

# Scaling Geometric Monitoring Over Distributed Streams

Alexandros D. Keros

June 23, 2016

Supervised by: Prof. V.Samoladas

# Table of contents

## Introduction

## Theoretical Background

The Geometric Monitoring Method

Related Work

Theoretical Tools

## Problem Statement & Implementation

Problem Statement

Implementation

## Experimental Results

Data & Setup

Experiments

## Conclusions & Future Work

Conclusion

Future Work

## Introduction

## Theoretical Background

- The Geometric Monitoring Method
- Related Work
- Theoretical Tools

## Problem Statement & Implementation

- Problem Statement
- Implementation

## Experimental Results

- Data & Setup
- Experiments

## Conclusions & Future Work

- Conclusion
- Future Work

# Data Stream Systems

(Brian Babcock et al. "Models and Issues in Data Stream Systems". In: *21st ACM SIGMOD-SIGACT-SIGART. PODS '02*. 2002)

- ▶ **Data streams:** Continuous, high volume, size unbound, violative, probably distributed
- ▶ *Pull paradigm*
- ▶ Centralizing and/or polling → prohibitive in terms of communication overhead
- ▶ Examples: telecommunication, sensor networks

# The Geometric Monitoring Method

(Izchak Sharfman, Assaf Schuster, and Daniel Keren. "A Geometric Approach to Monitoring Threshold Functions over Distributed Data Streams". In: *2006 ACM SIGMOD ICMD*. SIGMOD '06. 2006)

- ▶ Threshold monitoring
- ▶ Nodes communicate when needed
  - ▶ Local constraints
  - ▶ Violation resolution (*false alarms*)
- ▶ Arbitrary function monitoring
- ▶ Tight accuracy bounds
- ▶ A promising framework for *distributed data stream monitoring*

# Motivation

## Problems:

- ▶ increasing node population
- ▶ data volume
- ▶ data dimensionality
- ▶ arbitrary functions
- ▶ **communication - accuracy tradeoff**

## Need for:

- ▶ scalability warranties
- ▶ tight accuracy bounds
- ▶ incremental/real-time operation
- ▶ **Minimize communication while retaining accuracy bounds**

# Motivation

## Problems:

- ▶ increasing node population
- ▶ data volume
- ▶ data dimensionality
- ▶ arbitrary functions
- ▶ **communication - accuracy tradeoff**

## Need for:

- ▶ scalability warranties
- ▶ tight accuracy bounds
- ▶ incremental/real-time operation
- ▶ **Minimize communication while retaining accuracy bounds**

# Motivation

## Problems:

- ▶ increasing node population
- ▶ data volume
- ▶ data dimensionality
- ▶ arbitrary functions
- ▶ **communication - accuracy tradeoff**

## Need for:

- ▶ scalability warranties
- ▶ tight accuracy bounds
- ▶ incremental/real-time operation
- ▶ **Minimize communication while retaining accuracy bounds**



# Motivation

## Problems:

- ▶ increasing node population
- ▶ data volume
- ▶ data dimensionality
- ▶ arbitrary functions
- ▶ **communication - accuracy tradeoff**

## Need for:

- ▶ scalability warranties
- ▶ tight accuracy bounds
- ▶ incremental/real-time operation
- ▶ **Minimize communication while retaining accuracy bounds**

# Contributions

Expand the *geometric monitoring method*:

- ▶ heuristic method for violation resolution
- ▶ distance-based hierarchical node clustering (Daniel Keren, Guy Sagy, Amir Abboud, David Ben-David, et al. "Geometric Monitoring of Heterogeneous Streams." In: *IEEE TKDE* (2014))
- ▶ throughout method evaluation on synthetic and real-world datasets

# Contributions

Expand the *geometric monitoring method*:

- ▶ heuristic method for violation resolution
- ▶ distance-based hierarchical node clustering (Daniel Keren, Guy Sagy, Amir Abboud, David Ben-David, et al. "Geometric Monitoring of Heterogeneous Streams." In: *IEEE TKDE* (2014))
- ▶ throughout method evaluation on synthetic and real-world datasets

# Contributions

Expand the *geometric monitoring method*:

- ▶ heuristic method for violation resolution
- ▶ distance-based hierarchical node clustering (Daniel Keren, Guy Sagy, Amir Abboud, David Ben-David, et al. "Geometric Monitoring of Heterogeneous Streams." In: *IEEE TKDE* (2014))
- ▶ throughout method evaluation on synthetic and real-world datasets

## Introduction

## Theoretical Background

- The Geometric Monitoring Method
- Related Work
- Theoretical Tools

## Problem Statement & Implementation

- Problem Statement
- Implementation

## Experimental Results

- Data & Setup
- Experiments

## Conclusions & Future Work

- Conclusion
- Future Work

## Geometric Threshold Monitoring

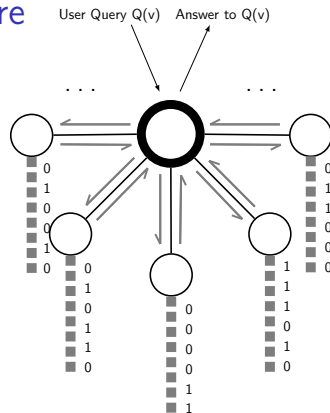
- ▶ Izchak Sharfman, Assaf Schuster, and Daniel Keren. “A Geometric Approach to Monitoring Threshold Functions over Distributed Data Streams”. In: *2006 ACM SIGMOD ICMD. SIGMOD '06*. 2006
- ▶ **Threshold monitoring:** arbitrary function  $f(\cdot)$ , threshold  $T$

$$f(\cdot) < T \text{ or } f(\cdot) > T$$

- ▶ **Idea:** decompose into local constraints at the nodes

# System Architecture

## Centralized Scenario



**Figure: Star-like network topology** example of the centralized scenario. The bold node represents the coordinator node. Dashed lines represent data streams and half arrows represent message exchanges.

