)$ be the *drift vector* for node S_i, and define its *private vector* as the change from the public vector, $p_i = v(0) + d_i$. Note that p_i, v_i are known to S_i, but not to the other nodes nor the coordinator C. The average of private vectors is the current global vector v: $\frac{1}{k}\sum_{i=1}^{k}(v(0)+d_i) = \frac{1}{k}\sum_{i=1}^{k} v_i = v$. Moreover, at time $t = 0$ all private vectors are equal to the public vector $v(0)$ and are inside the safe zone Z. In addition, C assigns to S_i a *slack vector* λ_i, such that the sum of all slacks is zero: $\sum_{i=1}^{k} \lambda_i = 0$.

GM replaces monitoring the global constraint $v \in A$ with monitoring the following local constraints: $p_i + \lambda_i = v(0) + d_i + \lambda_i \in Z$, and note that $\frac{1}{k}\sum_{i=1}^{k} p_i + \lambda_i = \frac{1}{k}\sum_{i=1}^{k} v_i = v$. Since convex sets are closed under averaging, if $p_i + \lambda_i \in Z$ for every node i, then $v \in Z$ and therefore $v \in A$. Thus, if

Algorithm 1 The three phases of the common GM protocol.

1) **Eager sync:** the coordinator C collects v_i from all nodes. It computes $v(0)$, the resulting approximation bounds and equivalent thresholds T, and the corresponding safe zone Z. It then sends them to all nodes for monitoring (phase 2).
2) **Monitoring:** nodes update v_i and monitor $p_i + \lambda_i \in Z$. While there is no violation, nodes remain in this phase. If there is a local violation, report it to coordinator (phase 3).
3) **Lazy sync:** when a node reports a violation, C tries to resolve it by incrementally polling nodes and updating slacks λ_i. If the violation is resolved, send updated slacks to participating nodes and go back to monitoring (phase 2). If the violation is not resolved, go to eager sync (phase 1).

[1]Since f and v_i can be defined arbitrarily over the local data, this captures a wide variety of complicated, non-linear functions

[2]In this work, we refer by "GM" to the body of techniques that share the common principles and protocols as outlined in [3], rather than to the specific initial technique described in [2].

$p_i + \lambda_i \in Z$, the node S_i can remain silent; and as long as all nodes are silent, it is guaranteed that the global monitored condition is satisfied.

2) Violation Resolution: If a *local violation* occurs – that is, $p_i + \lambda_i \notin Z$ – the node alerts C, which enters a *violation resolution phase* to check whether the average of the vectors $\{p_i\}_{i=1}^k$ is still in Z. One way to do this is *eager sync*, which is also done during initialization. The coordinator polls all nodes for their current local vectors v_i, defines the current time as the sync time $t = 0$, and sends an updated $v(0)$, approximation thresholds, and the safe zone Z to all nodes; local slacks λ_i are reset to zero. This guarantees that $v(0)$ and $p_i + \lambda_i$ are in the safe zone, and monitoring can continue.

However, frequently only a small percentage of the private vectors exit Z, while v remains in Z, hence the monitored condition continues to hold; in other words, a local violation (at a node) typically does not indicate a *global violation* – where $v \notin Z$. Executing an eager sync for such *false alarms* will therefore incur an unnecessarily large bandwidth. Hence, GM first tries a *lazy sync* phase to resolve local violations with less communication, by building a *balancing set* L: a set of nodes S_i such that their private vectors can be balanced using slack variables. The coordinator polls nodes that have not reported violations for their private vectors, and adds them to the balancing set L. If $\frac{1}{|L|}\sum_{i\in L}(p_i + \lambda_i) \in Z$, then the violation has been resolved: the coordinator sends re-balanced slacks λ_i to all nodes in L such that $p_i + \lambda_i \in Z$ for all nodes in L while $\sum_{i\in L}\lambda_i$ remains unchanged, and the monitoring continues. We refer to [7, Section 4.6] for the full details of violation resolution.

III. Methods

In this section we introduce two novel monitoring schemes: the Distance Scheme and the Value Scheme.

Most of the effort in GM research addressed the goal of finding efficient local conditions. However, with the notable exception of the recent [9] (which will be discussed in further detail in Section VI), when a local violation occurs, nodes – not just the violating ones, but all members of the balancing set – transmit their entire local data vectors to the coordinator. Thus while GM offers a widely-applicable solution for reducing the *number* of messages, their *size* may still be large, as it is proportional to the dimension d of the local vectors. Here, we address this problem, and study two schemes – the Distance Scheme and the Value Scheme – to the problem; these schemes are compatible with the large body of existing techniques that use the GM framework.

We next describe the Distance Scheme, which rests on a geometric result that allows to modify the violation resolution and lazy sync phases in GM, and replaces transmission of vectors with transmission of scalars. Then, we will discuss a value-based scheme, and compare it with the Distance Scheme. We will mathematically demonstrate that the Value Scheme is *inherently biased towards false alarms*, as opposed to the Distance Scheme, and that this bias becomes more evident as the monitored functions changes more rapidly. This analysis is backed by extensive experiments (Section V).

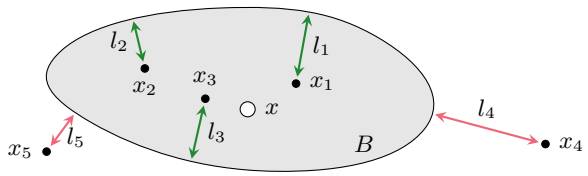


Fig. 1. Illustration of the Distance Lemma: if the sum of the signed distances l_i from vectors $x_1, \ldots, x_5$ to the closed convex set B is non-positive then their average x is inside B. Formally, $\sum_{i=1}^5 l_i \leq 0 \implies \frac{1}{5}\sum_{i=1}^5 x_i \in B$. x_4 is not in B and further away from the boundary than all other vectors, but is still balanced by the points inside B.

A. Distance Lemma

To resolve a local violation, it is necessary to determine whether the global vector v is still in Z, even if $p_i + \lambda_i \notin Z$ for some values of i; our goal is to achieve this while not communicating the entire local vector. The following Distance Lemma allows just that. Plainly put, the Lemma states that if the sum of absolute distances from the boundary of the points inside a convex set is larger than the corresponding sum for the points outside the same set, then the average of all points is inside the convex set (Fig. 1). Applying it enables, in some cases, to resolve a local violation (i.e. to conclude that $v \in Z$), while communicating a *single scalar* per node. The proof rests on representing each point outside Z as a combination of a boundary point and an "overflow" vector outside Z. We then balance the sum of the overflows with the distance to the boundary of vectors inside Z.

Lemma 1 (Distance Lemma)**.** *Let B be a closed convex set in a normed linear space, with the distance function induced by the norm (i.e the distance between vectors x, y is $||x-y||$). Let $x_1, \ldots, x_\ell$ be vectors inside B, and $x_{\ell+1}, \ldots, x_k$ vectors outside B. Denote the* signed distance *of x_i from the boundary of B by l_i (negative if $x_i \in B$, positive otherwise). Assume that $\sum_{i=1}^k l_i \leq 0$. Then the average of $x_1, \ldots, x_k$ is inside B.*

Proof. Since a convex set is closed under averaging, it suffices to produce a set of k vectors in B such that their sum is equal to $\sum_{i=1}^k x_i$.

