Geoinformatica (2010) 14:101–134
DOI 10.1007/s10707-009-0078-3

Spatial interpolation in wireless sensor networks: localized algorithms for variogram modeling and Kriging

Muhammad Umer · Lars Kulik · Egemen Tanin

Received: 18 May 2008 / Revised: 17 December 2008 /
Accepted: 30 January 2009 / Published online: 24 February 2009

Abstract Wireless sensor networks (WSNs) are rapidly emerging as the prominent technology for monitoring physical phenomena. However, large scale WSNs are known to suffer from coverage holes, i.e., large regions of deployment area where no sensing coverage can be provided. Such holes are the result of hardware failures, extensive costs for redeployment or the hostility of deployment areas. Coverage holes can adversely affect the accurate representation of natural phenomena that are monitored by a WSN. In this work, we propose to exploit the spatial correlation of physical phenomena to make monitoring systems more resilient to coverage holes. We show that a phenomenon can be interpolated inside a coverage hole with a high level of accuracy from the available nodal data given a model of its spatial correlation. However, due to energy limitations of sensor nodes it is imperative to perform this interpolation in an energy efficient manner that minimizes communication among nodes. In this paper, we present highly energy efficient methods for spatial interpolation in WSNs. First, we build a correlation model of the phenomenon being monitored in a distributed manner. Then, a purely localized and distributed spatial interpolation scheme based on Kriging interpolates the phenomenon inside coverage holes. We test the cost and accuracy of our scheme with extensive simulations and show that it is significantly more energy efficient than global interpolations and remarkably more accurate than simple averaging.

Keywords Wireless sensor networks · Coverage holes · Spatial interpolation · Kriging

M. Umer (✉) · L. Kulik · E. Tanin
National ICT Australia, Department of Computer Science & Software Engineering,
University of Melbourne, Melbourne, Victoria 3010, Australia
e-mail: mumer@csse.unimelb.edu.au

L. Kulik
e-mail: lars@csse.unimelb.edu.au

E. Tanin
e-mail: egemen@csse.unimelb.edu.au

1 Introduction

Continuous monitoring of physical phenomena has long been an important application domain for sensing and telemetry systems. Historically, a wide range of satellite based remote sensing and telemetry systems is available for continuous monitoring in various domains [34]. These systems are best suited for low spatiotemporal sampling resolution as they are known to suffer from cost and scalability issues when required to sample with high spatiotemporal resolution. For high resolution sampling, recent advances in hardware miniaturization and cost reduction have given rise to a new class of environment monitoring systems called wireless sensor networks (WSNs) [27]. WSNs comprise a large number of self-contained sensor nodes, each smaller than a human palm, providing an unprecedented capability to sense and monitor the physical world. Ranging from monitoring volcanic activity to microclimate in trees, WSNs have applications in numerous domains [35].

A WSN accomplishes a monitoring task by periodically sampling a physical phenomenon using sensor nodes present at different locations. The cost of production of such nodes is significantly less than the cost of traditional sensing devices. Therefore, it is typically assumed that deployment areas can be completely covered with a high density of nodes. However, experience with WSNs [1] and insights into future applications [23, 28] reveal that situations may arise where the assumption of exhaustive coverage does not hold. The resulting sparse network triggers *gaps* or *coverage holes* in the sample space. We argue that for an accurate monitoring of a physical phenomenon in such scenarios it is important to estimate the phenomenon inside coverage holes. In this paper we investigate such situations and propose novel interpolation methods to augment the reported data in presence of gaps.

1.1 Coverage holes in WSN

The existence of coverage holes in WSNs is commonplace and has motivated a number of recent studies on their detection, effects and management [1, 15, 38]. Typically, coverage holes are assumed to result from hardware failures or communication breakdowns in one or more WSN nodes. However, there is an important class of WSNs where coverage holes are an inherent characteristic of the deployment due to either the *scale* or the *hostility* of monitored area. For example, for a WSN that consists of sensor-enabled ocean buoys which serves as a hurricane or tsunami monitoring system, it might not be possible to deploy sensors with a high density as the network has to monitor a very large area [28]. Despite the lower cost of production of WSN nodes, certain large deployments may still suffer from resolution issues. Similarly, in applications such as NASA's VolcanoWeb [23], it may not be possible to deploy nodes at certain locations due to the hostile operating environment.

Current research has predominantly focused on the identification of holes in order to alert and restore the lost coverage of a WSN [15, 38]. However, the replacement and restoration of nodes may not be sensible in hostile environments or not possible due to prohibitive costs. Therefore, we take a fundamentally different approach to handle the coverage issues in phenomena monitoring systems and view coverage holes as a continuing reality for large WSNs. Our assertion is that coverage

issues require phenomena monitoring regimes that cope with missing data through other means than simply replacing or deploying more nodes. Thus, to improve the monitoring accuracy under various situations, we need methods to interpolate the missing readings based on the available data and application specific data models.

1.2 Spatial interpolation in WSN

Physical phenomena are characterized by spatial correlation, i.e., proximal sampling locations have similar values and vary together. Since WSN nodes sample a phenomenon at the points they are located at, spatial correlation in the data generated by proximal nodes is natural. A number of recent works in the WSN data acquisition domain propose to exploit this spatial correlation to perform energy efficient data collection (e.g., [32]). In traditional monitoring domains, such as remote sensing, estimation methods based on spatial correlation are commonly used to interpolate various spatial phenomena [8]. In this work, we suggest to exploit the spatial correlation in the WSN data to interpolate a phenomenon at locations that are not covered by sensor nodes.

Although simple approximation techniques such as the computation of an average could be applied as spatial interpolation, such methods are often quite inaccurate. For instance, spatial averaging methods, including simple and inverse distance weighted averages, either disregard spatial correlation or assume uniform spatial correlation. Such naive assumptions significantly affect the estimation accuracy in many situations. Consider the contour plots shown in Fig. 1 that represent two physical phenomena of opposing natures. The Sombrero surface [42] represents a phenomenon where spatial correlation is uniform in all directions while the Morrison surface [22] shows different correlation in different directions. Inverse

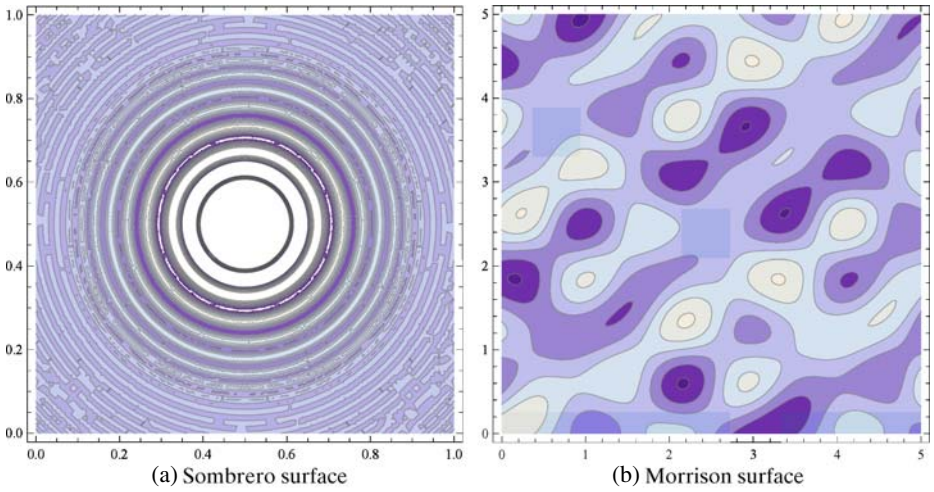


Fig. 1 Contour plots for two mathematical surfaces show different forms of spatial change. See [42] for further details on these surfaces (**a**, **b**)

distance weighted average could be a good interpolator for phenomena that evolve like a Sombrero surface. However, in most practical situations physical phenomena evolve in a more erratic way than uniformly increasing or decreasing in magnitude. Therefore, it is of paramount importance to incorporate the effect of various forms of spatial correlation in the spatial interpolation process. This observation is well-founded in the geostatistical domain where numerous studies show the strength of spatial correlation based interpolation methods over ordinary or inverse distance weighted averaging [42].

1.3 Our contributions

The power consumption of a sensor node is a function of its communication workload [27]. Thus, the sustainability and success of a WSN is tied to its communication efficiency. This important feature transcends the design requirements of all WSN specific techniques, such as routing, data acquisition and monitoring etc., and often traditional counterparts of these techniques do not suit WSNs. In this context, traditional interpolation methods, in particular those that extensively take advantage of spatial correlations, are also not readily applicable to WSNs. These methods either require global knowledge of the network [12, 30] or centralized data processing. Due to WSNs' dynamic nature and large scale it is prohibitively expensive to collect and maintain global information such as node locations, connectivity and sensed data. Similarly, centralized processing is highly energy inefficient as it could require forwarding the samples over a long distance involving a large number of nodes in the communication process.

In this paper, we present a distributed scheme to perform spatial interpolation in a WSN *locally*, i.e., physically closer to a coverage hole using only the data available at neighboring sensor nodes. In contrast to the existing methods, our interpolation technique does not require centralized processing or access to global data about the deployment. The proposed interpolation approach is a novel distributed formulation of well known geostatistical techniques: variogram modeling and Kriging. The proposed approach works in two steps. First, we build a correlation model of a phenomenon and distribute the model in the network. In the second step, the nodes neighboring a coverage hole collaborate and use the correlation model to interpolate the phenomenon inside the hole. Our main contributions can be summarized as follows:

- we propose a distributed data aggregation based approach for spatial correlation modeling in WSNs that outperforms a centralized approach significantly;
- based on the spatial correlation model, we present a distributed and localized formulation of Kriging based interpolation that eliminates the need for centralized processing and global data gathering;
- we perform extensive simulations to establish the strength of our interpolation approach as compared to its global counterpart and a simple averaging based solution.

Data aggregation and interpolation appear to have contradictory objectives; data aggregation summarizes one or more aspects of a dataset while interpolation expands it. Since we propose to apply both techniques, the contributions of this paper may

appear to be conflicting. One may argue that if summarization during the first step (data aggregation) leads to loss of information than the second step (interpolation) may become invalid. However, it is important to note that data aggregation does not always lead to a loss of information. Although some aggregation operators such as average, result in information loss, certain aggregate operations, for instance MIN or MAX, do not lose any information at all. Similarly, a correlation model uncovers valuable information about a dataset and has the potential to enrich the interpolation process. The scope of aggregated information, i.e., the spatial correlation model, in our scheme is to guide the interpolation scheme towards accurate estimation. Therefore, the aggregation and interpolation steps in our approach are complimentary, not contradictory.

The basic idea of the proposed interpolation scheme was introduced in [36]. In this paper, we build upon the basic ideas and introduce several new concepts for their implementation in realistic WSN settings. Furthermore, we present concrete algorithms and perform a thorough experimental analysis.

The rest of this paper is organized as follows: Section 2 introduces the geostatistical techniques used in this work; Section 3 presents our distributed approach towards correlation modeling in WSNs; Our distributed spatial interpolation scheme is presented in Section 4; Section 5 presents an extensive experimental study; Section 6 presents an analysis of simplifying assumptions made in this work; Section 7 discusses related work, while conclusions and future work are presented in Section 8.

2 Preliminaries

In this section, we formalize important concepts in phenomena modeling and interpolation underlying our work.