# Computational Model

## Statistics vectors

- ▶ the *monitoring function*  $f : \mathbb{R}^d \rightarrow \mathbb{R}$
- ▶ the *threshold*  $T \in \mathbb{R}$
- ▶ the *monitoring node set* :  $P = \{p_1, \dots, p_n\}$   
with *weights*  $w_1, \dots, w_n$
- ▶ the *data streams* :  $S = \{s_1, \dots, s_n\}$
- ▶ the  $d$ -dimensional *local statistics vectors* :  $\vec{v}_1(t), \dots, \vec{v}_n(t)$   
represent each node's data stream at time  $t$

## Global statistics vector

$$\vec{v}(t) = \frac{\sum_{i=1}^n w_i \vec{v}_i(t)}{\sum_{i=1}^n w_i}$$



# Computational Model

## Estimate vector

Infrequent communication between nodes/nodes-coordinator:

### Estimate vector

$$\vec{e}(t) = \frac{\sum_{i=1}^n w_i \vec{v}_i'}{\sum_{i=1}^n w_i}$$

- ▶ the last communicated *local statistics vector* of node  $p_i$  :  $\vec{v}_i'$
- ▶ *Local statistics divergence*:  $\Delta \vec{v}_i(t) = \vec{v}_i(t) - \vec{v}_i', i = 1, \dots, n$

### Centralized drift vector

$$\vec{u}_i(t) = \vec{e}(t) + \Delta \vec{v}_i(t) + \frac{\vec{\delta}_i}{w_i}$$

# Computational Model

## Balancing Process

### Centralized scenario

**Purpose:** resolve possible false alarms

#### Balancing vector

$$\vec{b} = \frac{\sum_{p_i \in P'} w_i \vec{u}_i(t)}{\sum_{p_i \in P'} w_i}$$

- ▶ the *balancing set*  $P'$ : a subset of nodes
- ▶ the *slack vector* at the nodes  $\vec{\delta}_i = \vec{\delta}_i' + \Delta \vec{\delta}_i$ ,  $\sum_{p_i \in P'} \Delta \vec{\delta}_i = \vec{0}$ :

$$\Delta \vec{\delta}_i = w_i \vec{b} - w_i \vec{u}_i(t) \quad \forall p_i \in P'$$

, readjusts the *drift vectors*.

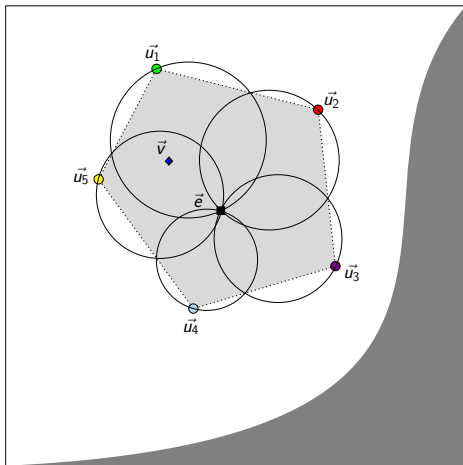
# Geometric Interpretation

## Convexity Property & Local Constraints

### Convexity Property

$$\vec{v}(t) = \frac{\sum_{i=1}^n w_i \vec{u}_i(t)}{\sum_{i=1}^n w_i}$$

**Figure:** Example of a convex hull (light gray) defined by the drift vectors  $\vec{u}_i, i = 1, 2, 3, 4, 5$  and bounded by spheres.



## Related Work

- ▶ *Safe Zones*: optimal local constraints fitted to nodes' data distributions (Daniel Keren, Guy Sagy, Amir Abboud, David Ben-David, et al. "Safe-Zones for Monitoring Distributed Streams". In: *First International Workshop on Big Dynamic Distributed Data, Riva del Garda, Italy, 2013*)
- ▶ Ellipsoidal bounding regions, decouplement of estimate vector from bounding ball construction (D. Keren et al. "Shape Sensitive Geometric Monitoring". In: *IEEE TKDE (2012)*)
- ▶ Simple shapes as local constraints, hierarchical clustering of nodes for participation to the balancing operation (Daniel Keren, Guy Sagy, Amir Abboud, David Ben-David, et al. "Geometric Monitoring of Heterogeneous Streams." In: *IEEE TKDE (2014)*)
- ▶ Prediction models based on velocity and acceleration (Nikos Giatrakos et al. "Prediction-based Geometric Monitoring over Distributed Data Streams". In: *2012 ACM SIGMOD ICMD. SIGMOD '12. 2012*)

## Multi-objective Optimization

- ▶ **Multiple**, possibly **conflicting** objectives to be *simultaneously* optimized
- ▶ *Pareto optimality*(*non-dominated solutions*): optimal solutions where none of the objective functions can be optimized without the simultaneous degradation of other objective functions' values.
- ▶ Let vector of  $m$  objectives  $F(x) = [F_1(x), F_2(x), \dots, F_m(x)]$ :

$$\min_{x \in \mathbb{R}^n} F(x)$$

$$\text{s.t. } l \leq x \leq u$$

$$G_i = 0, i = 1, \dots, k_e$$

$$G_j \leq 0, j = k_e + 1, \dots, k$$

- ▶ Finding *Pareto optimal* solutions is generally **NP-hard**.