For the vectors outside B ($\ell < i \leq k$), denote by c_i the point on the boundary of B which is closest to x_i, let $u_i = x_i - c_i$ be the vector from c_i to x_i, and note $|l_i| = ||u_i||$. Then:

$$\begin{aligned} &x_1 + \ldots + x_k \\ &= x_1 + \ldots + x_\ell + c_{\ell+1} + \ldots + c_k + (u_{\ell+1} + \ldots + u_k) \ . \end{aligned}$$

Next, denote $u = u_{\ell+1} + \ldots + u_k$ and $\alpha_i = \dfrac{|l_i|}{|l_1| + \ldots + |l_\ell|}$, and note $\sum_{i=1}^\ell \alpha_i = 1$. We'll decompose u to a weighted sum of ℓ vectors:

$$\begin{aligned} &x_1 + \ldots + x_k \\ &= x_1 + \ldots + x_\ell + c_{\ell+1} + \ldots + c_k + u \\ &= x_1 + \ldots + x_\ell + c_{\ell+1} + \ldots + c_k + (\alpha_1 + \ldots + \alpha_\ell) \cdot u \\ &= (x_1 + \alpha_1 \cdot u) + \ldots + (x_\ell + \alpha_\ell \cdot u) + c_{\ell+1} + \ldots + c_k \ . \end{aligned}$$

By definition, $c_{\ell+1} \dots c_k$ are inside B, so we only have to show that $x_i + \alpha_i \cdot u \in B$ for $i \in \{1 \dots \ell\}$.

Note that since $l_j \leq 0$ for $j \in \{\ell+1 \dots k\}$, and by the Lemma's assumption $\sum_{i=1}^{k} l_i \leq 0$, it holds that $|l_{\ell+1}| + \ldots + |l_k| \leq |l_1| + \ldots + |l_\ell|$. From the triangle inequality:

$$\begin{aligned} \|u\| &\leq \|u_{\ell+1}\| + \ldots + \|u_k\| \\ &= |l_{\ell+1}| + \ldots + |l_k| \leq |l_1| + \ldots + |l_\ell| \ , \end{aligned}$$

therefore, for $i \in \{1 \dots \ell\}$ it holds that:

$$\begin{aligned} \|\alpha_i \cdot u\| &= \frac{|l_i|}{|l_1| + \ldots + |l_\ell|} \|u\| \\ &\leq \frac{|l_i|}{|l_1| + \ldots + |l_\ell|} (|l_1| + \ldots + |l_\ell|) = |l_i| \ . \end{aligned}$$

Hence, since $x_i \in B$ and $|l_i|$ is the distance from x_i to the boundary of B, it follows that $x_i + \alpha_i \cdot u \in B$ for $i = 1 \dots \ell$, thus completing the set of k vectors in B. □

Note that this result is true for *any norm*, not just the standard Euclidean one; this will turn out to be useful, as choosing a suitable norm can further reduce communication (Section IV-B3).

It is quite possible that the average of the x_i is inside B, while the sum of the distances is positive (that is, the condition is not "if and only if"); still, in practice, as supported by the experimental results (Section V), the Distance Lemma substantially reduces communication overhead.

B. The Distance Scheme

Armed with the Distance Lemma, we now describe a modification to the GM monitoring protocol (Alg. 1 in Section II).

Rather than monitoring the original local constraint $p_i + \lambda_i \in Z$, each node computes l_i, the distance of its private vector p_i from the boundary of the safe zone Z. It then monitors the local scalar constraint $l_i + \lambda_i \leq 0$, where λ_i is a *scalar* slack variable assigned by the coordinator such that $\sum_{i=1}^{k} \lambda_i = 0$ (Sec. II-A1). If all local constraints are maintained, the Distance Lemma guarantees that $v \in Z \subseteq A$ and therefore the global condition is satisfied.

If a violation occurs, the nodes and coordinator proceed to the usual violation resolution phase (Section II-A2), except that during lazy sync we exchange scalar distances l_i and scalar slacks λ_i rather than the entire vectors. Thus violations can be resolved *while communicating only a very small volume of information*. As before, if the coordinator cannot resolve the violation by reassigning slacks such that $\sum_{i \in L}(l_i + \lambda_i) \leq 0$ while preserving their sum (zero), it switches to a standard eager sync where the entire local vectors are exchanged.

Note this is a variant of the original GM monitoring protocol: the only changes are in the constraint used for monitoring and lazy sync, and that slacks are now scalars rather than vectors. Therefore, the Distance Scheme is compatible with the large body of existing work on applying the GM protocol (and its extensions) to monitor a wide class of important functions.

To apply the Distance Lemma, it is necessary to compute, or bound, the distance of p_i from the boundary of the safe zone Z (recall that the distance can be defined with respect to any norm). We address this problem in Section IV.

C. The Value Scheme

For cases where computing the distance to the boundary of the safe zone is difficult, we now describe the Value Scheme: an easy-to-apply scheme with a similar protocol that is widely applicable and still bandwidth-efficient.

The Value Scheme is based on the *Convex Bound* (CB hereafter) variant of GM [5]. Briefly, CB uses the following basic property of convex functions $c()$:

$$c\left(\frac{x_1 + \ldots + x_k}{k}\right) \leq \frac{c(x_1) + \ldots + c(x_k)}{k} \tag{1}$$

This allows to very easily monitor the condition $c(v) \leq T$: the nodes locally check whether $c(p_i + \lambda_i) \leq T$, and remain silent as long as this condition holds. To extend this to general functions $f()$, CB tries to construct a convex function $c()$ such that $c(x) \geq f(x)$ for every x; then, one simply monitors the condition $c(v) \leq T$. CB is closely related to the convex safe zone approach: the equivalent safe zone is $Z = \{x \mid c(x) \leq T\}$, whose convexity immediately follows from convexity of $c()$. In [5], a few methods for constructing CB's are studied, and it is proved that they exist for a wide family of functions $f()$. Here we assume that the CB is already given. The equivalent notion for monitoring a lower bound is a lower concave bound.

Given a convex bound $c()$, the Value Scheme proceeds as follows. Each node monitors the local constraint $c(p_i) + \lambda_i \leq T$, where λ_i are scalar slacks that sum to zero. If all the local conditions hold, the global one holds as well:

$$\begin{aligned} c(v) &= c\left(\frac{1}{k}\sum_{i=1}^{k} v_i\right) = c\left(\frac{1}{k}\sum_{i=1}^{k} p_i\right) \leq \frac{1}{k}\sum_{i=1}^{k} c(p_i) \\ &= \frac{1}{k}\sum_{i=1}^{k} (c(p_i) + \lambda_i) \leq \frac{1}{k}(k \cdot T) = T \ . \end{aligned} \tag{2}$$

In case of a local violation at node S_i, the node sends $c(p_i)$ to the coordinator, which attempts to balance it with a set of balancing nodes using lazy sync (Sec. II-A2) by finding a subset of nodes L such that $\frac{1}{|L|}\sum_{i \in L}(c(p_i) + \lambda_i) \leq T$ and $\sum_{i=1}^{k} \lambda_i = 0$. As before, if lazy sync fails, the coordinator switches to eager sync.

D. Comparison of Value and Distance Schemes

The Distance and Value schemes share a common feature: both use scalar quantities in order to resolve local violations. For the Value Scheme (and also for the method in [9]), it is the value of a convex bound; for the Distance Scheme, it is the signed distance from Z's boundary. While these two schemes are related, they can yield rather different performance.

We start with a simple but illustrative examples of a convex function (recall that the monitoring is applied not directly to the function $f()$, but its convex bound $c()$). Fig. 2 (left) depicts an example in which the *function level sets* behave very differently from the *distance level sets*. Assume there are two nodes,

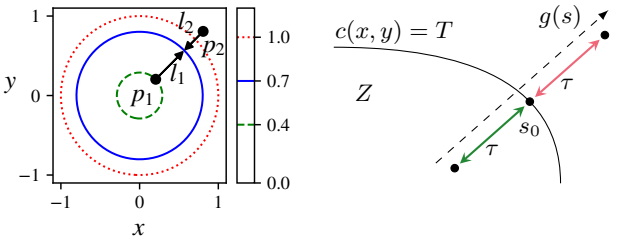


Fig. 2. **Left:** level sets for the function $c(x, y) = \exp(x^2 + y^2 - 1)$. While the Value Scheme will declare a violation, the Distance Scheme will not, since p_1 is farther away from the boundary of the safe zone (solid blue circle) than p_2 (i.e., $|l_1| > |l_2|$). **Right:** schematic sketch for the analysis of the Distance Scheme vs. the Value Scheme.

and we are monitoring the constraint $c(x, y) \leq T = 0.7$, hence Z is the closed disk bounded by the solid blue circle; further assume that p_1 lies on the dashed green circle (meaning $c(p_1) = 0.4$), and p_2 is on the exterior of the dotted red circle (meaning $c(p_2) > 1$), and both slacks $\lambda_1 = \lambda_2 = 0$. This implies the node holding p_2 will report a violation in both Distance and Value schemes. The Value Scheme is unable to resolve this violation lazily: it will compute the average of $c(p_1), c(p_2)$, which is higher than T, and no slack assignment to λ_1 and λ_2 can resolve this violation, since slacks must sum to 0. Conversely, the Distance Scheme will resolve the violation lazily, as the distance of p_1 to Z's boundary is far larger than p_2's, hence the sum of signed distances is below 0. The opposite cannot happen: it is impossible that the convex bound $c()$ increases so slowly when moving away from Z's boundary that the Distance Scheme reports a violation, while the Value Scheme does not. This is due to the fact a convex function *cannot* grow at a sub-linear rate.