2.1 Variogram modeling and Kriging

All spatial phenomena observe a certain continuity while evolving in space. Consequently, a correlation between observations at nearby locations is expected. Kriging is a spatial estimation method that exploits this correlation to interpolate unknown values of a phenomenon in the vicinity of known values [17]. In its simplest form, Kriging can be viewed as a weighted average method where the estimation of a physical phenomenon at a location, say x, is a linear combination of the values at locations around x. Considering the spatial correlation of physical phenomena, one could argue that the weight of each value in this linear combination should be based on an inverse geometric distance function. This approach however, is over-simplistic as it assumes *uniform* spatial correlation, which is not the case in nature.

Kriging extends the simplistic inverse distance method by realistic spatial correlation models. For estimating a phenomenon, it uses the statistical distance between locations rather than their geometric distance. For this purpose however, it is first required to define a metric for statistical distance. A typical measure of statistical distance in spatial phenomena is the experimental variogram (EV, $\gamma(h)$) defined as a function of samples of a phenomenon and distance between the corresponding sampling locations [17]. Assume a random variable Z represents a spatial phenomenon

and $Z(x)$ represents a sample at location x then for a given distance h (referred to as *lag*), the EV is defined as:

$$\gamma(h) = \frac{1}{2N(h)} \sum_{N(h)} \left[Z(x) - Z(x+h) \right]^2 \tag{1}$$

where, $N(h)$ is the number of data pairs at distance h.[1]

The spatial correlation in a phenomenon can be observed by computing the EV values for several lags using available samples. However, EV computation alone is not sufficient for Kriging as it often requires correlation between locations where no samples are available. For this purpose, a phenomenon's spatial correlation model is established by fitting a curve onto the computed EV values. The resultant spatial correlation or variogram model is used to find the correlation between phenomenon values at arbitrary locations.

2.2 Kriging computation

Once a variogram model is established for a given phenomenon, it can then be used for spatial interpolation using Kriging. Assume that a spatial phenomenon Z is represented by its realization $Z(x_1), Z(x_2), \ldots, Z(x_N)$ at locations $x_1, x_2, \ldots, x_N$, then the Kriging interpolator of Z at a point x_0 is given by [6]:

$$\hat{Z}(x_0) = \sum_{i=1}^{N} \lambda_i Z(x_i) \tag{2}$$

where λ_i are the weights fulfilling the unbiasedness condition, i.e., $\sum_{i=1}^{N} \lambda_i = 1$ and the expected error is $E[\hat{Z}(x_0) - Z(x_0)] = 0$ [17]. Kriging is an optimal estimator in the sense that it minimizes the estimation variance and is unbiased [6]. It can be shown that optimal weights λ_i for the Kriging interpolator can be computed from the following system of linear equations (SLE) [17]:

$$\Lambda = A^{-1} b, \tag{3}$$

where Λ is a vector comprising of Kriging weights λ_i and a Lagrange multiplier (added for computational reasons), A is the spatial correlation matrix of sample locations $x_1, x_2, \ldots, x_N$ and b is a vector whose elements represent the spatial correlation between x_0 and each $x_i \in \{x_1, x_2, \ldots, x_N\}$. All correlations are based on an appropriate variogram model defined for the spatial phenomenon under observation. The above Kriging interpolator can easily be extended to interpolate the phenomenon in an area, say B instead of a point as in the case above. This requires replacing the x_i to x_0 correlations from matrix $\boldsymbol{b}$ with the correlations between B and each $x_i \in \{x_1, x_2, \ldots, x_N\}$ [17].

[1] In practice, a tolerance of $\pm t$ units in the lag is expected since real-world datasets are generally not uniformly spaced.

Confidence in Kriging Interpolation A major strength of Kriging interpolation is that along with the interpolated value it can provide an estimate of error. The Kriging variance can be established as [17]:

$$Var\left(\hat{Z}(x_0) - Z(x_0)\right) = \left(A^{-1}b\right)^T b, \tag{4}$$

where, $(A^{-1}b)^T$ denotes the transpose of matrix $A^{-1}b$. If the estimation error is assumed to follow a normal distribution, the error interval can be calculated by using the Kriging variance as follows [10]:

$$\text{Error} = \pm t_{\frac{\alpha}{2},n-1}\frac{\sigma}{\sqrt{n}}, \tag{5}$$

where n is the number of samples, σ is the Kriging standard deviation and $t_{\frac{\alpha}{2},n-1}$ is the parameter obtained from t-Student distribution depending on confidence interval $(1-\alpha)$ and degrees of freedom $(n-1)$.

2.3 Choosing an appropriate variogram model

The variogram model for a phenomenon can be established by choosing a model that best fits its EV values. However, in practice this choice is limited by an important feature of the Kriging SLE, i.e., a unique solution of the Kriging SLE is only possible if the Kriging matrix (matrix A in Eq. 3) is non-singular and positive definite [17]. Since the Kriging matrix is built using a variogram model, the above consideration limits one to choose a positive definite model. Therefore, it is not possible to use an arbitrary curve fitted onto the EV values as a variogram model as such a model cannot guarantee positive definiteness of the resulting Kriging matrix. Hence, in practice only some well known functions are used as variogram models. Some typical choices of variogram models are:

– Spherical Model:

$$\gamma(h) = \begin{cases} n + s \times \left(1.5\frac{h}{a} - 0.5\left(\frac{h}{a}\right)^3\right) & \text{for } h \le a \\ n + s & \text{otherwise} \end{cases} \tag{6}$$

– Gaussian Model: $\gamma(h) = n + s \times \left(1 - \exp\left(-\frac{3h^2}{a^2}\right)\right)$

– Exponential Model: $\gamma(h) = n + s \times \left(1 - \exp\left(-\frac{3h}{a}\right)\right)$

where h is the distance between two points and n, s, a are *nugget, sill* and *range* of the EV, respectively; the *nugget* is the height of the jump of the variogram at the discontinuity at the origin, the *sill* is the limit of the variogram (tending to infinitely large distances) and the *range* is the distance for which the difference of the variogram from the sill is minimal.

The above described variogram models parameterize nugget, sill and range in a specific way and hence represent a certain pattern of spatial continuity [17].

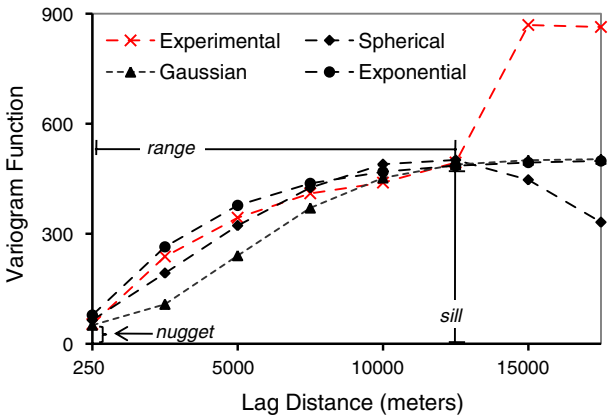


Fig. 2 EV and variogram models for the DEM dataset

For instance, the spherical model represents linear continuity at small distances but flattens out at larger distances. Similarly, the Gaussian model reaches its sill asymptotically and shows a parabolic behavior near the origins hence best suits extremely continuous phenomena. In above variogram models, nugget plays an important role as a large nugget effect represents lack of spatial correlation and hence reduces Kriging procedure to a simple averaging of the available data. For a given phenomenon, one of the above models is adopted based on how closely it matches the actual spatial correlation (observed from its EV values).

Figure 2 shows the omni-directional EV and spherical, Gaussian and exponential variogram models computed for Digital Elevation Model (DEM) dataset [37]. This dataset consists of samples located on a grid of 50 m resolution covering a 10 km^2 area of the state of Colorado, USA. As evident in Fig. 2, the DEM dataset exhibits a high degree of spatial correlation and hence we adopt the Gaussian model in our later experimentation with this dataset.

3 In-network variogram modeling

Variogram modeling is a fundamental step in the interpolation of a phenomenon based on its spatial correlation. Since the primary aim of our work is to distribute the interpolation task among WSN nodes, it is essential to construct and disseminate a variogram model to all nodes in the network. In a WSN, a simple approach for variogram modeling could construct the EV of a phenomenon by propagating the data from each WSN node to a base station and compute the EV centrally. However, such a global approach is guaranteed to suffer from excessive communication costs. We aim to replace the need for centralized processing and reduce communication costs by distributing the task of EV construction in the network. For this purpose, we model the construction of a phenomenon's EV as an aggregation query and propose a distributed solution that is based on in-network aggregation.

In-network aggregation schemes [21] for WSNs are motivated by two major factors. First, the nature of communication in a WSN is multihop, i.e., data from a node cannot be directly transmitted to the base station and hence follows a path

through other nodes in the network. Second, a node consumes significantly less energy for information processing than for communication. Therefore, in-network aggregation does not transmit the data of each individual node separately. Instead, a node first aggregates the incoming data from the nodes in its communication range and transmits only the aggregated information. Typically, in-network aggregation establishes a data collection tree rooted at the base station spanning all nodes in the network. An aggregation operator (such as AVERAGE) is then decomposed and nodes are synchronized such that each internal node waits for incoming data from its child nodes. Internal nodes perform partial aggregation on the incoming data and communicate only the answer to their parent nodes. An iterative use of this scheme by each internal node results in computation of correct aggregates and reduces the communication cost significantly.

One of the fundamental challenges faced by the in-network aggregation paradigm is the decomposition of aggregation operators. A typical tree-based in-network aggregation scheme can compute local aggregation operators, such as SUM or AVERAGE. However, exact computation of global aggregation operators, such as MEDIAN, requires to gather all the data at a central point. Therefore, in-network aggregation algorithms rely on localized approximation methods for such operators (e.g., [31]). We face a similar challenge while modeling the EV construction as an in-network aggregation problem. To explain this challenge further, we first formulate an aggregate query that constructs an EV in a WSN.

3.1 An aggregation query for experimental variogram construction

Our aim is to solve the EV construction problem in a distributed manner using an in-network aggregation plan. Following the WSN database query models, we address the EV construction as an aggregation query:

```
SELECT lags.distance, AVG(POWER(s1.value-s2.value,2))
FROM sensors AS s1, sensors AS s2, lags
WHERE
ABS(DISTANCE(s1.location, s2.location)-lags.distance) ≤ t
GROUP BY lags.distance
```

where sensors represents the table of sensor values for the phenomenon under observation, lags represents a table of required lag values and t is the tolerance level to accommodate the non-regular spacing between nodes. In a WSN, the sensors table can be viewed as being split across all nodes in the network such that each node only possesses the tuples that it generates. We assume that the table lags is distributed in the network and each node possesses the complete table.

Figure 3 presents the basic idea behind in-network execution of the above experimental variogram self-join (EVSJ) query. Leaf nodes communicate their sensed values and locations to parent nodes where partial aggregation is performed. In the case of EVSJ query, the partial aggregation refers to computing the pair-wise count and sum of squared differences of child node values for each lag, as shown in Fig. 3. All internal nodes suppress child nodes' data from further traversal and only transmit the partial aggregate along with their own values and location information. Final EV values for each lag can then be calculated at the root using partial aggregates.