# SQP

- ▶ The *Lagrangian function*:  $\mathcal{L}(x, \lambda) = F(x) + \sum_{i=1}^k \lambda_i G_i(x)$
- ▶ *Quadratic programming subproblems*:

$$\min_{d \in \mathbb{R}^n} \frac{1}{2} d^T H_t d + \nabla F(x_t)^T d$$

$$\nabla G_i(x_t)^T d + G_i(x_t) = 0, i = 1, \dots, k_e$$

$$\nabla G_i(x_t)^T d + G_i(x_t) \leq 0, i = k_e + 1, \dots, k$$

,where:

$H_t$ : Hessian of the Lagrangian function at iteration  $t$

$d$ : search direction

## The Savitzky-Golay Low-Pass Smoothing Filter

- Convolution based **smoothing**, **velocity** and **acceleration** estimation
- *Moving window averaging* paradigm:  $g_i = \sum_{n=-n_L}^{n_R} c_n f_{i+n}$
- *Least-squares fit* of polynomial:

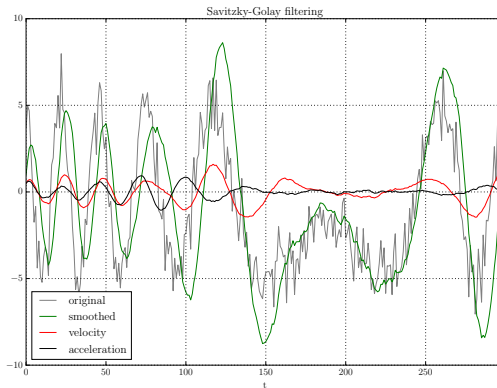
$$y_i(x) = a_0 + a_1 \frac{x - x_i}{\Delta x} + a_2 \left( \frac{x - x_i}{\Delta x} \right)^2 + \cdots + a_M \left( \frac{x - x_i}{\Delta x} \right)^M$$

,over window  $n_L + n_R + 1$ :

$$\sum_{j=i-n_L}^{i+n_R} (y_i(x_j) - f_j)^2 = \min$$

- Set  $g_i$  to the value of the fitted point  $x_i$ .

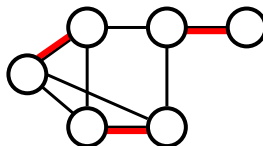
# Velocity and Acceleration Estimation via SG Filtering



**Figure:** Savitzky-Golay filtering of a signal with added Gaussian noise. The smoothing window is 50 points, centered at the far right.



# Maximum Weight Matching



Let  $G = (V, E)$  a graph:

- ▶ *maximum weight matching*  $M \subseteq E$  : a subset of edges where
  - ▶ no two edges share a common vertex
  - ▶ largest possible number of edges
  - ▶ maximizes the sum of weights

# Maximum Weight Matching

## The Primal-Dual Method

Constraints in Primal  $\iff$  Variables in Dual  
 Constraints in Dual  $\iff$  Variables in Primal

### The primal:

$$\begin{aligned}
 \max \quad & \sum_{(u,v) \in E} x_{u,v} w_{u,v} \\
 \text{s.t.} \quad & x_{u,v} \geq 0, (u,v) \in E \\
 & \sum_{u \in e: e \in E} x_e \leq 1, u \in V
 \end{aligned}$$

### The dual:

$$\begin{aligned}
 \min \quad & \sum_{u \in V} y_u \\
 \text{s.t.} \quad & y_u \geq 0, u \in V \\
 & y_u + y_v \geq w_{u,v}, (u,v) \in E
 \end{aligned}$$

## Introduction

## Theoretical Background

The Geometric Monitoring Method  
Related Work  
Theoretical Tools

## Problem Statement & Implementation

Problem Statement  
Implementation

## Experimental Results

Data & Setup  
Experiments

## Conclusions & Future Work

Conclusion  
Future Work

# Problem Formulation

**Reduce the communication burden** of the *Geometric Monitoring* method by:

- ▶ Optimally position *drift vectors* during the *balancing process*
- ▶ Appropriate node selection for inclusion in the *balancing set*

,in order to **increase scalability** in terms of:

- ▶ node population
- ▶ stream dimensionality.

# The Geometric Monitoring Framework

## Assumptions

- ▶ Coordinator-based scenario
- ▶ Instantaneous, loss-less, reliable communication
- ▶ Iterative operation based on time-steps
- ▶ System pause during violation resolution
- ▶ Coordinator node does not monitor a stream

# The Distance-based Hierarchical Clustering

## The Idea

Node selection for *violation resolution*:

- ▶ Elevate *randomness* of the initial *geometric monitoring* method (Izchak Sharfman, Assaf Schuster, and Daniel Keren. "A Geometric Approach to Monitoring Threshold Functions over Distributed Data Streams". In: *2006 ACM SIGMOD ICMD*. SIGMOD '06. 2006)
- ▶ Hierarchical node clustering scheme (Daniel Keren, Guy Sagy, Amir Abboud, David Ben-David, et al. "Geometric Monitoring of Heterogeneous Streams." In: *IEEE TKDE* (2014))
- ▶ Decouple matching from the data distribution at the nodes (Daniel Keren, Guy Sagy, Amir Abboud, David Ben-David, et al. "Geometric Monitoring of Heterogeneous Streams." In: *IEEE TKDE* (2014))
- ▶ Accurately follow *global statistics vector*
- ▶ Node "cancel each other out" during the *balancing process*