The following analysis demonstrates that compared to the Distance Scheme, the Value Scheme is *inherently more biased towards false alarms*. We compare the two approaches in the case in which the Distance Scheme "breaks even", i.e., there are two points, inside and outside Z, with equal distance τ from Z's boundary (Fig. 2, right). We show that in this case the average of the values of $c()$ at the two points will be greater than the threshold T, hence the Value Scheme will raise an alarm. For the sake of clarity we show this for dimension $d = 2$, but the argument holds for every d.

Let us look at the restriction of $c(x, y)$ to the line connecting the two points, and call it $g(s)$. Since it is the restriction of a convex function, it is also convex. So the question reduces to estimating the value of the average $\frac{g(s_0+\tau)+g(s_0-\tau)}{2}$ relative to $g(s_0) = T$. But it follows from Taylor's theorem that

$$\frac{g(s_0 + \tau) + g(s_0 - \tau)}{2} - T = \frac{\tau^2}{2} \left(g''(c_1) + g''(c_2)\right) \quad (3)$$

for some points c_1 resp. c_2 in $(s_0 - \tau, s_0)$ resp. $(s_0, s_0 + \tau)$. Since $g(s)$ is convex, $g''() \geq 0$ everywhere, hence the right-hand side of Eq. (3) is larger than zero, implying the average is larger than T. Therefore the Value Scheme will not be able to resolve this violation, while the Distance Scheme is successful.

It is possible to relate the average value of $g''()$ with the eigenvalues of the Hessian of $c()$, allowing to estimate the average "overshoot" of T. Intuitively, the more curved $c()$ is, the larger this overshoot will be, thereby increasing the advantage of the Distance Scheme over the Value Scheme.

IV. Applications

We now show how to apply the Distance Scheme to monitor the values of three oft-used functions: inner product, AMS F_2 moment frequency sketch [18], and an entropy sketch [19]. For each monitored function f we first define its safe zones using two auxiliary functions [5], which make distributed monitoring possible: (i) a *convex upper bound* $\overrightarrow{c}(v) \geq f(v)$ that defines an *upper safe zone*: $\overrightarrow{c}(v) < T \implies f(v) < T$; and (ii) a *concave lower bound* $\overleftarrow{c}(v) \leq f(v)$ that defines a *lower safe zone*: $\overleftarrow{c}(v) > T \implies f(v) > T$.[3] The Distance Scheme is applied to these safe zones. That is, by "distance from the safe zone" we mean distance from the level set $\overrightarrow{c}(x) = T$ (or $\overleftarrow{c}(x) = T$).

A. Inner Product

The inner product of two vectors is a fundamental function; it is extensively used in numerous applications, either as a similarity measure of data or as a basic component of other functions in data mining and machine learning applications.

We treat the input of the inner product as a vector of twice the dimension of the source data. Denote $v = [u, w]$ as the concatenation of the vectors u and w of size $\frac{d}{2}$ each. The inner product function is defined as follows:

$$F_{inner}([u, w]) = \langle u, w \rangle \ . \quad (4)$$

For $v = [u, w]$ and the public vector $v(0) = [u_0, w_0]$, the convex upper bound is:

$$\overrightarrow{c}(v) = \frac{1}{4}(\|u+w\|^2 - \|u_0-w_0\|^2 - 2\langle u_0-w_0, u-u_0-w+w_0\rangle)$$

and the concave lower bound is:

$$\overleftarrow{c}(v) = \frac{1}{4}(\|u_0+w_0\|^2 + 2\langle u_0+w_0, u-u_0+w-w_0\rangle - \|u-w\|^2) \ .$$

For the derivation of these bounds, please see [5].

1) L_2 distance to the inner product safe zone: The L_2 distance from a vector to the safe zone of the inner product has a closed-form solution, obtained using Lagrange multipliers. We briefly present the solution for the L_2 distance to the upper bound; calculating the distance to the lower safe zone is similar and thus omitted.

To calculate the distance from any vector $[u, w]$ to the surface $\{[x, y] \mid x, y \in \mathbb{R}^{\frac{d}{2}}, \overrightarrow{c}([x, y]) = T\}$, first define the Lagrangian

$$G(x, y, \lambda) = \|x - u\|^2 + \|y - w\|^2 + \lambda(\overrightarrow{c}([x, y]) - T) \ .$$

Solving the equations $\frac{\partial G}{\partial x} = 0$, $\frac{\partial G}{\partial y} = 0$, $\frac{\partial G}{\partial \lambda} = 0$ yields the following cubic equation in λ:

$$(\lambda - 1)(\lambda + 2)^2\|u_0 - w_0\|^2 + (\lambda + 2)^2 \left(4T + 2\langle u_0 - w_0, u - w\rangle\right) = 4\|u + w\|^2 \ .$$

Solving this cubic equation and then extracting x, y allows to compute the L_2 distance to the upper safe zone.

[3] As noted in Sec. II-A, this is the a common way of defining safe zones.

B. AMS F_2 Sketch

The second frequency moment is typically used to estimate the variance in a streaming data environment [8], [18]. Let $f_1, ..., f_N$ be the frequencies of the N unique items of a stream; then its second moment is $F_2 = \sum f_i^2$.

In a seminal paper, Alon et al. [18] describe an efficient, unbiased estimator for the second frequency moment of a stream, known as the the AMS F_2 sketch. They define the sketch as follows: a four-wise independent hash function is sampled, which uniformly maps the stream's values to either $+1$ or -1. The estimator accumulates the sums of all the $+1$'s and -1 values. The *expected value* of the sum squared is equal to the second moment value of the stream. Replicating the estimator in a table M of size $m \times q$ with different hash functions for each entry yields an estimate to the F_2 function with accuracy $q^{-\frac{1}{2}}$ and confidence 2^{-m}, where the estimated F_2 value is the median of the averages of norms squared of the table's rows. This collection is commonly referred to as the F_2 *sketch table*. Let $r_i = \frac{1}{\sqrt{q}}(M_{i1}, ..., M_{iq})$ denote the normalized vector of values at row i. The estimate for F_2 is:

$$F_{ams}(M) = \text{median}_{i=1}^{m} \|r_i\|^2 \ .$$

For clarity, and WLOG, we assume m is odd.

1) Deriving safe zones to the F_2 sketch function: To track the value of the AMS sketch, each node maintains its local sketch, which is an $m \times q$ sketch table. Then, the sum of the local sketches is the sketch table of the global stream. In order to make the global vector be the *point-wise mean* of the local vectors, and not their sum, each node multiplies its local vector by the number of nodes k.