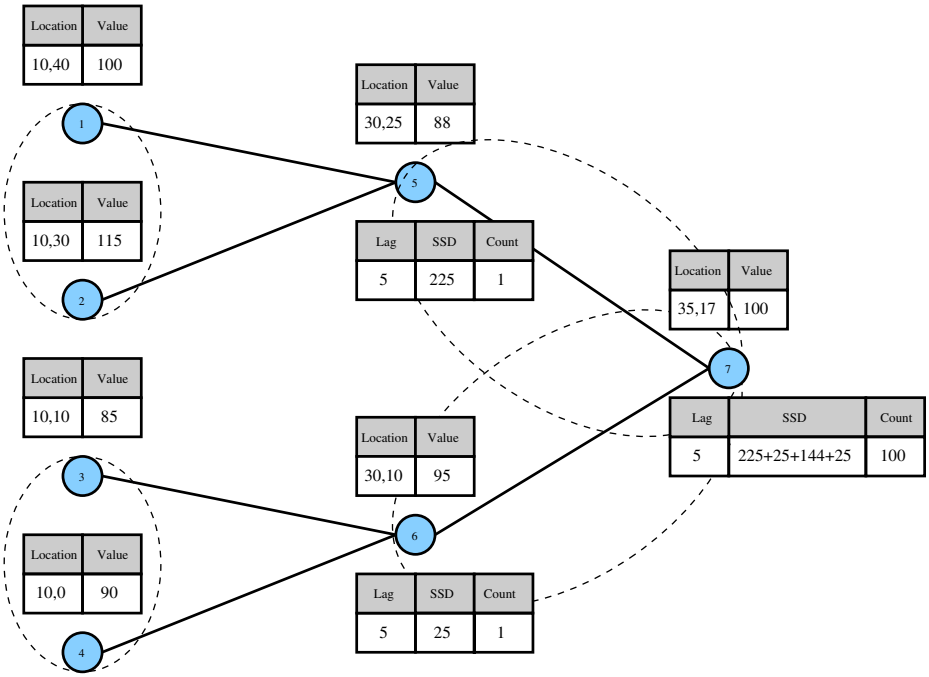


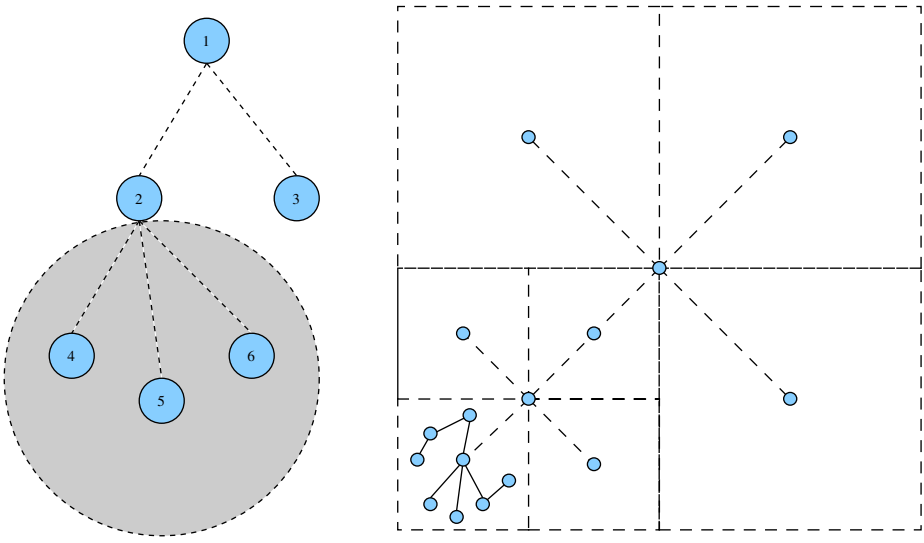
Fig. 3 In-network execution of the EVSJ query comprising a single lag (5 units). *Dashed circles* represent node pairs within a distance of 10 units to each other. Partial aggregate at internal nodes consists of tuples of the form: (lag, sum of squared difference (*SSD*) and count of node pairs)

The main challenge in the partial aggregation of the EVSJ query is the involvement of self-join on the table `sensors` because it requires the value of each node to be aggregated with all nodes that are within a certain distance. Therefore, an internal node can aggregate and suppress incoming data from a child node, say A, only if it receives the data from all nodes located within a distance of *lag* units from A.

Consider the example presented in Fig. 4a. For the EVSJ query, node 5's data is required to be aggregated with all nodes present within a certain distance, referred to here as node 5's *EV neighborhood* (as shown by the shaded circle in Fig. 4a). The nodes located inside node 5's EV neighborhood (nodes 4 and 6) do not have a communication link with it. Therefore, node 5's EVSJ aggregation cannot be fully performed until the data from itself and nodes 4 and 6 reaches a common parent node on the data aggregation tree. Clearly, if the common node is reached quickly, the energy savings will increase because node 5's data can be stopped from traversing up in the tree further. In this example, node 2 can aggregate node 5 as it fully covers node 5's EV neighborhood. However, the key challenge is to determine whether or not a given internal node (e.g., node 2) covers EV neighborhoods of its child nodes.

3.2 The quad suppress (QS) algorithm

In general, the in-network computation of the EVSJ query has the following challenge: each internal node should locally determine if it spans all nodes in the EV



(a) An example data aggregation tree for in-network computation of the EVSJ query

(b) Aggregation tree creation in the QS algorithm

Fig. 4 In-network execution of the EVSJ query using the QS algorithm (**a**, **b**)

neighborhood of its child nodes. These nodes can either aggregate child nodes' data or forward it in an unaggregated form. Moreover, to reduce the data communication costs, unaggregated data forwarding should be minimized.

To address this challenge, we present the Quad Suppress (QS) in-network aggregation algorithm. The QS algorithm comprises two main parts: (i) aggregation tree creation and (ii) data aggregation. The following subsections detail these components.

3.2.1 Aggregation tree creation

Typically, a WSN data aggregation tree is created in an *uncontrolled* or random manner [21]. The base station broadcasts a tree creation message and all receiving nodes join as its child nodes, forming the first level of the tree. These nodes in turn extend the tree by re-broadcasting the message to nodes further away from the base station. The message re-broadcast is continued until all nodes in the network have joined the aggregation tree. In this tree creation method, internal nodes have no control over the nodes that join their subtrees. Consequently, without an extensive message exchange an internal node cannot determine if it spans a given subset of nodes forming the respective EV neighborhoods of its child nodes.

In the QS algorithm, we propose to create the aggregation tree in a *controlled* and regionalized manner such that each internal node grows its subtree within a specific region. The EV neighborhood of a node is defined by its location and a predefined lag value. Therefore, in a regionalized tree an internal node can determine if it completely spans the EV neighborhood of a child node using only the child node's

Algorithm 1: QS tree formation algorithm for node n

```
// M: QS tree formation message comprising of:
// s_M: node id of message sender
// Q_M: a quadrant represented by a rectangular region
// l_M: node s_M's quadtree level (if s_M is a quadhead)
// C_M: expected location of level l_M + 1 quadhead in the quadrant
   Q_M (a fixed size rectangular region cocentric with Q_M)
// t_M: message type (quadtree creation message or random
   aggregation tree creation message)
```

Input: QS tree formation message M
Output: QS tree formation message M

```
1  if t_M = quadtree creation message then
2      if n is located inside C_M then
3          Save node s_M as level l_M quadhead
4          Set n as level l_M + 1 quadhead
5          a ← Area(Q_M)/4
6          if a is less than predefined quadtree resolution then
7              t_M ← random aggregation tree creation message
8              Broadcast M
9          else
10             l_M ← l_M + 1
11             s_M ← n
12             t_M ← quadtree creation message
13             Q' ← Q_M
14             foreach quadrant q in Q' do
15                 Q_M ← q
16                 Update C_M to be cocenteric with q
17                 Geocast M to C_M
18     else
19         Geocast M to C_M
20 if t_M = random aggregation tree creation message then
21     if n is inside Q_M and n has no parent then
22         Set s_M as aggregation tree parent
23         s_M ← n
24         Schedule partial aggregation transmission time period
25         Broadcast M
```

location information and predefined lag values. For regionalized tree creation, we propose a quadtree-based process that works as follows:

- the base station partitions the deployment space into four quadrants and *geocasts*[2] a quadtree creation message to constant sized regions defined around the centroids of each quadrant;
- nodes forward the geocasted message until it reaches the first node located inside the geocasted region (as shown in Fig. 4b we expect to have one such node in each quadrant and refer to these nodes as level 1 *quadheads*);
- each quadhead further partitions its quadrant and determines new subregions defined around the respective centroids of each partition;

[2] A routing strategy used in WSNs and Mobil Adhoc Networks (MANETs) that delivers data to all nodes located inside a specific region.

- quadheads geocast the quadtree creation messages to new subregions, received by level 2 quadheads;
- level 2 quadheads continue to recursively partition their corresponding quadrants and the partitioning and geocasting process terminates when a predefined quad-tree resolution is reached;
- final level quadheads create random aggregation trees in their respective quadrants if there are many nodes in that quadrant.

The tree creation scenario explained above is illustrated in Fig. 4b, while Algorithm 1 lists the process for an arbitrary node n. Upon receiving the QS tree formation message, node n first determines the message type as either quadtree or random aggregation tree creation message. For a quadtree creation message, node n first determines if it is the intended quadhead (Algorithm 1 line 2). If node n is the intended quadhead and it finds the partitioning level to have reached the predefined resolution, it stops the quadtree creation process and starts a random aggregation tree (lines 6 to 8). Otherwise, it further partitions the current quadrant and geocasts the tree formation messages to next level quadheads (lines 14 to 17). If node n does not fulfill the quadhead criteria, it simply geocasts the message to the intended quadhead (line 19). If a random aggregation tree creation message is received, node n first checks if it has not joined the tree yet and is located in the right quadrant (line 21). If this is the case, it joins the tree and broadcasts the message to its neighboring nodes in order to extend the tree further (lines 22 to 25).

3.2.2 Data aggregation

After creating the aggregation tree, the QS algorithm uses a phased approach for data aggregation. Assume that the tree creation results in a tree with i levels. In the first phase of data aggregation, ith level quadheads gather sensed data and location information from all direct child nodes. Quadheads also gather data from indirect child nodes if they head a random data collection tree. After receiving all the data, each quadhead scans the incoming data for nodes with EV neighborhoods completely inside its quadrant. Data from all such nodes is aggregated at this level and suppressed from further traversal.

It is expected that the ith level quadheads may not cover the EV neighborhoods of all nodes in their quadrant. This is caused by a node's location, for instance, a quadhead may not be able to cover the EV neighborhood of a node located on the border of its quadrant. Consequently, quadheads are required to transmit the data from such nodes in unaggregated form. In the second phase of data aggregation, ith level quadheads transmit the partial aggregates and any unaggregated data to the quadheads at level $i-1$. These quadheads perform further aggregation and reduce the unaggregated data as a large number of previously uncovered nodes will now have their EV neighborhood covered. The resultant partial aggregates are forwarded to the quadheads at level $i-2$. The aggregation and data reduction process is repeated at each level of the tree until all data reaches the base station.

Once all aggregated data reaches the base station, a variogram model can be fitted to it and Kriging parameters such as nugget, sill and range can be estimated. If this variogram model is intended for use at node level in a decentralized way, the base station simply broadcasts the model to all nodes in the network.