# The Distance-based Hierarchical Clustering

## The Weight Function

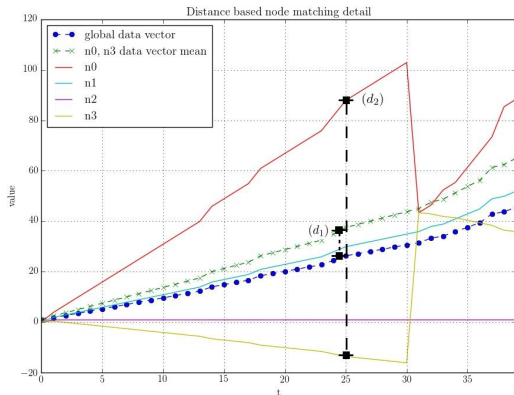
### Weight function

$$w_{i,j} = \sum_{t=t_0}^{t_{end}} [(f(\vec{v}_{global}(t)) - f(\frac{\vec{v}_i(t) + \vec{v}_j(t)}{2})) + (|\vec{v}_i(t) - \vec{v}_j(t)|)]$$

# The Distance-based Hierarchical Clustering

## Example

**Figure:** Distance based node matching operating on 4 nodes ( $\{n_0, n_1, n_2, n_3\}$ ). Distance  $d_1$  : the distance of the data vector mean of the paired nodes  $n_0$  and  $n_3$  from the global mean (*global data vector*), distance  $d_2$  : denotes the in-between distance of data vectors  $\vec{v}_0(t)$  and  $\vec{v}_3(t)$  of the node pair. Both distances are taking part in the edge weighting process.





# The Heuristic Balancing

## The Idea

In prior work:

- ▶ identical handling of nodes during the balancing process
- ▶ ignore stream idiosyncrasies and monitoring function peculiarities

The *heuristic balancing* method:

- ▶ optimally position *drift vectors in space*
- ▶ take into account stream behaviour (*velocity* and *acceleration*)

## The Optimizing Function

$$\max \min \frac{(T - x_i) - accel_i(t_{lv}) * t^2}{vel_i(t_{lv})}, \forall n_i \in P'$$

where:

$P'$  : the balancing set

# The Heuristic Balancing

## Example

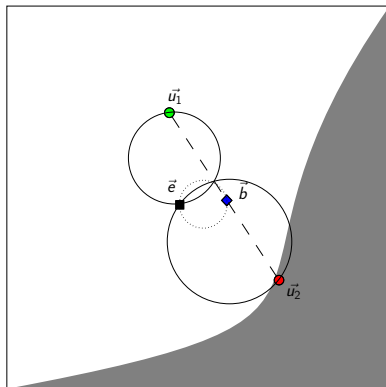


Figure: The classic balancing method.

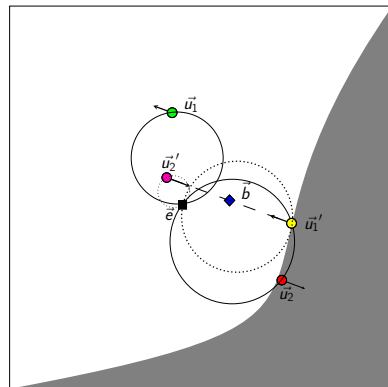


Figure: The heuristic balancing method.

## Introduction

## Theoretical Background

- The Geometric Monitoring Method
- Related Work
- Theoretical Tools

## Problem Statement & Implementation

- Problem Statement
- Implementation

## Experimental Results

- Data & Setup
- Experiments

## Conclusions & Future Work

- Conclusion
- Future Work

# Acronyms for Implemented Methods

GM: the initial geometric monitoring method ([Izchak Sharfman](#), [Assaf Schuster](#), and [Daniel Keren](#). "A Geometric Approach to Monitoring Threshold Functions over Distributed Data Streams". In: *2006 ACM SIGMOD ICMD. SIGMOD '06*. 2006)

HM: the proposed heuristic balancing process

DIST: the proposed distance-based hierarchical node clustering

DISTR: the distribution-based hierarchical node clustering

([Daniel Keren](#), [Guy Sagy](#), [Amir Abboud](#), [David Ben-David](#), et al. "Geometric Monitoring of Heterogeneous Streams." In: *IEEE TKDE* (2014))

# Synthetic Data

- ▶  $v_i(t_{k+1}) = v_i(t_k) + (1 - \lambda)u_k + \lambda u_{k+1}$
- ▶ 1-dimensional
- ▶ Velocities sampled from user-specified Gaussian distribution
- ▶  $\lambda$  smoothing parameter
- ▶ Additive Gaussian noise
- ▶ 3 sets of datasets: *LIN*, *INT*, *NOISE*
- ▶ First 20% of data streams used as training data, when needed

# Synthetic Data

## Examples

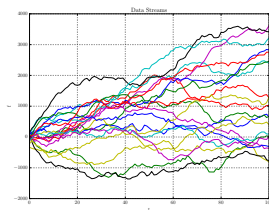
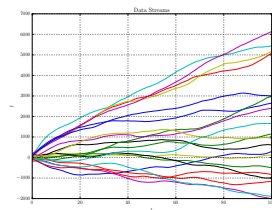
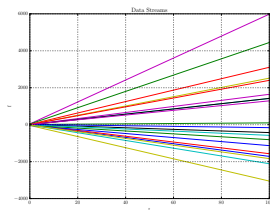


Figure: *LIN* local statistics streams of 20 nodes

Figure: *INT* local statistics streams of 20 nodes

Figure: *NOISE* local statistics streams of 20 nodes

## Real-world Data

- ▶ “European Environmental Agency - AQ e-Reporting” database  
(*European Environmental Agency - AQ e-Reporting.* )
- ▶ Hourly measurements of  $NO_2$  and  $NO$ , in micro-grams per cubic meter, averaged over a window of five days for a whole year.
- ▶ Nodes correspond to randomly selected air quality measurement stations across Austria.
- ▶ First month of measurements used as training data, when needed.