We apply the Convex Bound method [5] to derive safe zones for F_{ams}. In order to bound the function from above, since the norm squared function is convex, only the median has to be treated. We bound the median by taking the maximum of the $\|r_i\|^2$ value of the $m' = \lfloor \frac{m}{2} + 1 \rfloor$ rows with the minimal $\|r_i\|^2$ value at sync time. Let I_U be the set of indices of the m' rows with smallest $\|r_i\|^2$ at sync time; the convex upper bound is thus:

$$\overrightarrow{c}(M) = \max_{i \in I_U} \|r_i\|^2 \ .$$

To derive a concave lower bound, we first bound each term $\|r_i\|^2$ by the tangent plane $\langle 2r_i(0), r - r_i(0) \rangle$ where $r_i(0)$ is the vector r_i of the public vector $v(0)$. Then, we bound the median function by taking the minimum of the tangent planes values of the m' rows with the maximal $\|r_i\|^2$ value at sync time. Let I_L be the set of indices of the m' rows with largest $\|r_i\|^2$ at sync time; the concave lower bound is thus:

$$\overrightarrow{c}(M) = \min_{i \in I_L} \langle 2r_i(0), r - r_i(0) \rangle \ .$$

2) L_2 Distance to the Convex Upper Bound: We now show how to derive the distance to the safe zone defined by the convex upper bound; deriving the lower concave bound is very similar and thus omitted. Since the monitoring is performed over vectors, the L_2 distance of the "flattened" sketch table over the vector space is actually a Frobenius norm-metric over matrices for the non-flattened sketch-table; thus, we'll refer to the Frobenius norm-metric as L_2 distance over the sketch-table space: $\|v_1 - v_2\| = \|M_1 - M_2\|_F$ where the vector v_i is the flattened version of M_i.

We claim that the distance vector to the boundary for a sketch table M which is *inside* the level set $\overrightarrow{c}$ – i.e. $\overrightarrow{c}(M) < T$ – 'elevates' exactly one row of M to a squared norm of T. This occurs since $\overrightarrow{c}$ is a *maximum* function over M's I_U rows, so the squared norm of just one row out of I_U has to be increased to "reach" the level-set.

The minimal L_2 change to increasing any row r_i of the sketch table to a squared norm T is the distance from the q-dimensional vector r_i to the q-dimensional origin-centered sphere with radius $\sqrt{T}$. This distance has a simple closed-form solution, $\sqrt{T} - \|r_i\|_2$. Hence, the inner-L_2 distance is the minimum out of the distances of the I_U rows of M to the q-dimensional sphere:

$$\min_{i \in I_U} \left(\sqrt{T} - \|r_i\|_2 \right) \ .$$

In contrast, computing the *outer*-L_2 distance to the level-set $\{M \mid \overrightarrow{c}(M) = T\}$ means we have to decrease the norm of *all* I_U's rows whose squared norm exceeds T. To decrease the row r_i to norm T, the L_2 distance behaves as a L_2 distance to a zero-centered q-dimensional sphere center with radius $\sqrt{T}$. So the L_2 distance from outside is the accumulation of those distances for the rows in I_U whose squared norm exceeds T:

$$\sqrt{\sum_{i \in I_U, \|r_i\|_2 > T} \left(\|r_i\|_2 - \sqrt{T} \right)^2} \ .$$

3) An Alternative Norm: Recall that the Distance Scheme can resolve its violation when the inner-distances are larger than the outer-distances. We claim that for the above L_2 distance, the outer-distance inherently tends to be "heavier" than the inner-distance, and thus isn't suitable for the Distance Scheme. Notice that both the inner and outer distances are computed using the distance to a q-dimensional sphere; though, the inner distance to the safe-zone is the distance of just one row to the sphere, while the outer-distance is an *accumulation* of distances to the sphere of *multiple* rows of the sketch-table. Hence, many non-violating vectors inside the safe zone may be required to balance the other violating vectors.

Since the Distance Lemma can be used with any norm, we introduce a "hybrid" norm $L_\sim$ for the matrix M, defined as:

$$\|M\|_\sim = \max_i \left(\|r_i\|_2 \right) \ . \tag{5}$$

The $L_\sim$ norm exploits the structural properties of the safe zone and the AMS F_2 function: $\overrightarrow{c}$ is a maximum function over the rows of the sketch table. Calculating the $L_\sim$ distance to the surface of the convex bound requires a change solely to *one* row r_i out of I_U (specifically, the row with the maximal L_2 distance), *both* from inside and from outside of the safe zone. It thus overcomes the problem with the L_2 norm described before. The $L_\sim$ distance to the surface from inside is the same as L_2: $\min_{i \in I_U} \left(\sqrt{T} - \|r_i\|_2 \right)$, and from outside, the

$L_\sim$ distance is the maximum of the L_2 distances of I_U's rows to the q-dimensional sphere:

$$\max_{i \in I_U, \|r_i\|_2 > \sqrt{T}} \left(\|r_i\|_2 - \sqrt{T} \right) .$$

Note that $L_\sim$ is *at least* as good as L_2, since its outer-distance is lower or equal to L_2's, while their inner-distances are equal.

C. Entropy Sketch

Entropy is widely used in various applications [7], for example identifying distributed denial of service attack and network anomalies [20]. The Shannon entropy of the probability vector $(p_1, ..., p_N)$ is defined as: $H(p_1, ..., p_N) = \sum_{i=1}^{N} -p_i \ln(p_i)$.

Reducing the bandwidth for monitoring the entropy can be made more efficient by using appropriate sketches. A near space-complexity optimal entropy-sketch was proposed in [19], where the sketch is a linear projection of the probability vector. The linear projection is performed by a multiplication matrix with i.i.d elements drawn from $F(x; 1, -1, \pi/2, 0)$ [19]. The sketch of the global probability vector is the mean of the sketches of the nodes' probability vectors, since the operation of multiplying the probability vector by the projection matrix is linear. The entropy approximation of the d-dimensional linear projected vector $(y_1, ..., y_d)$ is:

$$\tilde{H}(y_1, ..., y_d) = \ln(d) - \ln \left(\sum_{i=1}^{d} e^{y_i} \right) . \tag{6}$$

Safe zones for this entropy sketch are described in [7]: since the entropy sketch function $\tilde{H}$ is already concave, it can be used directly for the lower bound. For the upper bound, a tangent plane to the function is used.

1) Distance to the Safe Zone: Standard methods such as Lagrange Multipliers do not yield a closed-form solution for the L_2 distance to the level set of $\tilde{H}$. Instead, we use the L_1 distance, since the distance lemma works for every norm metric (Section III-A). For the upper safe zone (i.e. the tangent plane), the L_1 distance is used as well; this distance has a closed form solution, as detailed in [21]. We'll now lay out the mathematical details of calculating the L_1 distance of a vector v to the level set of $\{w \mid \tilde{H}(w) = T\}$.

2) L_1 distance from inside the lower safe zone: We calculate the *inner* L_1 distance to the safe zone by utilizing [22, Corollary 2.11], which states that a point within a convex set, and its closest L_1 point on the set's boundary, differ in only one coordinate. So, since $\tilde{H}$ is composed of a *separable* sum of e^{v_j} functions, the non-zero argument for increasing $\tilde{H}$ with minimum L_1 is achieved where the e^{v_j} value is maximal. Let i be the index of the maximal argument in v. Then, the vector u with minimal L_1 norm s.t. $\tilde{H}(v + u) = T$ is all zeros, except at the component u_i, whose value is calculated by the equation $\tilde{H}(v + u) = T$:

$$\ln(d) - \ln \left(\sum_{j \neq i} e^{v_j} + e^{v_i + u_i} \right) = T$$

Algorithm 2 Find q which minimizes $\tilde{H}(q)$ on the surface of the L_1-sphere centered at v with radius δ.

Input: vector $v \in \mathbb{R}^d$, L_1 sphere radius δ
Output: $q \in \mathbb{R}^d$, $\|q - v\|_1 = \delta$, which minimizes $\tilde{H}(q)$

```
q ← v
j = argmax_i q_i
S ← {j}
while ||q − v||_1 < δ and |S| < d do
    ℓ = argmax_{i∉S} v_i
    D = min{δ − ||q − v||_1, |S| · (q_j − v_ℓ)}
    for all i ∈ S do
        q_i ← q_i − D/|S|
    S ← S ∪ {ℓ}
if ||q − v||_1 < δ then
    for all i ∈ 1 ... d do
        q_i ← q_i − (δ − ||q − v||_1)/d
```

which yields:

$$u_i = \ln \left(e^{\ln(d) - T} - \sum_{j \neq i} e^{v_j} \right) - v_i .$$

3) L_1 distance from outside the upper safe zone: Conceptually, the *outer* L_1 distance to the convex safe zone $\{w \mid \tilde{H}(w) \geq T\}$ can be computed by "inflating" an L_1 sphere around v, until it intersects the surface of the safe zone. To do so efficiently, we propose an algorithm to compute the minimal $\tilde{H}$ value on a L_1 sphere with radius δ; then, we perform binary search on the radius δ of the L_1 sphere for which this minimal $\tilde{H}$ value is T.