Algorithm 2: QS data aggregation algorithm for node n

```
// t: time period for which node n listens for child nodes' data
// M: QS data aggregation message comprising of:
// s_M: node id of message sender
// UAGG_M: unaggregated information at s_M as a list of tuples of
   the form (node id, location, sensed value) belonging to nodes
   whose EV neighborhood is not covered by s_M
// AGG_M: result of partial aggregation at s_M as list of tuples of
   the form (lv_t, SUM_t, COUNT_t) where lv_t is a lag value, SUM_t is
   the sum of squared differences of node values for node pairs
   located at a distance of lv_t to each other and COUNT_t is a count
   of node pairs forming SUM_t
```

Output: QS data aggregation message M

```
1  Initialize QS data aggregation message M
2  s_M ← n
3  l ← location of node n
4  v ← sensed phenomenon value at node n
5  Add tuple (n,l,v) to UAGG_M
6  while t has not expired do
7      Listen for QS data aggregation messages from child nodes
8      Update M's UAGG_M and AGG_M lists using corresponding lists from each received message
9  if node n is not a quadhead then
10     Transmit M to parent node
11 else
12     foreach tuple i in UAGG_M do
13         Compute EV neighborhood E of node in tuple i
14         if node n's quadrant encloses E then
15             foreach tuple j in UAGG_M do
16                 if i ≠ j then
17                     d ← distance between nodes in i and j
18                     Find a tuple t from AGG_M such that |d − lv_t| ≤ tolerance
19                     SUM_t ← SUM_t+ squared difference between sensed values in i and j
20                     COUNT_t ← COUNT_t + 1
21             Remove i from UAGG_M
22     Transmit M to quadhead one level higher
```

Algorithm 2 formally presents the QS aggregation process for an arbitrary node *n*. In the first phase, nodes simply forward the data received from each child node towards the quadhead in unaggregated form using the previously computed random aggregation tree (Algorithm 2, lines 1 to 10). Since this tree is not created in a regionalized manner, in-network aggregation for the EVSJ query is not started until the data reaches a quadhead. If node *n* is a quadhead it attempts to partially aggregate and hence reduce the unaggregated information. The partial aggregation has the following steps: node *n* scans the incoming unaggregated data for all nodes with an EV neighborhood completely within node *n*'s quadrant (lines 12 to 14); for each pair of nodes (i,j) in the unaggregated data, where i is a covered node and j is located inside i's EV neighborhood, node *n* computes the distance between i and j (lines 12 to 17); it then determines the appropriate lag value lv depending on the computed distance and updates the partial sum of squared differences and pair counts corresponding to lv (lines 18 to 20). After aggregating the covered nodes data, node *n* removes it from the unaggregated data list (line 21). After analyzing all

incoming data, node *n* forwards the aggregated message to the quadhead one level above itself. The same process is repeated at each quadhead.

3.2.3 Analysis

Cost comparison Figure 5 shows a significant difference in energy expenditure of the QS algorithm and a random tree based global data collection using 2500 samples from the Digital Elevation Model (DEM) dataset covering a 500 m^2 area of the state of Colorado [37]. The difference in the performance of the two algorithms can be explained by their data forwarding behavior. The first data collection scheme creates the aggregation tree randomly, thus an internal node cannot determine whether or not it covers the EV neighborhood of its child nodes. Consequently, all internal nodes propagate their data to the base station in unaggregated form. On the other hand, the QS algorithm reduces the communication costs significantly by performing the aggregation in the network using location information as soon as a node's EV neighborhood is covered.

Performance tuning and accuracy An important parameter in the QS algorithm is the resolution of the final level of the quadtree. We propose to set the value of this parameter according to the expected EV range of a phenomenon, i.e., the distance at which the spatial correlation in the phenomenon values is expected to become insignificant. If this resolution is kept too small, a large number of nodes from one cell will be required to be aggregated with many nodes from neighboring cells. Consequently, only limited aggregation would be possible in the lower levels of the tree. Similarly, a very coarse resolution will produce larger cells and still reduce the benefit of aggregation. A good balance can be found by setting the resolution equal to the expected EV range.

The accuracy and quality of variogram modeling using the QS algorithm depends on the large scale correlation structure of a target phenomenon. Since, the QS algorithm computes the variogram by considering pairs of values within a certain distance (the expected EV range), it cannot reflect any large scale correlation structure. A centralized method, on the other hand, can analyze node pairs present

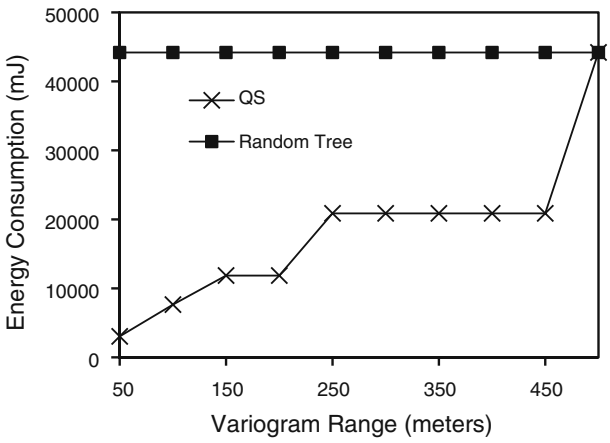


Fig. 5 Comparison of energy expenditure in QS and random tree algorithms for EV construction on 2500 samples of the DEM dataset. Energy expenditure is measured as a function of number of messages and byte content per message (see Section 5 for details on energy computation)

at any distance and hence can reflect both short and large scale correlation well. In practice however, most physicals phenomena have strong local spatial correlation. Therefore, for most situations the accuracy of variogram model established by the QS algorithm is expected to be close to that reached by centralized methods. In Section 5, we experiment with this aspect of the QS algorithm in detail and analyze its effect on interpolation accuracy in different situations.

4 Distributed Kriging (DISK) algorithm

In this section we explain our interpolation approach, the DISK algorithm, which is a distributed computation of the Kriging interpolation.

Equation 3 shows that the matrix inverse calculation poses a major bottleneck in the Kriging operation. To reduce this computation usually a restricted neighborhood method, referred to as Kriging with moving window, is employed where only the sample points within a certain distance from the interpolation point are used in the Kriging matrix. We argue that in large scale WSN settings, the moving window approach calls for a localized Kriging computation (as according to our experiments, pumping all window data to a base station several hops away proves to be expensive). Similarly, naive distributed approaches that assume the presence of global knowledge at each node are not practical as well, since there is no reliable and inexpensive way of maintaining global network information on sensor nodes. In this context, a localized approach replaces the need of global knowledge with knowledge sharing among neighboring nodes and becomes increasingly efficient as the difference between the size of neighborhood (window) and the size of network grows.

4.1 Formulation

The basic building block of the DISK algorithm is an iterative approach of the Gaussian elimination method that allows us to localize and distribute the solution of Kriging SLE presented in Eq. 3. In terms of Kriging SLE, the Gaussian elimination method can be interpreted as the process of finding a sequence of elementary row operations, or linear maps, that transforms matrix A to its reduced row-echelon form. Applying the same linear maps on matrix b would then yield to the solution of Kriging SLE. The aim of the DISK algorithm is to compute the Kriging operator in a recursive fashion so that the process of finding the linear maps can be distributed among the nodes participating in the Kriging estimation.

The basic idea of our iterative elimination approach is presented in Fig. 6. If the nodes inside a Kriging window are assumed to be aligned along a chain, each node k adds a new variable, i.e., its Kriging weight (λ_k), and its corresponding linear combination ($\sum_{i=1}^{k} \lambda_i c_{ki} = b_k$) to the Kriging SLE. In terms of matrix operations, each node in the chain adds a new row and column to the matrix A required to be inverted while not changing the original entries. We can then order the elimination process on the basis of following recursive formulation of matrix A:

$$A_k = \begin{bmatrix} 1 & K^T \\ K & A_{k-1} \end{bmatrix} \tag{7}$$

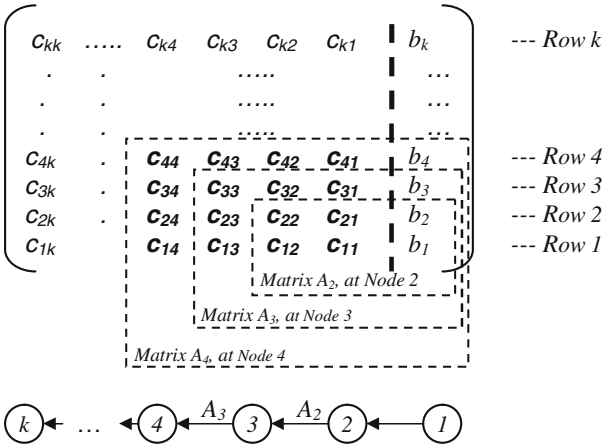


Fig. 6 Iteratively building Kriging SLE

where $k \geq 2$ and

$$K = \begin{bmatrix} c_{k(k-1)} \\ \dots \\ c_{k1} \end{bmatrix} \quad A_2 = \begin{bmatrix} c_{22} & c_{21} \\ c_{12} & c_{11} \end{bmatrix}$$

Now for A_k, we define $\Phi(T_k) : A_k \rightarrow A_k^{\Phi}$ as a group of all linear maps enumerated by the following recursive definition:

$$\Phi\left(T_{k-1}\right) \tag{8}$$

$$R_k \leftarrow R_k + (-k_{1i} \times R_{k-i}) \; \forall i \in \{1, 2 \dots k-1\} \tag{9}$$

$$R_k \leftarrow R_k \times \frac{1}{a_{11}} \tag{10}$$

$$R_{k-i} \leftarrow R_{k-i} + (-k_{i1} \times R_k) \; \forall i \in \{1, 2 \dots k-1\} \tag{11}$$

where,

- $k \geq 2$.
- R_i represents the ith row of A_k and a_{11}, k_{i1}, $k_{1i}, i = 1, 2 \dots k-1$ represent elements of matrices A_k and its corresponding K and K^T constituent vectors, respectively.
- $\Phi(T_{k-1})$ represents all linear maps defined for matrix A_{k-1}.
- $\Phi(T_1)$ comprises one row operation defined as: $R_k \leftarrow R_k \times \frac{1}{a_{11}}$.

An immediate consequence of above formulation can be specified as the following Lemma:

Lemma 1 *If $A_{k-1} = I$, the identity matrix, and A_k defined in terms of Eq. 7, then $A_k^{\Phi} = I$.*

Theorem 1 *Let A_k be an invertible matrix and $\Phi(T_k) : A_k \rightarrow A_k^{\Phi} = I$ be the group of linear maps corresponding to A_k. The solution of SLE $\lambda = A_k^{-1} b$ can be obtained by a recursive application of linear maps from $\Phi(T_k)$ on b.*

The above formulation allows us to find the linear maps required to solve the Kriging SLE in an iterative manner. Consider the example in Fig. 6. The first intermediate node (2) in the chain initiates the iterative process by computing the required transformations for A_2 and transmit the composite linear map, T_2, to the node above it (node 3). Node 3 computes the row and column entries for A_3 and according to the definition above, first applies the received linear map, T_2 on A_3 followed by linear maps that reduce all new row entries to 0, reduce the pivot element to 1 and reduce all new column entries to 0, i.e., applying Eqs. 9, 10 and 11, in that order. Node 3 then forwards the composite linear map T_3 to the next node in chain. The iterative application of the same procedure at each intermediate node in the chain results in a final composite linear map, T_k at root node of the chain. T_k can then be applied on vector b to compute the solution of the Kriging SLE resulting in the required Kriging weighting vector: Λ.

As a byproduct of above process, Kriging variance and hence a confidence interval for a Kriging estimate can be computed easily. As shown in Eq. 4, once vector Λ is computed, the Kriging variance can be computed simply as $\Lambda^T b$.

4.2 DISK algorithm implementation

In this section we outline the sequence of operations required to implement the DISK algorithm in actual WSN settings. In general, our DISK algorithm based interpolation scheme works through the following steps:

- the base station broadcasts a query specifying the phenomenon to be sampled;
- if coverage holes are not known in advance, the network applies a discovery method, such as [20, 38], to identify nodes bordering a coverage hole;
- border nodes disseminate the coverage hole knowledge to all nodes within a threshold distance to the coverage hole;
- all nodes that receive the coverage hole knowledge along with the border nodes form a *Kriging neighborhood*;
- finally, DISK is used by all nodes in the Kriging neighborhood to interpolate the values inside the hole area.