# Real-world Data

## Examples

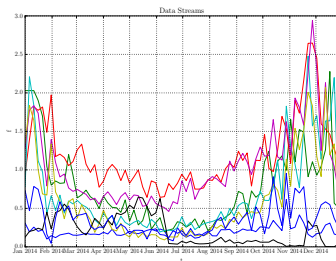


Figure: Streams of 8 nodes monitoring the ratio  $NO/NO_2$ .

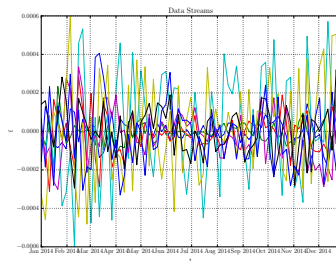
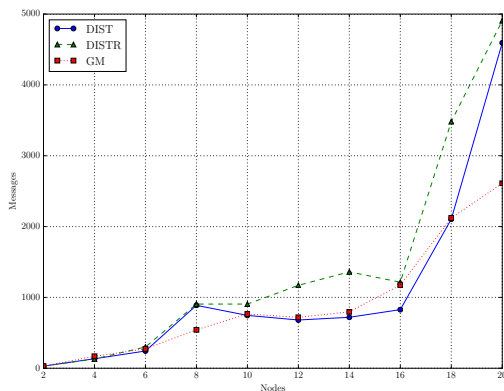


Figure: Streams of 8 nodes monitoring the variance of  $NO_2$  air pollutant.

# GM, DIST, DISTR Comparison

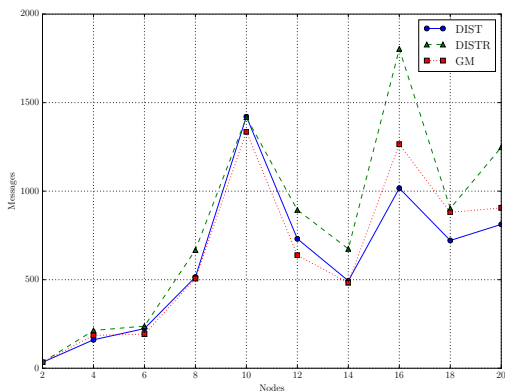
*LIN* dataset



**Figure:** Communication costs of methods GM, DISTR and DIST for the *LIN* dataset.

# GM, DIST, DISTR Comparison

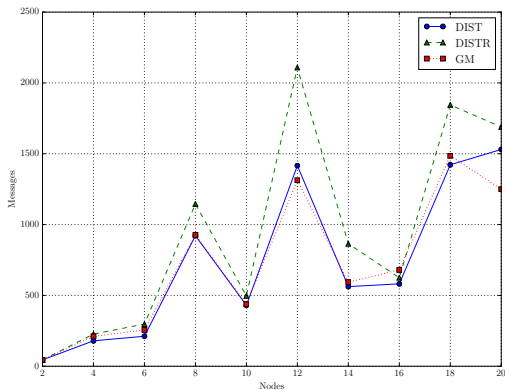
*INT* dataset



**Figure:** Communication costs of methods GM, DISTR and DIST for the *INT* dataset.

# GM, DIST, DISTR Comparison

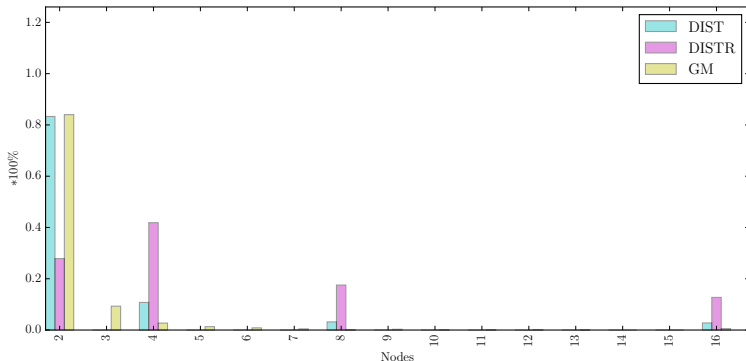
*NOISE* dataset



**Figure:** Communication costs of methods GM, DISTR and DIST for the *NOISE* dataset.

# GM, DIST, DISTR Comparison

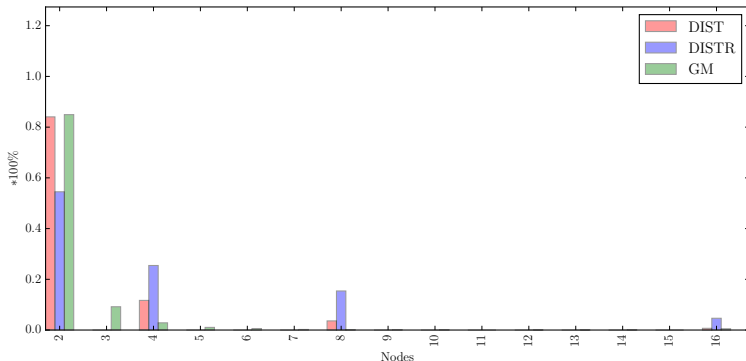
*LIN* dataset



**Figure:** Number of nodes participating in violation resolutions as a fraction of total Local Violations, for the *LIN* dataset.