Let v be the center of an L_1-sphere with radius δ and let q be the vector on the surface of the sphere for which $\tilde{H}$ is minimized. We observe that v and q differ by the *most dominant* (i.e., greatest) components of v, because $\tilde{H}$ is composed of a *separable* sum of e^{v_i}. We propose the following steps to evaluate q:

1) Initialize q to v.
2) Find j, the index of the maximum component of v.
3) Decrease q_j to the value of v's second largest element v_m, while preserving $\|q - v\|_1 \leq \delta$.
4) Jointly decrease q_j and q_m to the value of v's third largest component v_p, while preserving $\|q - v\|_1 \leq \delta$.
5) Continue as above; stop when $\|q - v\|_1 = \delta$.

Algorithm 2 describes our $\tilde{H}$ minimization process in detail. It is similar to the *least angle regression* (LARS) algorithm [23] for L_1-regularized regression, which operates on the components that are *most correlated* with the response. Due to lack of space, we omit the proof of correctness; instead, we refer the reader to [23].

V. Experimental Results

We empirically evaluate the communication cost of the Distance and Value schemes using several real-world datasets, and compare the results to other Geometric Monitoring (GM) schemes.

Our main evaluation metric is *bandwidth ratio*: the bandwidth required for distributed monitoring divided by the bandwidth required by centralization – sending local vectors to the coordinator and computing the function there. The lower the ratio is, the better. Our aim is to evaluate the proposed schemes, independent of any particular choice of network and communication protocol. We therefore follow [13] by defining message bandwidth as the number of floating point values in a message, with small control messages such as coordinator polling assigned a bandwidth cost of 1. We stress that centralization is not necessarily a naive or weak baseline. For example, for sketching approaches such as the AMS sketch or the entropy sketch, the size of local vectors d is already orders of magnitude smaller than the original data size; centralization can thus be state-of-the-art for pure sketching.

A. Experimental Setup

We focus on comparing the proposed bandwidth reduction schemes to standard GM, which is our main contribution, rather than evaluating how well GM is for any particular function. Thus, we have heavily relied on established work on using GM for distributed approximation; we refer the reader to that work for a thorough evaluation of GM for approximating inner product, AMS sketches, and entropy (Section IV). We evaluate the following schemes:

- **Vector Scheme:** standard GM protocol (Section II, Alg. 1) using state-of-the-art safe zones described in Section IV.
- **FGM:** Functional Geometric Monitoring [9], which replaces the GM protocol with a distributed counting protocol to reduce bandwidth. The convex bounds $\overrightarrow{c}()$, $\overleftarrow{c}()$ described in Section IV are used to derive FGM's *safe functions* [9]. To make sure the approximation is equivalent to the other approaches, we set the quantization parameter ϵ_ψ to 0.
- **Distance Scheme:** the Distance Scheme as described in Section III using the safe zones and distances derived in Section IV.
- **Value Scheme:** the Value Scheme as described in Section III.
- **RLV:** Real Local Violations [5], where nodes use the admissible region A (the global constraint) for monitoring instead of the safe zone Z, and run lazy sync when $p_i \notin A$. RLV is a hard benchmark to beat. Unlike the other schemes, it is allowed to break the approximation bounds: since A is not convex, it is possible that $f(v) > T$ but locally all $f(v_i) \leq T$. Moreover, by definition any local violation in RLV ($p_i \notin A$) implies local violation in other schemes ($p_i \notin Z$) but the reverse is not true.
- **Oracle:** nodes have global knowledge and run eager sync if and only if a true violation occurs: $f(v) > T$. While unrealistic, it provides a lower bound on bandwidth [8].

B. Functions and Datasets

We evaluate the benefit of the Distance and Value schemes on the three functions described in Section IV: inner product, AMS sketch for F_2, and an entropy sketch. To make our evaluation more realistic, for each of the approximated functions we use a real-world dataset.

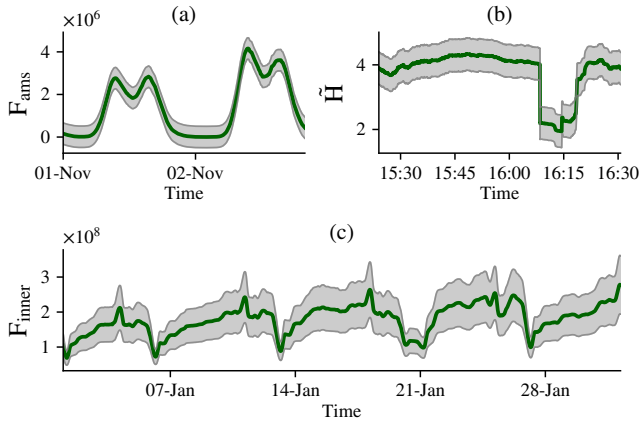


Fig. 3. The value of the monitored function over time for the different datasets, with approximation bounds shown as gray band. (a) F_2 AMS approximation for cellular phone activity in Milano, with additive approximation of $\pm 5 \cdot 10^5$. (b) Sketched entropy of source IP address in CTU dataset 3, with additive approximation of ± 0.5. (c) Inner product between between CMT and VTS taxi trips, with multiplicative approximation of $\times(1 \pm 0.3)$.

For inner product, we use the New York City Taxi Trip Dataset [24]. It documents taxi trips in New York City between 2010 and 2013, totalling 690M trips. Each entry records the date and time in which the trip took place, the pickup and drop-off coordinates, the vendor, the number of passengers, the fare, and so on. We use the inner product function to track the correlation of the geographical composition of trips between two taxi vendors, VTS and CMT, over the last 24 hours. Specifically, we divided the city into 101×101 geographical zones, and defined local vectors of size $d = 2 \times 101^2 = 20402$, where the first 101^2 elements are the number of drop-offs in each zone for the vendor VTS in a sliding window of 24 hours, and the other 101^2 elements are the number of trips for the vendor CMT for the same window. We monitor the inner product of the two sub-vectors, which reflects the correlation between typical routes of the two vendors in January 2013 using 24 hour sliding window with a step size of 1 hour, with a multiplicative approximation bounds $1 \pm \epsilon$. To determine which node receives a record, we divide the city to equal-sized areas and use the pickup location of the trip to determine the node. Unless otherwise noted, we use $k = 64$ nodes, and vector dimension of 2×101^2. Figure 3(c) shows the value of the approximated function over time.

For AMS F_2, we tracked the F_2-moment of cellular phone activity in Milano, Italy. The Mobile Phone Activity dataset [25] contains records of measurements of phone activity inside the city of Milano during November and Decemeber 2013 in 10 minutes intervals (total of 8784 activity records of size $10^4 \times 10^4$ each). The datatset divides Milano into 10^4 regions and contains inter-region activity, resulting in a phone activity vector of size $10^4 \times 10^4$ every 10 minutes. We use the AMS sketch to track the F_2 value of this high-dimensional

vector using an additive approximation $\pm\epsilon$ computed over a sliding window of 4 hours, with a step of 10 minutes. Unless otherwise noted, we use a sketch of size $d = 14 \times 11$ and $\epsilon = 5 \cdot 10^5$. We distributed phone activity records into nodes according to the grid-based location of where the phone call originated from: we divided the 10^4 parts of the city into 4×4 areas which results in 16 nodes, unless specified otherwise. Also, unless mentioned otherwise, we use the norm $L_\sim$ (sec. IV-B3 eq. 5) when monitoring with the Distance Scheme. Figure 3(a) shows the value of the AMS F_2 moment in the dataset over time.