We divide the coverage hole area in a grid of size $h \times v$, where h and v are the number of grid cells along horizontal and vertical axis, respectively. We use a moving window model for Kriging the phenomenon value for each grid cell [17]. The value for a grid cell $x_{ij}, 0 < i \leq h, 0 < j \leq v$ is interpolated using only the nodes that fall inside the Kriging window defined by a search ellipse centered on x_{ij}. We assume that the variogram model of the phenomenon is distributed in the network during a previous variogram modeling phase using the QS algorithm.

The basic idea behind DISK is to distribute the Kriging computation among the nodes in a neighborhood such that each node only performs part of the overall processing. Our iterative computation requires the nodes to be organized in a communication chain. Such a chain ensures that only one node is added to the Kriging SLE at each intermediate stage and the diagonalized matrix of the preceding stage remains intact. As a result, a node at level i in the chain has to compute new linear maps for $2i$ entries it adds to the Kriging SLE. The challenge is to create this communication chain reliably in absence of global information such that all nodes inside the Kriging neighborhood are included.

We propose a novel tree-based method that performs chain processing required for iterative elimination. The nodes in the Kriging window are organized in a tree, referred to as *DISK tree*, rooted at an arbitrary border node. To save from setup costs, border nodes start establishing the DISK tree while disseminating the coverage hole information. The root node starts the DISK procedure by initiating a depth-first tree traversal. In general, an intermediate node k keeps forwarding the Kriging data it receives from one of its traversed children to untraversed children until all child nodes are traversed. At this stage the node k computes A_k (as defined in Eq. 7) and corresponding composite linear map T_k. Node k then forwards all composite linear maps, its position and the observed value of the phenomenon to its parent node.

Figure 7 shows an example DISK tree spanning 10 nodes. In this example, the initialization request follows the path from node 100 to 110 terminating at node 109. Leaf node 109 initializes the Kriging SLE with its values and sends the message back to node 110. Consequently, node 110 forwards the received message to its next child, i.e., node 115. Since node 115 is not a leaf node it forwards the message that is finally received by node 116. Node 116 performs the computations described in Section 4.1 to extend the current Kriging system and sends the extended system back to node 115. As node 115 has no untraversed child left, it extends the Kriging system further by adding its value and performs the computations. The same process is repeated at each internal and leaf node of the tree until the root node 100 receives the data and has no more untraversed child left. The chain in Fig. 7 shows the sequence of nodes that extend the Kriging SLE.

Algorithm 3 details the steps to compute DISK for an arbitrary node n. Algorithm 3 assumes that a DISK tree is already created in the Kriging neighborhood

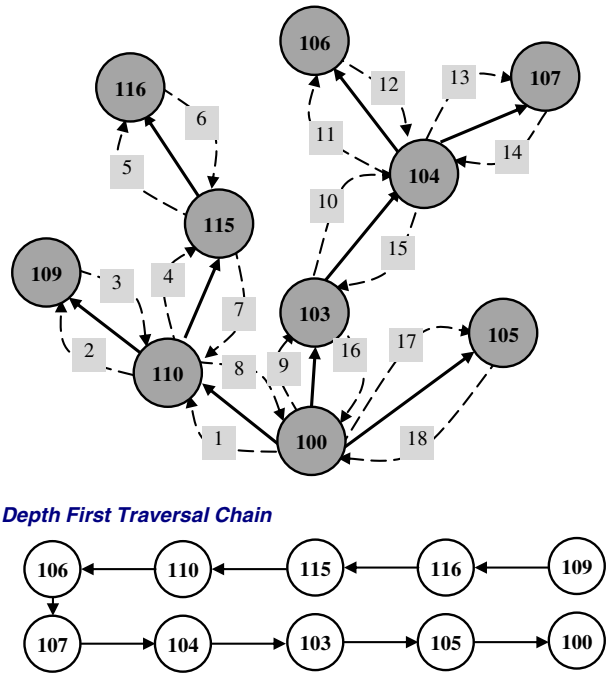


Fig. 7 Execution of the DISK algorithm on a DISK tree. *Dashed arrows with labels* show sequence of messages passed in the depth first traversal of the DISK tree

where node n is located. Lines 1 and 2 of Algorithm 3 implement the depth first traversal of the DISK tree. If node n has traversed all child nodes (or if it has no child nodes), it iterates through the list of grid cells formed for the interpolation of the coverage hole (line 3). For each grid cell that has a Kriging window enveloping node n, it extends the corresponding Kriging system using the recursive procedure explained in Section 4.1 (lines 6 to 12). If n is not the root of DISK tree, it simply forwards the updated Kriging data to its parent node. Otherwise, it computes the final interpolation estimates for each grid cell using Theorem 1 (lines 15 to 19). It then forwards the interpolation estimates to the base station (line 22).

4.3 Cost analysis

The worst-case time complexity of DISK for a given Kriging system at an individual node is $O(N^2)$, where N is the Kriging size or number of nodes in the Kriging system. The number of operations performed by a node present at level k in the DISK chain is equal to the total number of linear maps in T_k enumerated by Eqs. 8–11. As explained in the algorithm above, the node at level k in the chain will first execute all linear maps sent to it from the node at level $k-1$. It then computes linear maps for $2(k-1)+1$ new values it adds to the Kriging matrix in the form of one new row and one new column. The total number of linear maps in T_k, denoted by $|T_k|$, can be given as:

$$|T_k| = |T_{k-1}| + 2k - 1$$

Here, $k \geq 2$ and the base case for the recursion is give by $T_1 = 1$, i.e., the number of linear maps computed by the first node of DISK chain. Solving the above recursion yields

$$|T_k| = k^2$$

The aggregate complexity of DISK computation over N nodes will then be $\sum_{i=1}^{n} |T_i| \approx O(N^3)$.

For an m node neighborhood, the total number of messages generated during the execution of DISK algorithm is $3m$, i.e. m messages transmitted during the tree creation and $2m$ transmitted during the depth-first traversal.

5 Experimental study

We performed extensive simulations to evaluate the performance of the DISK algorithm with respect to the size of the network, the size of the Kriging neighborhood, node to base station distance and node density. A fundamental goal of this experimentation is to demonstrate the scalability of our approach in large deployments. Therefore, we experiment with data sets that are large both in terms of node density and volume. In this paper, we present the results of simulations based on two datasets: a Digital Elevation Model (DEM) dataset from the state of Colorado [37] and simulated traffic data for the city of Melbourne, Australia. Both datasets represent large WSN deployments, i.e, network sizes from approximately

1000 to 5000 nodes with deployment area ranging between 8 km^2 to 18 km^2 for DEM data and 16 km^2 to 36 km^2 for traffic data.

An important consideration in using DEM and traffic data for this experimentation is to show the useability of our interpolation approach at the opposite ends of the spectrum, i.e., a phenomenon that is highly spatially correlated on one end and highly dynamic road traffic data on the other end. Contour plots in Fig. 8 clearly show the opposing nature of these datasets. Smooth contours of DEM data (Fig. 8a) show strong local spatial correlation in this dataset. On the other hand, noisy contours of traffic data (Fig. 8b) show extremely limited small-scale spatial correlation. The traffic data does show the presence of large-scale spatial correlation, a fact that represents the nature of road traffic. For instance, the traffic level on a major arterial road in a city may be quite different from the level on an adjacent suburban road. However, similar traffic levels are usually recorded on arterial roads that may be at a larger distance to each other but connect the same parts of the city.

Algorithm 3: DISK algorithm for node n

```
// G: list ofgrid cells overlaying the coverage hole
// P: DISK data message containing hole locations and interpolated
   values
// M: DISK interpolation message sent by node s_M comprising of
   list KS of Kriging systems where each tuple is of the form
   (g,L_g,V_g,B_g,T_g) defined as follows:
// g: centroid of a grid cell in G
// L_g: list of node locations
// V_g: list of node values
// B_g: list of correlations between the expected phenomenon value
   at g and each location in L_g
// T_g : list of linear maps
```

Input : DISK interpolation message: M
Output : Data message P, if node n is root of DISK tree; interpolation message M, otherwise

```
1  foreach DISK tree child c of node n do
2      Forward M to c
3  foreach grid cell g in G do
4      Compute Kriging window W for g
5      if location(n) is inside W then
6          Retrieve Kriging system from KS corresponding to g
7          k ← 1 + total entries in L_g
8          Compute A_k for g  //as described in Section 4.1
9          Compute composite linear map T_k for A_k
10         T_g ← T_k
11         Add location(n) to L_g
12         Add value(n) to V_g
13 if n is root of DISK tree then
14     Create an empty DISK data message P
15     foreach grid cell g in G do
16         Retrieve kriging system from KS corresponding to g
17         Compute vector λ of Kriging weights by applying linear maps T_g on vector B_g
18         Compute interpolation value v as: v ← λ · V_g
19         Save g's location and its computed interpolation v in P
20     Transmit P to the base station
21 else
22     Transmit M to parent node in DISK tree
```

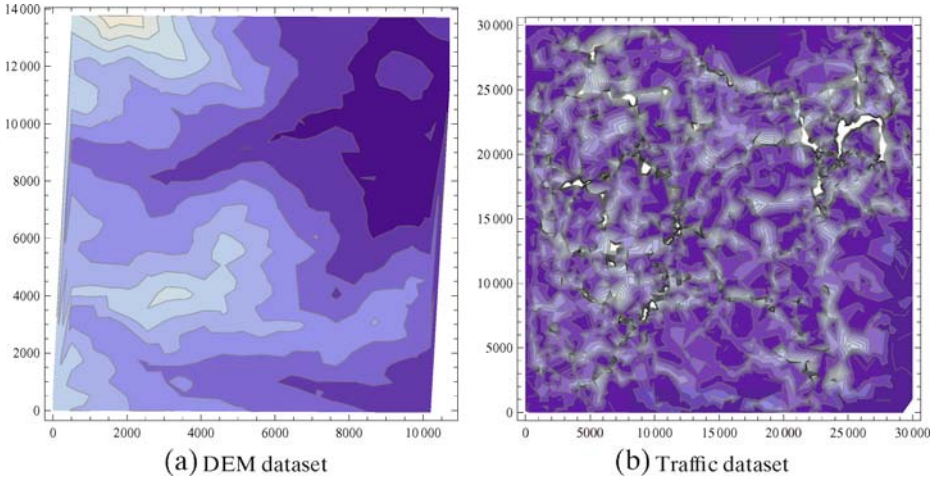


Fig. 8 Contour plots for DEM and Traffic datasets used in experimentation (**a**, **b**)

5.1 Global Kriging (GK) and averaging (AVG) algorithms

In order to compare the cost and accuracy of the DISK algorithm, we also simulate a Global Kriging (GK) algorithm that is a typical in-network data aggregation tree. GK assumes the full knowledge of node and coverage hole locations at the sink. Similar to DISK, GK also uses the moving window model for Kriging and hence interpolates using data only from a neighborhood. However, it performs its interpolation centrally by propagating the data from a neighborhood to the base station. Kriging computation of GK is restricted to the same neighborhood as DISK in order to ensure a fair cost comparison between the two methods.

A major distinction between interpolations by GK and DISK is the variogram model used in each method. In the DISK approach, we first create a variogram model locally, using the QS algorithm, and distribute the model to each node in the network. In GK, the variogram model is built centrally after collecting data from all nodes at the base station. The use of different models in each interpolation method leads to a difference in their accuracy. It is possible to use the global variogram model in DISK, in which case its interpolation results and accuracy will exactly match those of GK. However, in our experiments we use different models to analyze the impact of localized variogram modeling on interpolation accuracy.