# GM, DIST, DISTR Comparison

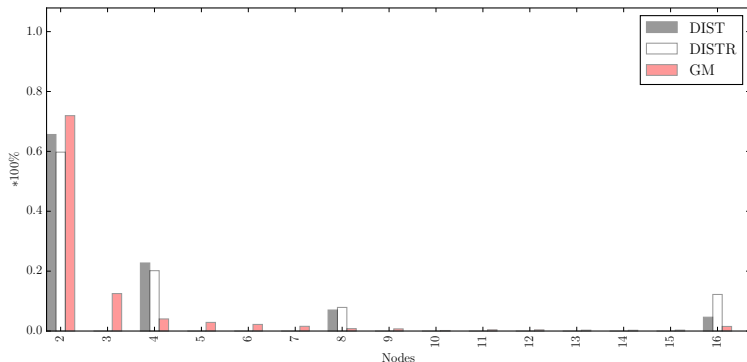
*INT* dataset



**Figure:** Number of nodes participating in violation resolutions as a fraction of total Local Violations, for the *INT* dataset.

# GM, DIST, DISTR Comparison

*NOISE* dataset



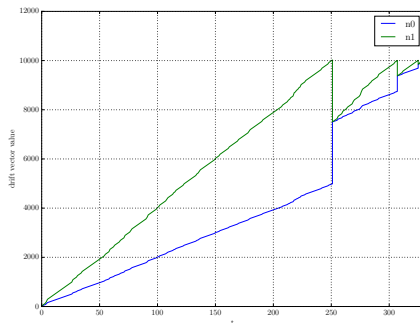
**Figure:** Number of nodes participating in violation resolutions as a fraction of total Local Violations, for the *NOISE* dataset.



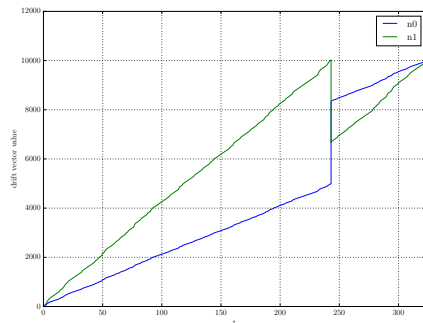


# GM, HM Comparison

## Drift vectors



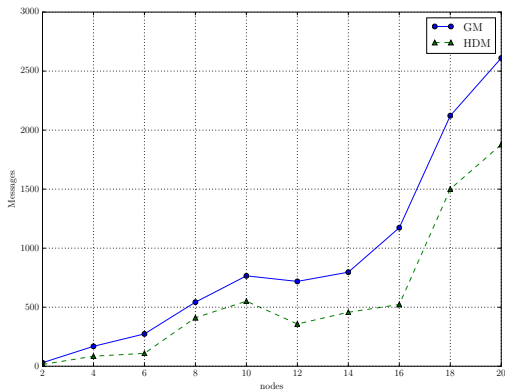
**Figure:** Drift vectors of 2 nodes, as formulated by the GM algorithm.