For entropy, we monitor network traffic to detect attacks. The CTU-13 dataset [26] consists of 13 datasets of records of network traffic with different attacks. If the entropy of source IP addresses suddenly drops, it may indicate a possible attack since a small range of IP addresses dominate network traffic [27], [28]. We therefore aim to monitor the entropy of the frequency of source IP addresses in the third dataset in CTU-13 with 4.7M network flows, defined over a sliding window of 6 minutes and a step of one second. Since monitoring 2^{32} IPv4 addresses is infeasible, we use an entropy sketch to reduces the cardinality of the vectors to a sketch of size d, and monitor that instead. The network communication was distributed to k nodes according to the third byte of the source IP address. Unless specified otherwise, we monitor an additive approximation of the sketch with $\epsilon = 0.5$ with a sketch size of $d = 200$, and with $k = 10$ nodes. Figure 3(b) shows the value of sketched entropy in the dataset over time.

C. Number of Nodes k

We first consider the effect of the number of nodes k on communication cost. Previous work has shown that GM-based methods can incur higher communication cost as the number of nodes k increases, as resolving violations involves more and more nodes [4], [7], [15].

Figure 4 shows the resulting bandwidth ratio as we vary the number of nodes in each run. We make several observations.

First, the Distance and Value schemes are effective in reducing bandwidth costs. The Distance Scheme has substantially lower bandwidth cost compared to both the Value Scheme and the state-of-the-art (FGM and Vector Scheme). The bandwidth cost for the Value Scheme is as low or slightly lower than that of the counting-based FGM. We discuss this equivalence in more detail in Section VI. The Vector Scheme, on the other hand, incurs higher bandwidth costs since any violation results in transmission of large vectors. For AMS F_2 (Fig. 4(b)) with $k \geq 256$ it incurs as much bandwidth as centralization.

Second, we observe large differences in the scalability of the different approaches. While the Vector Scheme has high bandwidth ratio, its cost does not change much as we increase k. For inner product (Fig. 4(a)), its cost does not increase with more nodes (consistent with previous work [5]). Conversely, the Value Scheme and FGM scale poorly as we increase the number of nodes; in particular, for entropy (Fig. 4(c)) both schemes perform worse than the Vector Scheme when $k > 20$.

Notably, the Distance Scheme scales much better than either the FGM and the Value Scheme as we increase the number of nodes k, while simultaneously using less (sometimes much less) bandwidth than other approaches. Indeed, for inner product and AMS, the bandwidth ratio of the Distance Scheme is close to or better than that of both the challenging RLV scheme (which is allowed to break approximation bound) and the unrealistic Oracle.

We conclude that the Distance Scheme combines the scalability of the Value Scheme with superior bandwidth reduction.

D. Dimension d

To evaluate how the dimension of the local and global vectors affects bandwidth ratio, we repeat each experiment with a different number of dimensions d. For the entropy sketch we directly set the size of the sketch in order to control d. Similarly, for the AMS sketch we vary the width and the height of the sketch table: we set the width to $\lceil\sqrt{d}+1\rceil$ and the height to $\lfloor\sqrt{d}-1\rfloor$. For inner product, we divide the city to smaller zones to increase the number of entries in the vectors. Figure 5 shows the resulting bandwidth ratio when approximating inner product, F_2 moment, and entropy.

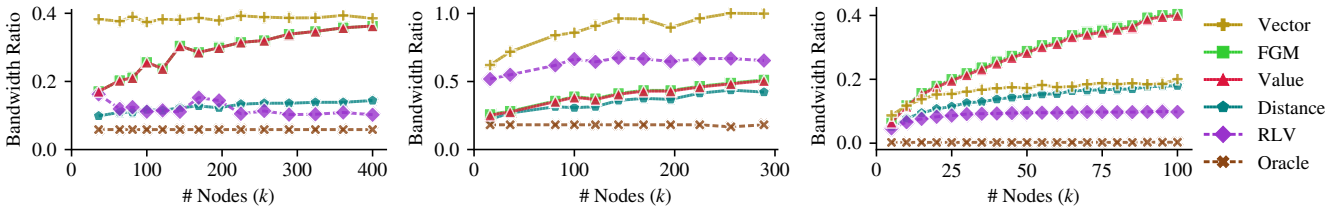
The effect of local vector size d on bandwidth ratio is similar for the different schemes, and depends more on the monitored function. For both inner product and the entropy sketch, increasing d results in increasing bandwidth consumption for the Distance scheme, Value Scheme, FGM and Vector Scheme. In contrast, for AMS F_2, changing d does not affect the bandwidth ratio in any of the the monitoring scheme. Interestingly, for the entropy sketch, the Value Scheme and FGM are more sensitive to larger vectors: increasing d causes a steeper increase of the bandwidth consumption ratio than in the Vector Scheme and Distance Scheme. This results in the FGM and the Value Scheme performing worse for the entropy sketch when $d \geq 300$, as it did for $k \geq 20$ in Fig. 4(c).

As before, we observe that the Distance Scheme consistently outperforms the other schemes, especially in higher dimensions. Its bandwidth ratio is similar or superior to RLV across all tested functions.

Finally, the Value Scheme performs similarly to FGM, with a slight advantage for low d, shown in Fig. 5(b) and Fig. 5(c). This is due to FGM sending more messages than the Value Scheme. While both the Value Scheme and FGM both have the same number of full (eager) synchronizations, FGM sends more messages than the Value Scheme, as we later show in Sec. V-H. In higher dimensions the cost of these extra messages is dominated by the cost of eager syncs, which send the full vector of size d. The relation between the Value Scheme and FGM is further discussed in Sec. VI.

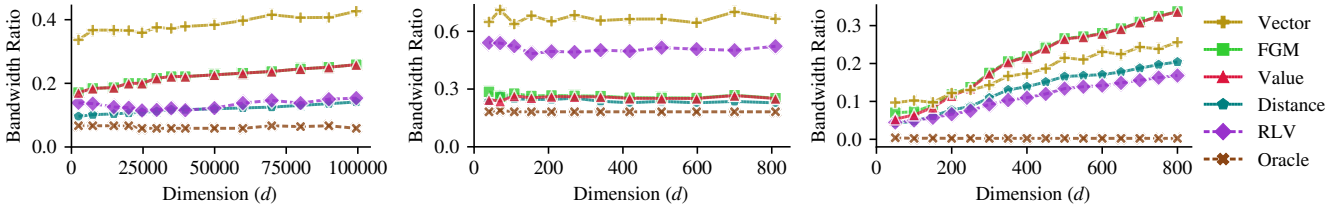
E. Bandwidth and Latency Simulations

Our previous simulation uses an idealized network. Here we simulate transfer volume and latency when using standard network protocols. Since performance depends on the network stack, we include results for two stacks. For the high-end stack (WiFi) we assume UDP/IPv4 over a reliable[4] IEEE



(a) Inner product (b) AMS F_2 (c) Entropy sketch

Fig. 4. Communication cost (bandwidth ratio) when approximating the inner product, F_2 moment, and entropy with different number of nodes k.



(a) Inner product (b) AMS F_2 (c) Entropy sketch

Fig. 5. Communication cost (bandwidth ratio) as a function of the dimension of local vectors d.

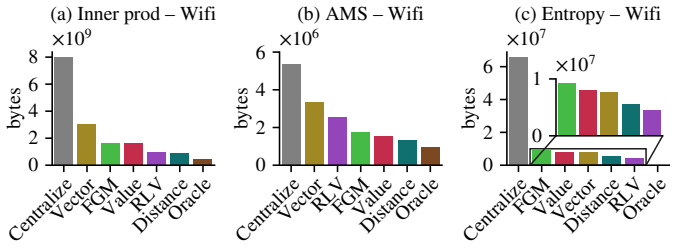


Fig. 6. Total transfer volume over WiFi for (a) inner product, (b) AMS F_2, and (c) entropy sketch, using default parameters (Sec. V-B). The gray bar "Centralize" shows the bandwidth incurred by centralizing data updates.