In addition to GK, we also report the energy efficiency and accuracy results of an in-network aggregation based averaging scheme (AVG). This scheme simply computes the average value of all nodes present inside the Kriging neighborhood. We include the AVG method in our comparison to establish the design space for our proposed method. GK represents an ideal interpolation scheme based on global correlation data while AVG represents a quick and dirty estimation method. We show that our proposed scheme is much cheaper in cost as compared to GK and provides much higher accuracy than simple averaging.

5.2 Simulation setup

Our experiments are based on simulations of the Mica motes [7] implemented in the NS-2 simulation software [25]. We use the total data transmission in the network as a measure of energy usage and hence as the metric for cost comparison. The amount of data transmission in a network can be related to the energy expenditure by a function such as $\varepsilon = \sigma_s + \delta_s x$, where ε is the total amount of energy spent in sending a message with x bytes of content, and σ_s and δ_s represent the per-message and per-byte communication costs, respectively [32]. When aggregated for all nodes, the above energy relation represents the cumulative effect of number of messages and message size sent by each node.

In the experiments below, we do not incorporate the cost of variogram modeling in the cost of DISK and GK algorithms. We consider variogram modeling to be a setup cost incurred by both the algorithms. Like DISK, GK also requires a global variogram model for its computation. Since during interpolation it retrieves node values from Kriging neighborhood only, it cannot use the same values for building its variogram. Hence, GK has to use a variogram model created previously. Our experiments show that the cost of building the variogram in a localized manner (i.e., for DISK) is much smaller than that for building it in a global manner (i.e., for GK). Although DISK requires $O(n)$ messages to broadcast the variogram model in the network, the difference of the two costs still remains significant as shown in Fig. 5. Therefore, even if variogram modeling cost is considered, it will still be smaller for DISK than GK. Moreover, variogram modeling is performed only rarely as compared to interpolation. We discuss this feature of variogram modeling in more detail in Section 6.

We measure the accuracy based on cross-validation of the interpolation with the known values. The accuracy is computed as the root mean square error (RMSE). We measure the relationship between the deployment area and the Kriging neighborhood as a ratio between their sizes (in terms of number of nodes), referred to as the *network to neighborhood size ratio* (NNR). We repeat each experiment for four different coverage hole locations (one in each quadrant of the deployment area) and report the mean energy expenditure and RMSE.

5.3 Results and discussion

We have run multiple experiments studying different aspects of DISK in comparison to GK. Each of the following experiments isolate the effect of an individual factor on the performance of DISK and GK in terms of cost and accuracy. In the DEM dataset experiments we estimate the altitude at various points, whereas in the traffic dataset experiments we estimate the number of cars at various locations.

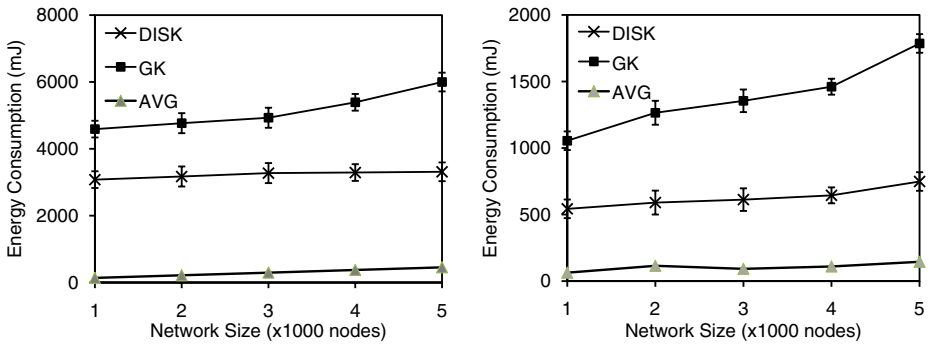
5.3.1 The effect of network size

In this experiment we analyze the scalability of DISK and GK with increasing network size. In each step, we increase the network size and the Kriging neighborhood such that the NNR stays constant at 2%. Since the node distribution for both DEM and traffic datasets is uniform, we gain the effect of increasing network size by

expanding the deployment area uniformly in all directions. For DEM data we start with a deployment area of 8 km^2 comprising approximately 1000 nodes deployed in a regular grid with 250 m resolution. We increase the deployment area up to 18 km^2 (approximately 5000 nodes) such that each increment adds approximately 1000 nodes. Similarly, for traffic data we start with a deployment area of 16 km^2 comprising approximately 1000 nodes deployed in a uniform random distribution. In this case, the deployment area is increased up to 36 km^2 (approximately 5000 nodes) with an addition of approximately 1000 nodes in each step. We assume that in all experiments the base station is located at the geographic center of the deployment area.

Figure 9a shows the effect of increasing network size on the energy expenditure in the DEM dataset experiments. It can be observed that DISK scales remarkably well with the increase in network size while GK increases in cost. For the network of 1000 nodes, DISK uses about 60% of energy used by GK while for a network of 5000 nodes its energy usage reduces to about 48% of GK. Figure 9b shows a similar scalability behavior for DISK and GK in the experiments with the traffic dataset. This behavior is expected because the energy consumption of DISK depends upon the number of nodes inside the Kriging neighborhood. On the other hand, the energy consumption of GK depends heavily on the network size as all node values from the neighborhood have to be propagated to the sink. As expected, AVG shows the least cost, several times smaller than the costs of both, DISK and GK.

Figure 10 shows a comparison of the accuracy of DISK, GK and AVG algorithms. AVG shows the worst accuracy, showing an error nearly twice as much as errors for DISK and GK. Both Kriging techniques interpolate using the same Kriging neighborhood in all experiments. The difference in accuracy can be attributed to the fitness of the variogram model used by a technique to the spatial correlation structure of corresponding dataset. DISK uses the localized variogram model built through the QS algorithm while GK creates the variogram model centrally. Resultantly, DISK interpolation suits datasets that exhibit high spatial correlation over short distances, such as the DEM dataset used in our experiments. For DEM dataset, we observe that the localized variogram model enables DISK to achieve similar (even slightly better) accuracy than GK. On the other hand, spatial correlation over small distances is low



(a) DEM dataset (b) Traffic dataset

Fig. 9 Effect of network size on energy expenditure (**a**, **b**)

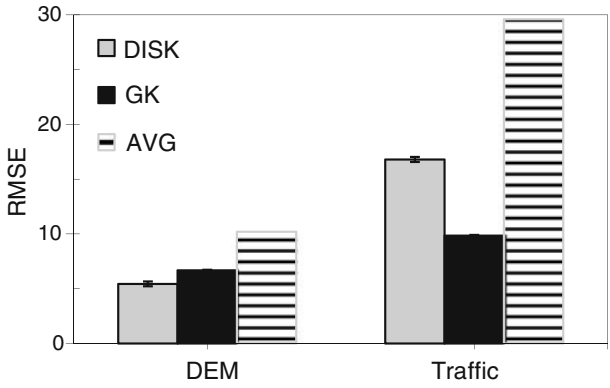


Fig. 10 Accuracy of DISK and GK

in traffic data resulting in an increased interpolation error for DISK. Although GK also suffers in this case, its accuracy degrades less than that of DISK.

For this experiment, Table 1 presents the mean estimation error for various confidence intervals, computed using Eq. 5. As discussed in Section 4, Kriging variance and confidence intervals can be computed for *free*, as a byproduct in the DISK algorithm. In practice, this calculation can be of huge value for the DISK algorithm as users can be kept informed regarding a system's confidence in the computed estimate. If the error interval is large, a user may opt for GK or may choose to refresh the variogram model.

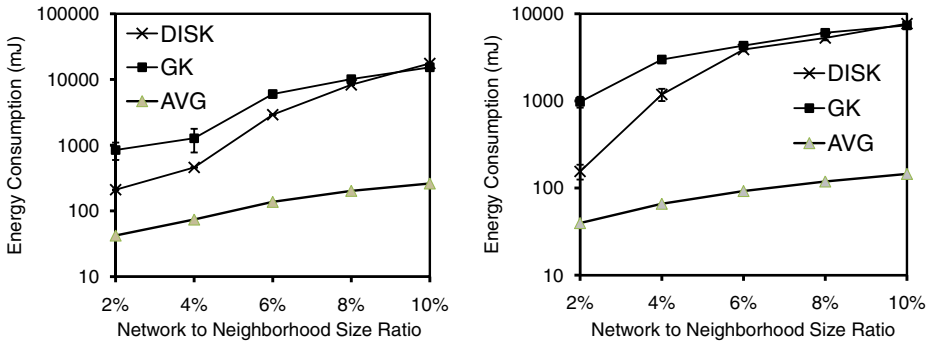
5.3.2 The effect of the Kriging neighborhood size

As noted earlier, DISK replaces the need of global knowledge with message passing between a neighborhood of nodes. This behavior results in a direct proportionality between the number of nodes in a neighborhood and the communication cost of DISK. Moreover, as sensor values from the neighborhood are used in estimation, the size of neighborhood also drives the estimation accuracy of the DISK approach. In this experiment, we analyze the impact of the size of Kriging neighborhood on the cost and accuracy of DISK. We increase the Kriging neighborhood size by increasing the NNR in equal steps while the deployment area is kept fixed (18 km^2 and 16 km^2 for DEM and traffic datasets, respectively).

The results in Fig. 11 show that increasing the neighborhood size results in a rapid increase in the communication cost of both GK and DISK. For GK, increasing neighborhood area results in more nodes in the neighborhood and hence more data to be propagated to the base station. Similarly, the communication cost of DISK rises with the increase in neighborhood size as more nodes are involved in the Kriging process. However, we observe from Fig. 12 that for accurate Kriging the NNR should not be increased beyond a certain value as further samples add noise to the

Table 1 Error statistics for DISK showing error in Kriging estimates in various confidence intervals

Confidence interval	75%	80%	85%	90%	95%	97.5%	99%	99.5%
DEM	0	0	1.8	1.8	2	3.618	3.9	5.12
Traffic	0	3.35	3.35	3.35	6.7	13.4	14.1	17.6



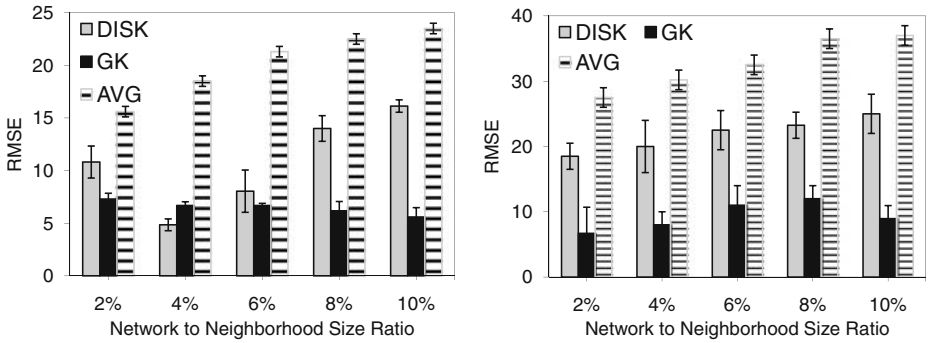
(a) DEM dataset (b) Traffic dataset

Fig. 11 Effect of the size of Kriging neighborhood on energy expenditure (energy is shown in the logarithmic scale) (**a**, **b**)

estimation and lower the accuracy. This result follows the well founded geostatistical assertion that as samples come from farther and farther away, the appropriateness of representing a phenomenon as a random function becomes doubtful [17]. As shown in Fig. 12a, the most accurate estimation results for DEM data are obtained for 4% NNR where the cost of DISK is only about 40% of GK. Due to lower spatial correlation in the traffic data, estimation error increases with increasing NNR and best result is achieved at 2% NNR (Fig. 12b). In both the cases, GK's accuracy is not effected much by the neighborhood size mainly due to the insensitivity of its variogram model towards spatial correlation over short distances.