**Figure:** Drift vectors of 2 nodes, as formulated by the HM algorithm.

# GM, HDM Comparison

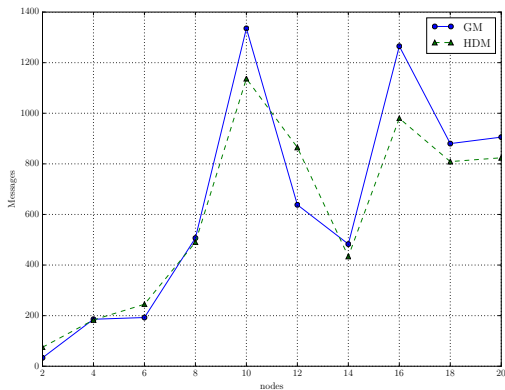
*LIN*



**Figure:** Communication cost of methods GM and HDM for the *LIN* dataset.

# GM, HDM Comparison

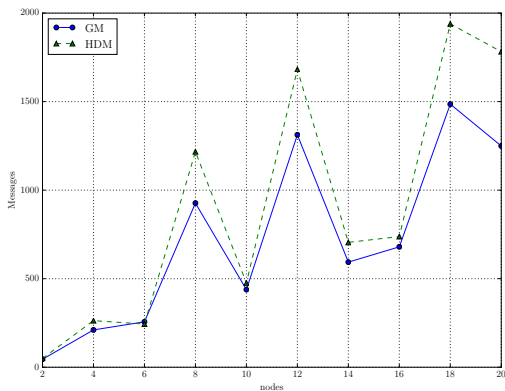
*INT*



**Figure:** Communication cost of methods GM and HDM for the *INT* dataset.

# GM, HDM Comparison

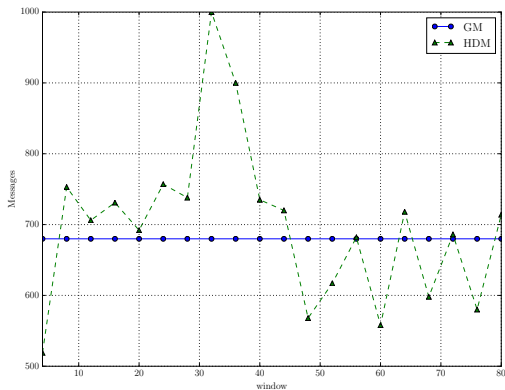
*NOISE*



**Figure:** Communication cost of methods GM and HDM for the *NOISE* dataset.

# GM, HDM Comparison

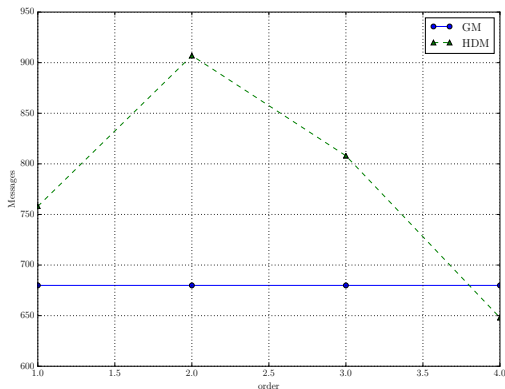
*NOISE* - window



**Figure:** Communication cost of methods GM and HDM for the *NOISE* dataset. Approximation order is set to 1.

# GM, HDM Comparison

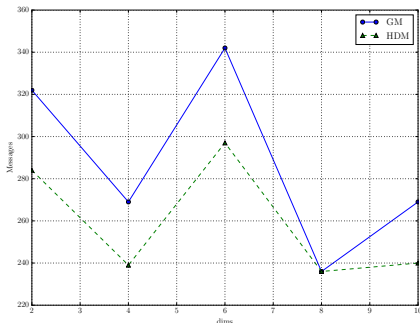
*NOISE* - order



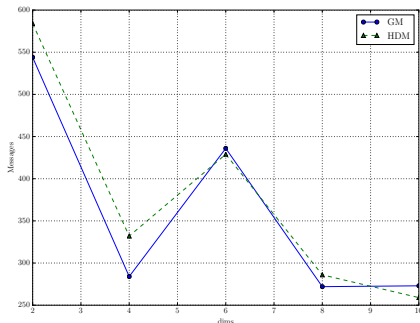
**Figure:** Communication cost of methods GM and HDM for the *NOISE* dataset. The Savitzky-Golay window size is set to 24.

# GM, HDM Comparison

## Dimensions - Quadratic Function



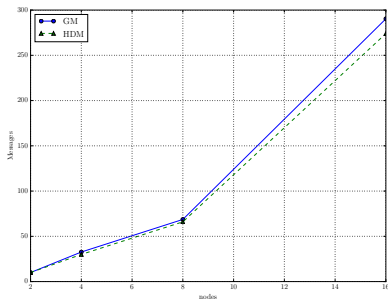
**Figure:** Communication cost of methods GM and HDM for the *LIN* dataset.



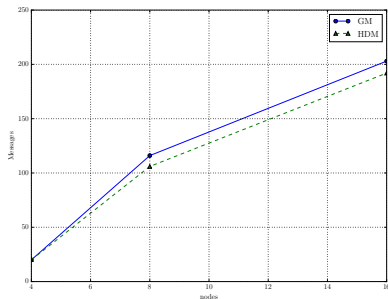
**Figure:** Communication cost of methods GM and HDM for the *NOISE* dataset.

# GM, HDM Comparison

## Air Pollution Monitoring



**Figure:** 4 to 16 nodes, variance monitoring of  $NO_2$ . The Savitzky-Golay window size is set to 6, the order is set to 2.



**Figure:** 4 to 16 nodes, when monitoring  $NO/NO_2$ . The Savitzky-Golay window size is set to 10, the order is set to 1.



## Introduction

## Theoretical Background

- The Geometric Monitoring Method
- Related Work
- Theoretical Tools

## Problem Statement & Implementation

- Problem Statement
- Implementation

## Experimental Results

- Data & Setup
- Experiments

## Conclusions & Future Work

- Conclusion
- Future Work

## Summary & Concluding Remarks

The *Geometric monitoring* method:

- ▶ An efficient framework for monitoring distributed data streams
- ▶ Scalability can be improved → reduce communication costs

Our contributions:

- ▶ Distance-based hierarchical node clustering
- ▶ Heuristic balancing method based on SQP and Savitzky-Golay Filtering
- ▶ Detailed evaluation of proposed methods

Comments:

- + Communication reduction of up to 60%
- + Methods fully compatible with the rest of the work
  - Parameter tweaking for satisfactory results
  - Multi-objective optimization can be computationally expensive

# Future Work

- ▶ Multi-objective optimization solvers
- ▶ More elaborate optimizing fuctions
- ▶ Sophisticated prediction models (Gaussian processes)
- ▶ Parameter estimation techniques

The End

Thank you  
Questions?

Assume a moving object  $i$  at point  $x_i$ , with acceleration  $a_i$  and current velocity  $v_i$ . Let  $v_f$  be the object's final velocity when it reaches a threshold point  $T$  at time  $t$ , from which it deviates by  $d = T - x_i$ . Let current time be  $t = 0$ .

Distance (or *Displacement*) in terms of velocity and acceleration is described by:

$$d = v_i t + a_i t^2 \quad (1)$$

For which it holds:

$$\begin{aligned} d &= v_i t + a_i t^2 \Leftrightarrow \\ T - x_i &= v_i t + a_i t^2 \Leftrightarrow \\ t &= \frac{(T - x_i) - a_i t^2}{v_i} \end{aligned}$$

Thus,  $t$  is the expected time the moving object reaches the threshold point  $T$ .

# Geometric Interpretation

**Theorem** (Izchak Sharfman, Assaf Schuster, and Daniel Keren. "A Geometric Approach to Monitoring Threshold Functions over Distributed Data Streams". In: *2006 ACM SIGMOD ICMD. SIGMOD '06*. 2006)

Let  $\vec{x}, \vec{y}_1, \dots, \vec{y}_n \in \mathbb{R}^d$  be a set of vectors in  $\mathbb{R}^d$ . Let  $\text{Conv}(\vec{x}, \vec{y}_1, \dots, \vec{y}_n)$  be the convex hull of  $\vec{x}, \vec{y}_1, \dots, \vec{y}_n$ . Let  $B(\vec{x}, \vec{y}_i)$  be a ball centered at  $\frac{\vec{x} + \vec{y}_i}{2}$  and with radius of  $\|\frac{\vec{x} - \vec{y}_i}{2}\|_2$  i.e.,  $B(\vec{x}, \vec{y}_i) = \{\vec{z} \mid \|\vec{z} - \frac{\vec{x} + \vec{y}_i}{2}\|_2 \leq \|\frac{\vec{x} - \vec{y}_i}{2}\|_2\}$ , then  $\text{Conv}(\vec{x}, \vec{y}_1, \dots, \vec{y}_n) \subset B(\vec{x}, \vec{y}_i)$ .