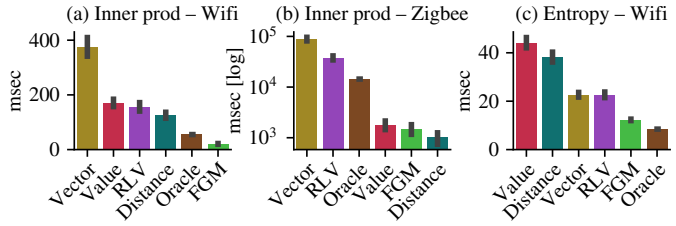


Fig. 7. Average time to resolve violations for inner product over WiFi (a) and Zigbee (b), and entropy sketch over WiFi (c). Error bars show std. deviation.

802.11g WiFi link at 54Mbit/sec with an additional 12 bytes per payload for sequencing and control, resulting in messages of size up to 2304 bytes, headers of size 40, and payload of up to 283 double precision scalars per message. For the low-end stack (Zigbee) we assume an IEEE 802.15.4 with 250Kbps bandwidth in a star topology, yielding headers of size 36, up to 128 bytes per message, and a payload of up to 11 scalars per message. For both settings we assume 4ms latency.

Figure 6 shows total WiFi transfer volume (bandwidth used) for all schemes and functions in default configuration (see Sec. V-B). For inner product and AMS F_2, our previous observations generally hold: the Distance and Value schemes reduces transfer volume substantially over the original Vector Scheme, with the Distance Scheme as good or better than the unrealistic RLV. For WiFi the default entropy sketch vectors easily fit within a message, meaning that the Distance Scheme is only a little better than the Vector Scheme, while the Value Scheme and FGM are slightly worse. This is not unexpected, since the Distance and Value schemes were designed to address bandwidth in high-dimensional data. The picture changes for the Zigbee protocol: its smaller payload means all schemes generally incur additional bandwidth due to the need to split messages (figure omitted for lack of space). The Vector Scheme in particular suffers from increased overhead, requiring 10.7MB for entropy sketch, while the Distance Scheme requires only 7.4MB; both however are much better than centralization (91MB). Interestingly, for entropy sketch FGM uses more bandwidth than even the Vector Scheme for both WiFi and Zigbee (12MB). As we discuss in Sec. V-H, FGM sends many more messages than the other schemes due to the counting protocol, which results in extra bandwidth.

Figure 7 shows the average time needed to resolve a violation. Lazy sync can increase this time since the coordinator contacts nodes one by one, adding round-trips. Conversely, FGM does not suffer from higher latency since some violations are resolved via a quantum message that requires no response, while full sync messages are equivalent to eager sync and done for all nodes in parallel. Oracle time is constant in our simulation since it only exchanges fixed-size vectors.

Surprisingly, for inner product on WiFi the Vector Scheme has the highest violation resolution latency (376ms) due to the

[4]Message reliability is a complex subject outside the scope of this work; we assume it is handled by the lower level transport (with potential increase in bandwidth for all schemes).

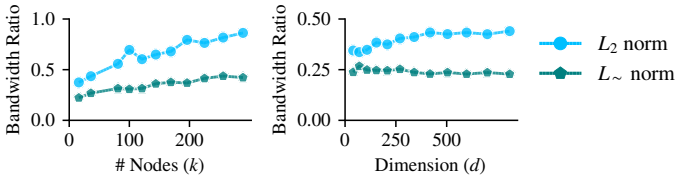


Fig. 8. Bandwidth consumption of the AMS F_2 function relative to the choice of norm, when changing the number of nodes (left) and the dimension (right).

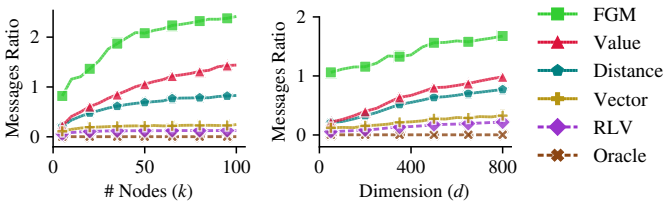


Fig. 9. Entropy sketch: ratio between the number of messages sent by different schemes to the number of messages sent by the centralized approach

large size of the vector (∼160KB), while the Distance Scheme only requires around 125ms. For Zigbee, the difference is even more dramatic: network bandwidth and message payload are so small that the benefits of the Distance Scheme are magnified. This is another benefit of the Distance and Value schemes: for low-bandwidth or high-dimensional settings, the time it takes to transmit a full vector ($\frac{160\text{KB}\times 8}{54\text{Mbps}} = 23$ms on WiFi) is large enough that the benefit of bandwidth reduction can outweigh the extra time needed for lazy sync. Even if we used eager sync for the Vector Scheme, the time it would take to resolve a violation would still be around 100ms. For the entropy sketch, d is lower, meaning that the Vector Scheme takes on average 22ms to resolve violations, while the Distance and Value schemes take 38ms and 44ms respectively. AMS F_2 results are similar to entropy, but the Distance and Value schemes slightly faster (32ms and 35ms, respectively).

We conclude that for high dimensional data or low-bandwidth networks, the Distance and Value schemes dramatically reduce the time it takes to resolve violations over the Vector Scheme. For low-dimensional data, they introduce a mild overhead but are still are fast enough to monitor data that changes multiple times every second.

F. Approximation Tightness ϵ

As with any distributed approximation algorithm, the communication cost of GM methods varies with the tightness of the approximation. We explored the effect of varying ϵ on the bandwidth ratio (figures ommitted for lack of space). As expected, tighter approximations result in more communication across all schemes. The approximation factor affects all schemes similarly: the relative order and advantage of the different schemes remains roughly the same across a wide range of ϵ values across all three functions (with the possible exception of very tight approximations where centralization is preferred anyway since the bandwidth ratio is close to or above 1.0). This suggests that our prior conclusions hold across a range of ϵ values.

G. Choice of Norm

As discussed in Sec. IV-B3, some functions, such as AMS F_2, have a structure that we can exploit to provide a norm tailored to that function. We now demonstrate that such a tailored norm can have a substantial effect on the bandwidth ratio. Fig. 8 shows that the bandwidth ratio of monitoring AMS F_2 using the Distance Scheme with the tailored $L_\sim$ norm (5) is $\times$1.4–2.4 times lower than with the L_2 norm. This confirms our conjecture in Sec. IV-B3 that for the convex safe zone of the AMS F_2, the L_2 norm tends to be "heavier" from outside of the safe zone, and thus have more false alarms. Since the Distance Scheme can accommodate any norm, engineering such well-suited norms presents an opportunity for further optimization. We leave such discussion for future work.

H. Number of Messages

As we discuss in Sec. III, most prior GM research focused on the number of messages sent by the schemes, rather than bandwidth. The number of messages can be an important metric in some settings, for example when simply turning on the wireless device incurs significant power consumption compared to sending one bit [10], [11]. We follow previous work and compare the different schemes based on their *messages ratio*: the number of messages transmitted by a scheme divided by to the number of messages needed for centralization. Fig. 9 shows the messages ratio for entropy sketch as we increase the number of nodes k and dimension d.

The Vector Scheme sends much fewer messages than the other schemes even as we increase the number of nodes and dimensions. The next best scheme is the Distance Scheme, followed by the Value Scheme. Notably, FGM sends substantially more messages than all other schemes. Even though FGM and the Value Scheme achieve the same bandwidth ratio, FGM sends about twice as many messages as the Value Scheme. We attribute this to the counting protocol at the heart of FGM, which is designed to reduce bandwidth but not necessarily the number of messages: nodes must report every local *quantum* crossing [9]. We observed similar behavior for inner product and AMS F_2 (figures omitted due to lack of space).

VI. Related Work

The most closely related work is functional geometric monitoring (FGM) [9], which follows the spirit of GM principles but replaces the GM monitoring protocol with a distributed monotonic counting protocol. FGM is similar to the Value Scheme, since both decide whether initiate a violation resolution phase based on the value of a convex bound for the monitored function. Therefore, it suffers from the inherent bias towards false alarms (Section III-D), and is generally outperformed by the proposed Distance Scheme.