5.3.3 The effect of distance of base station from the Kriging neighborhood

An important factor that determines the cost of GK and DISK is the distance of the base station to a Kriging neighborhood. The DISK algorithm only transmits the interpolation values from a Kriging neighborhood to the base station, whereas GK propagates all data to the base station. We expect that as the physical distance



(a) DEM dataset (b) Traffic dataset

Fig. 12 Effect of the size of Kriging neighborhood on accuracy (**a**, **b**)

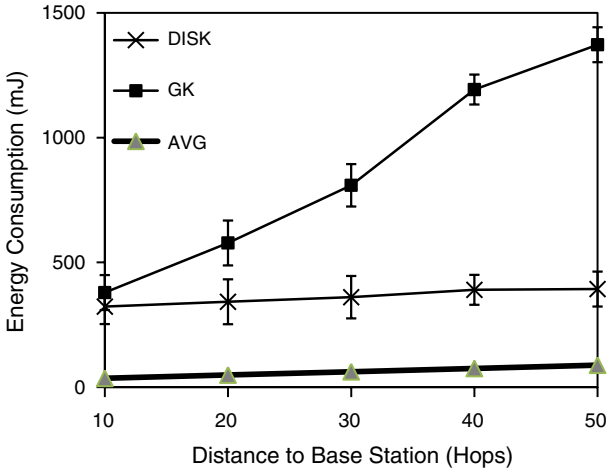


Fig. 13 Effect of distance of the base station from the Kriging neighborhood

increases between the base station and a Kriging neighborhood, the cost of GK will increase drastically. In this experiment, we analyze the scalability of both approaches for increasing distances between the base station and a Kriging neighborhood.

We perform this experiment on the DEM dataset where the network size and NNR are kept fixed at 5000 nodes and 5%, respectively. We assume that the base station is present at the geographical center of the deployment space and for each observation point we move the coverage hole away from the base station. Figure 13 shows the results of this experiment. As expected, the impact of increasing the sink distance is much larger on GK as the data transmission increases linearly with each hop. For instance, in this experiment the communication cost of GK increases by approximately 18 mJ per hop traversed outside the neighborhood. For DISK, on the other hand, this cost is approximately 1 mJ per hop as only interpolation results are routed from the neighborhood to the sink.

5.3.4 The effect of network density

Increase in network density results in more nodes per unit of the deployment area and hence increases the Kriging neighborhood size. As shown in Section 5.3.2, communication costs of both DISK and GK increases with increasing neighborhood sizes. We observe a similar behavior in this experiment, i.e., the communication costs of both approaches rises with increasing network density.

We perform this experiment on the DEM dataset and keep the deployment area fixed at 8 km^2. We start with a network size of approximately 1000 nodes and increase it to 2000 nodes while refining the resolution of the deployment grid. As shown in Fig. 14, the communication costs of both algorithms increase with increasing network density.

6 Further analysis

To simplify the presentation of our interpolation scheme, we do not directly address certain issues that can affect its accuracy and cost. It is important to note that for

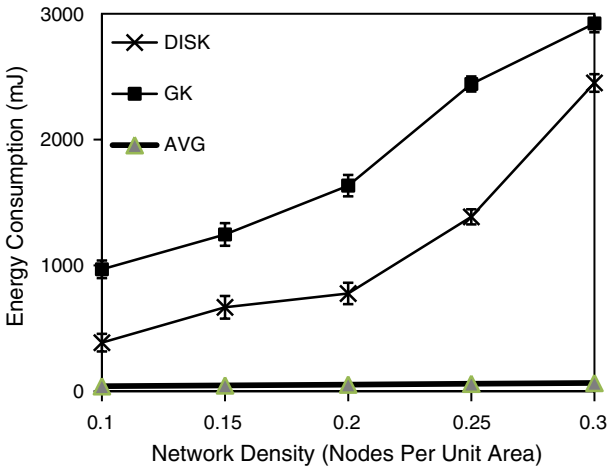


Fig. 14 Effect of the network density

a given variogram model and set of samples, the DISK algorithm computes exactly the same solution as any centralized Kriging solver would compute. Therefore, the accuracy of the DISK algorithm is largely dependent on the quality and accuracy of variogram modeling and hole discovery methods. In following subsections, we discuss major factors that influence the cost and accuracy of DISK algorithm.

6.1 Impact of variogram modeling

Theorem 1 establishes that the DISK algorithm will produce results identical to a centralized Kriging solver if both computations are performed using the same variogram model. However, as shown in Section 3, it is not possible to establish an exact variogram model in a distributed manner. Hence, to remain purely decentralized in our approach, we propose the use of QS algorithm for a distributed approximation of the variogram model prior to the use of DISK algorithm. This decision makes our approach more scalable.

Although variogram modeling, both in a distributed or centralized manner, incurs significant message cost, it is performed rarely. Therefore, we exclude the variogram building cost when computing the cost of interpolation in DISK or GK. The variogram model of a phenomenon characterizes the correlation among sensor nodes as the phenomenon evolves spatially. The intrinsic stationarity assumption states that this correlation depends only on the distance and the orientation of two nodes with respect to each other [6]. Therefore, node values may vary with time but the correlation among nodes is expected to stay uniform. For instance, an air-conditioning plant may cause the temperature at different points in a building to rise or fall but the spatial correlation in temperature values at these points should stay largely uniform. Therefore, once a variogram model is built there is no need of updating it each time Kriging interpolation has to be performed in a different part of the network. Situations with extremely dynamic correlation structure are rare. In such situations, all correlation based interpolation techniques are likely to perform poorly, a case not considered in our research.

6.2 Impact of coverage hole detection

We classify WSN coverage holes as *deployment* or *communication* holes. In applications such as NASA's VolcanoWeb [23], deployment holes exist naturally as it is not possible to deploy nodes in certain hostile areas. We argue that such deployment holes are known to WSN users in advance and hence active hole detection methods are not required to be used prior to interpolation. Communication holes, on the other hand, result when a significant number of nodes in close proximity face hardware or network failures. Only for such cases a hole detection method (such as [20, 38]) is required to be used to disseminate the knowledge of a coverage hold among the nodes in its neighborhood. In this context, typical errors such as overestimation or underestimation of hole area and missing smaller holes may arise that can affect interpolation accuracy. However, we consider accurate hole detection to be an orthogonal research direction to our work.

6.3 Impact of network gaps

Message loss during radio transmission is a common problem faced by WSNs. Typically, WSN communication protocols handle message loss by attempting retransmission of the lost message until a successful acknowledgment is received [11]. Retransmission is one of the most effective communication strategies catering to the intermittent message loss that occur routinely. We propose to employ a similar strategy in DISK implementation. In cases where message loss occurs due to failure of a large number of nodes, a communication hole detection strategy has to be employed. Similarly, the use of resilient routing schemes such as multi-path trees [24] could be explored.

7 Related work

The problem of phenomenon estimation in a WSN that partially covers a deployment area is explored in [41]. The basic idea of this approach is to first identify a subset of WSN nodes that provide enough samples for the estimation of a region of interest. The samples are then routed to the base station that estimates the phenomenon using a model based on partial differential equations. The major limitations of [41] in comparison to our proposed approach is its global nature. Their method assumes complete knowledge of the network at the base station and pumps all data from the sampled nodes to the base station. As shown in our experiments for increasing deployment sizes (as well as under other circumstances where coverage holes are distant) routing all sample values to the base station becomes increasingly inefficient.

The Distributed Regression method [12] for global kernel regression closely compares to DISK. This method is based on the notion that a number of local (or regional) correlation structures can be identified inside a given deployment. The authors suggest that message passing for a global regression task can be optimized by distributing the global computation among the constituent regions. Although we propose the idea of distributed interpolation on a regional basis as well, the concept of a region in our method is fundamentally different from [12]. For DISK, the interest in a region is not based upon its correlation pattern but its vicinity to a coverage hole.

Hence, the task of interpolation can be localized to the nodes forming a Kriging neighborhood around the coverage hole. Moreover, the iterative formulation of the Kriging method in DISK reduces the computation required by each node in the neighborhood. In [13] and [19] distributed regression concepts are further extended towards methods for optimal placement of sensor nodes in a WSN.

In [30], a WSN data aggregation approach based on spatial interpolation is proposed. It is argued that the aggregation quality can be improved if the aggregation operator takes into account the position of a sensor node and weighs its sensed value accordingly. For instance, the values reported by nodes located in the sparse parts of a deployment should be weighed higher than those located in the dense parts. Using a localized scheme for spatial interpolation and aggregation, the authors show considerable improvement in aggregation accuracy. However, a major limitation of the spatial interpolation methods proposed in [30] is the use of geometric distance rather than statistical distance between sampled and interpolated locations. This form of interpolation implicitly assumes uniform spatial continuity in the monitored phenomenon which is rarely the case in nature. In contrast, we propose spatial interpolation based on a model of spatial continuity that is naturally customized for a particular dataset, i.e., its variogram.

In [18] phenomenon interpolation using a distributed Gaussian kernel method, referred to as spatial window query over phenomena (SWOP), is proposed. SWOP first clusters the nodes based on their proximity. It then reduces the amount of data required to transmit to the base station by aggregating data inside each cluster using kernel estimation; a special spatial moving average method. We view SWOP as a data reduction method orthogonal to our approach. It is possible to use SWOP or other sample size reduction methods to optimize the Kriging process inside a neighborhood.

The use of data models to represent WSN data is proposed by [4, 9]. In [9] the authors present the BBQ framework that uses a statistical model along with live data acquisition such that queries can be responded in a probabilistic manner. In [4] the Ken architecture is proposed, which maintains a dynamic probabilistic model for the WSN data. In a WSN based on Ken, the base station answers all queries based on the data model. The nodes only send their data to the base station when it starts differing from the model's prediction. In [2] a similar approach is pursued where a Bayesian network model is used to answer target tracking queries in probabilistic terms.

A number of in-network aggregation schemes for WSNs have been proposed recently. Some in-network aggregation techniques, such as TAG (Tiny AGgregation) [21], are application independent and use a generic tree to compute all aggregate queries. In contrast, several recent approaches propose to tailor the data collection paths specifically to a group of similar queries [3, 26, 29]. In the WSN literature queries similar to the EVSJ query have been referred to as spatial join or proximity queries. Join queries have only recently received attention in the WSN research [5, 14, 39, 40].

In [40], the pruner-based acquisitional protocol (AQP) is proposed that relates closely to the QS algorithm proposed in our work. The AQP aims at efficient aggregation for queries that join nodes' records based on a distance threshold. Similar to the QS algorithm, the AQP also partially aggregates a node a's data as soon as it reaches an internal node that covers all nodes that are within a required distance to node a. However, unlike the QS algorithm, the AQP does not specify a particular

tree construction method and relies on passing extra information between the nodes. The QS algorithm, on the other hand, does not require any extra information as the tree is created in a predefined way. This tree construction method also decreases the number of hops that a node's data has to traverse in unaggregated form. It is also important to note that the aim of the QS algorithm is fundamentally different than all of the related works mentioned above, i.e., to support the task of correlation modeling, not to optimize query processing.