# Non-linear Constraint Optimization

## Primal Descent

---

**Algorithm 1:** Generic primal descent

---

```
1 begin
2   Choose initial point  $x_0 \in X$  and set  $t = 0$  ;           /* Initialization */
3   while maximum iteration limit OR convergence do       /* Search */
4      $t = t + 1$ ;
5     Determine search direction  $d_t$ ;
6     Determine step length  $s_t$ , so that  $f(x_t + s_t d_t) < f(x_t)$ ;
7     Update;
8   end
9 end
```

---

# Feasible Directions

*Usable feasible direction  $d_t$ :*

- ▶ a small disposition towards direction  $d_t$  does not violate any constraint i.e.,

$$d_t^T \nabla G(x_t) \leq 0$$

- ▶ a move towards  $d_t$  reduces the objective functions value i.e.,

$$d_t^T \nabla F(x_t) < 0$$

.



---

**Algorithm 2:** Iterative network operation

---

**Data:** *monitoringNodes*: a list of Monitoring nodes,  
*coordinator*: the Coordinator node

```
1 begin
2   initialization;
3   repeat
4     foreach node  $\in$  monitoringNodes do
5       | node.DataVectorUpdate();
6       | node.ComputeDriftVector();
7     end
8     foreach node  $\in$  monitoringNodes do
9       | node.CheckForViolation();
10      | if localViolation then
11        | | node.Report();
12        | | coordinator.Balance();
13      | end
14    end
15  until globalViolation;
16 end
```

---

# The Distance-based Hierarchical Clustering

## The Algorithm

---

**Algorithm 3:** Recursively create Monitoring node pairs and hierarchy

---

```
1 Function DistancePairer(nodes,i)
    Data: nodes =  $[(n_1, [\vec{v}_1(t_0), \dots, \vec{v}_1(t_{end})]), \dots, (n_k, [\vec{v}_k(t_0), \dots, \vec{v}_k(t_{end})])]$ : list of
        nodes with their respective data vectors, i: pair type, initial=1
    Result: nodeHierarchy: dictionary of Type-k pairs
2 if length(nodes) = 1 then                                // recursion stopping condition
3     return nodeHierarchy;
4 end
5 g = CreateCompleteGraph(nodes);                          // complete graph with nodes as
    vertices
6 foreach (ni, nj) ∈ g.Edges() do                        // assign weights to edges
7      $w_{i,j} = \sum_{t=t_0}^{t_{end}} [(f(\vec{v}_{global}(t)) - f(\frac{\vec{v}_i(t) + \vec{v}_j(t)}{2})) + (|\vec{v}_i(t) - \vec{v}_j(t)|)]$ ;
8     g.edge(ni, nj).weight = wi,j;
9 end
10 nodeHierarchy(Type-i) = g.maximalWeightMatching();        // node pairs of
    Type-i
11 DistancePairer(nodeHierarchy(Type-i), i * 2);
12 end
```

---

# The Heuristic Balancing

## The Function Implementation

$$\min -z$$

$$\text{s.t. } z \leq g(h(\vec{e}, \vec{u}_0), vel_0, accel_0, T)$$

$$z \leq g(h(\vec{e}, \vec{u}_1), vel_1, accel_1, T)$$

$$\vdots$$

$$z \leq g(h(\vec{e}, \vec{u}_n), vel_n, accel_n, T)$$

$$\vec{b} = \frac{1}{\sum_{i=0}^n w_i} \sum_{i=0}^n (w_i * \vec{u}_i) \quad , \forall n_i \in P'$$

where:

$g : \mathbb{R}^4 \rightarrow \mathbb{R}$ , the heuristic optimization function

$h : \mathbb{R}^d \rightarrow \mathbb{R}$ , the max val of the monitoring function  $f(\cdot)$  in  $B(\vec{e}, \vec{u}_i)$ ,

$d$  : the data vector dimensionality

# The Heuristic Balancing

## The Algorithm

---

**Algorithm 4:** Heuristic Balancing

---

```
1 Function RepMessageReceived(<  $n_i, v_i, u_i, vel_i, accel_i$  >)  
2   | add  $n_i$  to balancing set  $P'$ ;  
3   | Balance();  
4 end  
5 Function Balance( $P'$ )  
6   | if  $length(P') = 1$  then  
7     | RequestNode();           // request node based on respective gathering scheme  
8   end  
9   |  $\vec{b} = \sum_{P'} \frac{w_i * \vec{u}_i}{w_i}$ ;  
10  | if  $B(\vec{e}, \vec{b})$  is monochromatic then  
11    | /* heuristic optimization procedure,                                     */  
12    | /* returns the optimal drift vector positions in set  $O$                  */  
13    |  $O = \text{DriftVectorOptimizationProblem}()$ ;  
14    | foreach  $n_i \in P'$  do  
15      |  $\Delta\delta_i = w_i * \vec{u}_i' - w_i * \vec{u}_i$ ;      //  $\vec{u}_i'$  denotes the optimal drift vector position  
16      | Send(< ADJSLK,  $n_i, \Delta\delta_i$  >);  
17    | end  
18  | end  
19 end
```