FGM's use of counting protocol is a key difference between FGM and the Distance and Value schemes (and existing

GM approaches). Using a counting protocol greatly reduces monitoring bandwidth, and allows reasoning about and even bounding the communication cost of FGM, which is difficult to do for GM.

However, unlike the Distance and Value schemes, which are designed to work with any existing GM technique, replacing the standard GM monitoring protocol means that FGM is no longer compatible with a rich set of GM optimization techniques that rely on this shared geometric protocol. For example, dynamically predicting changes in v and the safe zone Z over time can substantially reduce the amount of local violations and hence communication [12], [13]. While replacing the local constraint $p_i \in Z$ with the predicted version of Z is straightforward in GM protocol, the equivalent is difficult to achieve in FGM since that would require updating the state of the counter and the *quantum* θ. Similarly, the use of safe zones in the GM protocol allows simultaneous monitoring multiple functions without incurring the communication costs of monitoring each individual function [3], as well as combining safe zones to monitoring complex functions built from simpler ones [14]. Examples of other such techniques include probabilistic geometric monitoring [29] and applications of GM, such as [16].

Moreover, if monitoring the same function using the same convex bound $c()$ (Value Scheme) and its equivalent safe function ϕ (FGM), the Value Scheme will require an eager sync if and only if FGM would have an equivalent full synchronization. This is because the condition for performing an eager (or full) synchronization is the same in both methods: $\frac{1}{k}\sum c(p_i) > T$ for the Value Scheme if and only if $\frac{1}{k}\sum \phi(p_i) > 0$ for FGM (note the threshold T is already incorporated in ϕ). Therefore, when monitoring functions with high cardinality, eager syncs dominate the bandwidth consumption, so both methods consume similar bandwidth.

VII. Conclusions

The Distance Scheme is a novel bandwidth-efficient adaptation for the Geometric Monitoring (GM) family of distributed monitoring techniques. By communicating scalars to resolve local violations, it substantially reduces bandwidth while remaining compatible with and benefiting from previous work on GM. We evaluate its performance on three non-trivial functions using real-world datasets, and show that the Distance Scheme substantially reduces both bandwidth and number of messages compared to the current SotA. For cases in which the Distance Scheme is difficult to apply, we describe the simpler Value Scheme, which matches SotA bandwidth performance and reduces the number of messages, while retaining compatibility with prior work.

References

[1] G. Cormode, "The continuous distributed monitoring model," *SIGMOD Record*, vol. 42, no. 1, pp. 5–14, 2013.

[2] I. Sharfman, A. Schuster, and D. Keren, "A geometric approach to monitoring threshold functions over distributed data streams," in *SIGMOD*, 2006.

[3] A. Lazerson, M. Gabel, D. Keren, and A. Schuster, "One for all and all for one: Simultaneous approximation of multiple functions over distributed streams," in *DEBS*, 2017.

[4] M. Gabel, D. Keren, and A. Schuster, "Monitoring least squares models of distributed streams," in *KDD*, 2015.

[5] A. Lazerson, D. Keren, and A. Schuster, "Lightweight monitoring of distributed streams," *TODS*, vol. 43, no. 2, p. 9, 2018.

[6] G. Yehuda, D. Keren, and I. Akaria, "Monitoring properties of large, distributed, dynamic graphs," in *IPDPS*, 2017.

[7] M. Gabel, D. Keren, and A. Schuster, "Anarchists, unite: Practical entropy approximation for distributed streams," in *KDD*, 2017.

[8] M. Garofalakis, D. Keren, and V. Samoladas, "Sketch-based geometric monitoring of distributed stream queries," *PVLDB*, vol. 6, no. 10, pp. 937–948, 2013.

[9] V. Samoladas and M. N. Garofalakis, "Functional geometric monitoring for distributed streams." in *EDBT*, 2019.

[10] G. Anastasi, M. Conti, M. Di Francesco, and A. Passarella, "Energy conservation in wireless sensor networks: A survey," *Ad Hoc Networks*, vol. 7, no. 3, pp. 537 – 568, 2009.

[11] C. Stylianopoulos, M. Almgren, O. Landsiedel, and M. Papatriantafilou, "Geometric monitoring in action: a systems perspective for the internet of things," in *LCN*, 2018.

[12] N. Giatrakos, A. Deligiannakis, M. Garofalakis, I. Sharfman, and A. Schuster, "Prediction-based geometric monitoring over distributed data streams," in *SIGMOD*, 2012.

[13] M. Gabel, A. Schuster, and D. Keren, "Communication-efficient distributed variance monitoring and outlier detection for multivariate time series," in *IPDPS*, 2014.

[14] M. Garofalakis and V. Samoladas, "Distributed query monitoring through convex analysis: Towards composable safe zones," in *ICDT*, 2017.

[15] A. Lazerson, I. Sharfman, D. Keren, A. Schuster, M. Garofalakis, and V. Samoladas, "Monitoring distributed streams using convex decompositions," *PVLDB*, vol. 8, no. 5, pp. 545–556, 2015.

[16] O. Papapetrou and M. Garofalakis, "Continuous fragmented skylines over distributed streams," in *ICDE*, 2014.

[17] R. Bernstein, M. Osadchy, D. Keren, and A. Schuster, "Lda classifier monitoring in distributed streaming systems," *Journal of Parallel and Distributed Computing*, vol. 123, pp. 156–167, 2019.

[18] N. Alon, Y. Matias, and M. Szegedy, "The space complexity of approximating the frequency moments," *J. Comput. Syst. Sci.*, vol. 58, no. 1, pp. 137–147, 1999.

[19] P. Clifford and I. Cosma, "A simple sketching algorithm for entropy estimation over streaming data," in *Artificial Intelligence and Statistics*, 2013, pp. 196–206.

[20] C. Arackaparambil, S. Bratus, J. Brody, and A. Shubina, "Distributed monitoring of conditional entropy for anomaly detection in streams," in *IPDPSW*, 2010.

[21] E. Melachrinoudis, "An analytical solution to the minimum L_p-norm of a hyperplane," *J. Math. Anal. Appl*, vol. 211, no. 1, pp. 172–189, 1997.

[22] H. J. Tuenter, "Minimum L_1-distance projection onto the boundary of a convex set: Simple characterization," *J. Optim. Theory Appl.*, vol. 112, no. 2, pp. 441–445, 2002.

[23] B. Efron, T. Hastie, I. Johnstone, R. Tibshirani *et al.*, "Least angle regression," *The Annals of statistics*, vol. 32, no. 2, pp. 407–499, 2004.

[24] B. Donovan and D. Work, "New York city taxi trip data (2010-2013)," 2016. [Online]. Available: https://doi.org/10.13012/J8PN93H8

[25] G. Barlacchi, M. De Nadai, R. Larcher, A. Casella, C. Chitic, G. Torrisi, F. Antonelli, A. Vespignani, A. Pentland, and B. Lepri, "A multi-source dataset of urban life in the city of Milan and the province of Trentino," *Scientific data*, vol. 2, p. 150055, 2015.

[26] S. Garcia, M. Grill, J. Stiborek, and A. Zunino, "An empirical comparison of botnet detection methods," *computers & security*, vol. 45, pp. 100–123, 2014.

[27] A. Lakhina, M. Crovella, and C. Diot, "Mining anomalies using traffic feature distributions," *ACM SIGCOMM computer communication review*, vol. 35, no. 4, pp. 217–228, 2005.

[28] K. Xu, Z.-L. Zhang, and S. Bhattacharyya, "Profiling internet backbone traffic: behavior models and applications," *ACM SIGCOMM Computer Communication Review*, vol. 35, no. 4, pp. 169–180, 2005.

[29] N. Giatrakos, A. Deligiannakis, M. N. Garofalakis, D. Keren, and V. Samoladas, "Scalable approximate query tracking over highly distributed data streams with tunable accuracy guarantees," *Inf. Syst.*, vol. 76, pp. 59–87, 2018.