8 Conclusions and future work

Coverage holes are a reality for WSNs and it is often important to estimate the information within a coverage hole. Spatial interpolation particularly suits this problem as spatial correlations in measurements is common in WSNs. The challenge, however, is to perform the interpolation with high accuracy and minimal communication costs. We address this challenge by a novel method that uses a two step approach towards spatial interpolation: it first models the spatial correlation in the WSN data and then interpolates data for coverage holes based on this model. Our interpolation scheme, DISK, a distributed and localized Kriging computation is highly scalable compared to centralized interpolation. We show that the cost of local Kriging is determined by the size of the neighborhood where the Kriging operation is performed. We further show that for accurate results it is not only important to establish a better variogram model but also to restrict the neighborhood size within a certain threshold as increasing sizes lead to higher noise and lower accuracy.

A recent development in the WSN domain is the emergence of mobile WSNs including *controlled* [33] and *uncontrolled* [16] mobile WSNs. Static WSNs fulfill the sensing coverage requirement by massive and redundant deployment of nodes. On the other hand, mobile WSNs aim to replace the requirement of a large number of static nodes by a small set of constantly moving nodes providing *on demand* sensing coverage. We are currently extending our interpolation scheme for mobile WSNs. In such a setting the problem of coverage holes exists by design as only some parts of a network are covered at any given time. These systems can greatly benefit from spatial interpolation as it reduces node movement and communication costs. However, a direct application of our proposed interpolation scheme is not possible as mobile WSNs do not have fixed coverage holes and Kriging neighborhoods. Thus, methods are required that are localized as well as resilient to node movement.

References

1. Ahmed N, Kanhere SS, Jha S (2005) The holes problem in wireless sensor networks: a survey. Mob Comput Commun Rev 9(2):4–18
2. Biswas R, Thrun S, Guibas LJ (2004) A probabilistic approach to inference with limited information in sensor networks. In: Proceedings of IPSN, Berkeley, 26–27 April 2004, pp 269–276
3. Bonfils BJ, Bonnet P (2003) Adaptive and decentralized operator placement for in-network query processing. In: Proceedings of IPSN, Palo Alto, 22–23 April 2003, pp 47–62
4. Chu D, Deshpande A, Hellerstein J, Hong W (2006) Approximate data collection in sensor networks using probabilistic models. In: Proceedings of ICDE, Atlanta, 3–7 April 2006, p 48

5. Coman A, Nascimento MA (2007) A distributed algorithm for joins in sensor networks. In: Proceedings of SSDBM, Banff, 9–11 July 2007, p 27
6. Cressie NA (1993) Statistics for spatial data. Wiley, New York (1993)
7. Crossbow Technologies (2007) Crossbow Technologies homepage. http://www.xbow.com
8. Curran PJ, Atkinson PM (1998) Geostatistics and remote sensing. Prog Phys Geogr 22(1):61–78
9. Deshpande A, Guestrin C, Madden SR, Hellerstein JM, Hong, W (2004) Model-driven data acquisition in sensor networks. In: Proceedings of VLDB, Toronto, August 2004, pp 588–599
10. Gambino F, Kopp VC, Costa JFCL, Kopp JC, Fallon G, Davies N (2004) Incorporating uncertainty in coal seam depth determination via seismic reflection and Geostatistics. In: Proceedings of 7th international geostatistics congress, Banff, 26 September–1 October 2004, pp 537–542
11. Gnawali O, Yarvis M, Heidemann J, Govindan R (2004) Interaction of retransmission, blacklisting, and routing metrics for reliability in sensor network routing. In: Proceedings of IEEE SECON, Santa Clara, 4–7 October 2004, pp 34–43
12. Guestrin C, Bodik P, Thibaux R, Paskin M, Madden S (2004) Distributed regression: an efficient framework for modeling sensor network data. In: Proceedings of IPSN, Berkeley, 26–27 April 2004, pp 1–10
13. Guestrin C, Krause A, Singh AP (2005) Near-optimal sensor placements in Gaussian processes. In: Proceedings of ICML, Bonn, 7–11 August 2005, pp 265–272
14. Gupta H, Chowdhary V (2007) Communication-efficient implementation of join in sensor networks. Ad Hoc Netw 5(6):929–942
15. Huang CF, Tseng YC (2005) The coverage problem in a wireless sensor network. Mob Netw Appl 10(4):519–528
16. Hull B, Bychkovsky V, Zhang Y, Chen K, Goraczko M, Miu A, Shih E, Balakrishnan H, Madden S (2006) Cartel: a distributed mobile sensor computing system. In: Proceedings of SenSys, Boulder, November 2006, pp 125–138
17. Isaaks E, Srivatava RM (1989) An introduction to applied geostatistics. Oxford, New York
18. Jin G, Nittel S (2008) Towards spatial window queries over continuous phenomena in sensor networks. IEEE Trans Parallel Distrib Syst 19(4):559–571
19. Krause A, Guestrin C, Gupta A, Kleinberg J (2006) Near-optimal sensor placements: maximizing information while minimizing communication cost. In: Proceedings of IPSN, Nashville, 19–21 April 2006, pp 2–10
20. Kröller A, Fekete SP, Pfisterer D, Fischer S (2006) Deterministic boundary recognition and topology extraction for large sensor networks. In: Proceedings of SODA, pp 1000–1009. ACM, New York (2006)
21. Madden SR, Franklin MJ, Hellerstein JM, Hong W (2005) TinyDB: an acquisitional query processing system for sensor networks. ACM Trans Database Syst 30(1):122–173
22. Morrison JL (1974) Observed statistical trends in various interpolation algorithms useful for first stage interpolation. Can Cartogr 11(2):142–159
23. NASA Volcano Sensorweb (2007) Volcano sensorweb homepage. http://sensorwebs.jpl.nasa.gov/
24. Nath S, Gibbons PB, Seshan S, Anderson ZR (2004) Synopsis diffusion for robust aggregation in sensor networks. In: Proceedings of SenSys, Baltimore, 3–5 November 2004, pp 250–262
25. Network Simulator (2006) Network simulator homepage. http://www.isi.edu/nsnam/ns/doc
26. Pattem S, Krishnamachari B, Govindan R (2004) The impact of spatial correlation on routing with compression in wireless sensor networks. In: Proceedings of IPSN, Berkeley, 26–27 April 2004, pp 28–35
27. Pottie GJ, Kaiser WJ (2000) Wireless integrated network sensors. Commun ACM 43(5):51–58
28. Secure CITI (2006) Exploratory research on sensor-based infrastructure for early Tsunami detection. http://www.cs.pitt.edu/s-citi/tsunami/
29. Sharaf MA, Beaver J, Labrinidis A, Chrysanthis PK (2004) Balancing energy efficiency and quality of aggregate data in sensor networks. VLDB J 13(4):384–403
30. Sharifzadeh M, Shahabi C (2006) Utilizing Voronoi cells of location data streams for accurate computation of aggregate functions in sensor networks. GeoInformatica 10(1):9–36
31. Shrivastava N, Buragohain C, Agrawal D, Suri S (2004) Medians and beyond: new aggregation techniques for sensor networks. In: Proceedings of SenSys, Baltimore, 3–5 November 2004, pp 239–249
32. Silberstein A, Braynard R, Yang J (2006) Constraint chaining: on energy-efficient continuous monitoring in sensor networks. In: Proceedings of SIGMOD, Chicago, 26–29 June 2006, pp 157–168

33. Somasundara AA, Jea DD, Estrin D, Srivastava MB (2006) Controllably mobile infrastructure for low energy embedded networks. IEEE Trans Mob Comput 5(8):958–973
34. Strangeways I (2003) Measuring the natural environment. Cambridge University Press, New York
35. Tolle G, Polastre J, Szewczyk R, Culler D, Turner N, Tu K, Burgess S, Dawson T, Buonadonna P, Gay D, Hong W, Hong W (2005) A macroscope in the Redwoods. In: Proceedings of SenSys, San Diego, 2–4 November 2005, pp 51–63
36. Umer M, Kulik L, Tanin E (2008) Kriging for localized spatial interpolation in sensor networks. In: Proceedings of SSDBM, Hong Kong, 9–11 July 2008, pp 525–532
37. USGS Digital Elevation Models (2007) USGS digital elevation models homepage. http://data.geocomm.com/dem/
38. Wang Y, Gao J, Mitchell JS (2006) Boundary recognition in sensor networks by topological methods. In: Proceedings of MobiCom, Los Angeles, 23–29 September 2006, pp 122–133
39. Yang X, Lim HB, Özsu TM, Tan KL (2007) In-network execution of monitoring queries in sensor networks. In: Proceedings of SIGMOD, Beijing, 12–14 June 2007, pp 521–532
40. Yiu ML, Mamoulis N, Bakiras S (2009) Retrieval of spatial join pattern instances from sensor networks. GeoInformatica 13(1):57–84
41. Zhang H, Moura JMF, Krogh B (2005) Estimation in sensor networks: a graph approach. In: Proceedings of IPSN. IEEE, Piscataway, p. 27
42. Zimmerman D, Pavlik C, Ruggles A, Armstrong M (199) An experimental comparison of ordinary and universal Kriging and inverse distance weighting. Math Geol 31:375–390

Muhammad Umer is a PhD candidate at the Department of Computer Science and Software Engineering (CSSE) at the University of Melbourne. His research is mainly focused on developing localized and adaptive algorithms for query processing in sensor networks using methods from spatial statistics domain. He holds industrial experience in mobile application development ranging from asset management and tracking to distributed enterprize systems. His work at Melbourne University is partly funded by Victoria Research Lab of the National ICT Australia (NICTA) where he is associated with the NICTA Open SensorWeb Architecture (NOSA) project. He is also associated with Sensing, Ubiquity, Mobility (SUM) research lab at the CSSE department. He has published several papers in refereed conferences and regularly serves as external reviewer for international conferences including ACM GIS, Database Systems for Advanced Applications (DASFAA) and Australasian Database Conference (ADC).

Dr. Lars Kulik's overall research goal is to develop a theory of spatial information for building pervasive computing systems that anticipate, adapt and respond to the needs of users, and provide services based on the user's location and context. Specifically, his research focuses on efficient algorithms for moving objects, information dissemination algorithms in sensor networks, and spatial algorithms in pervasive computing environments. He also researches negotiation-based models for location privacy, and robust algorithms that cope with imperfection, especially in the context of mobile and location-aware computing. Dr Kulik is a lecturer at the Department of Computer Science and Software Engineering at the University of Melbourne.

Dr. Egemen Tanin's research areas include spatial databases and distributed data management. He has finished his PhD at the University of Maryland at College Park on accessing and browsing large databases over the Internet in 2001. His work to access large data such as satellite images was later used by the Global Change Master Directory of NASA. During this time, he has also worked as a software engineer developing one of the first distributed object infrastructures, Cybele. Afterwards, he has focused on spatial data and accessing large spatial databases over the Internet. He has developed the APPOINT (Approach for Peer-to-Peer Offloading the INTernet) system to help users of the Internet efficiently access large spatial data in a distributed fashion. He has joined the University of Melbourne in 2003, where he was awarded two Early Career Researcher Grants for his work on developing indices and algorithms for accessing spatial data over distributed environments in a decentralized manner. He developed the first P2P spatial index in collaboration with researchers at the University of Melbourne and at the University of Maryland. Some of this work is now patented and being commercialized by his students. He was then invited to spend the second half of 2007 in NEC Labs, Cupertino, CA, USA, as a visiting researcher on distributed spatial data management. He is currently on the program and organizing committees of multiple international conferences and workshops and has presented various invited talks in the area of spatial and distributed data management.