Cross-layer interactions in energy efficient information collection in wireless sensor networks with adaptive compressive sensing

Chun Tung Chou[1] Rajib Rana[2] Wen Hu[1]

[1] School of Computer Science and Engineering, University of New South Wales, Sydney, Australia
ctchou@cse.unsw.edu.au; rajibr@cse.unsw.edu.au
[2] Autonomous Systems Laboratory, CSIRO ICT Centre, Brisbane, Australia
Wen.Hu@csiro.au

Technical Report
UNSW-CSE-TR-0909
May 2009

THE UNIVERSITY OF NEW SOUTH WALES

School of Computer Science and Engineering
The University of New South Wales
Sydney 2052, Australia

Abstract

We consider the problem of using Wireless Sensor Networks (WSNs) to measure the temporal-spatial field of some scalar physical quantities. Our goal is to obtain a sufficiently accurate approximation of the temporal-spatial field with as little energy as possible. We propose an adaptive algorithm, based on the recently developed theory of adaptive compressive sensing, to collect information from WSNs in an energy efficient manner. The key idea of the algorithm is to perform "projections" iteratively to maximise the amount of information gain per energy expenditure. We prove that this maximisation problem is NP-hard and propose a number of heuristics to solve this problem. This maximisation problem can also be viewed as a routing problem with a metric which is a non-linear function of the data field; therefore the problem is cross-layer in nature, requiring information in both application and network layers. We evaluate the performance of our proposed algorithms using data from both simulation and an outdoor WSN testbed. The results show that our proposed algorithms are able to give a more accurate approximation of the temporal-spatial field for a given energy expenditure.

1 Introduction

We consider the problem of using Wireless Sensor Networks (WSNs) to measure the temporal-spatial field of some scalar physical quantities, e.g. the variation of temperature over a certain geographical area over a certain time. One method of measuring such a temporal-spatial field (or data field for short) is to have all the sensors in the WSNs return their measurements to a sink (or data fusion centre) at regular time intervals. This method gives the maximum amount of information on the data field but at the same time requires the maximum amount of energy to collect the information. Instead, it may be possible to use a reduced amount of energy to obtain a sufficiently accurate approximation of the data field. This is possible if the sensor measurements are correlated, which happens when the sensors are densely deployed. This paper proposes an adaptive algorithm to obtain a sufficiently accurate approximation of the data field with as little energy as possible. Although there is a rich literature on adaptive sampling for WSNs, e.g. [7, 16], to the best of our knowledge, our algorithm is the first one that uses *adaptive compressive sensing*.

Compressive sensing [5, 3, 8] is a collection of recently proposed sampling methods in Information Theory. The promise of compressive sensing is that it can obtain a sufficiently accurate approximation of an unknown data field by using a small number of generalised measurements, which are known as *projections* in the compressive sensing literature. (The concept of projections will be explained in Section 2.) In adaptive compressive sensing [11, 19], these projections are iteratively computed in order to extract the maximum amount of information from the unknown data field with as few projections as possible.

Although the existing adaptive compressive sensing algorithms can obtain good approximation of the data field with a smaller number of projections than their non-adaptive counterparts, these algorithms cannot be directly applied to WSNs because they do not take into consideration the energy required to acquire a projection in WSNs. For data collection in WSNs, the goal is to collect as much information with as little energy (rather than as few projections) as possible. In this paper, we propose a method to iteratively compute projections that maximises the ratio of information gain to the energy required to acquire the information in order to realise energy-efficient information collection in WSNs (Sections 2 and 3). We show that this maximisation problem is NP-hard and propose a number of heuristics to solve this problem (Section 3). We evaluate the performance of our proposed algorithms using simulation data and data collected from an outdoor WSN (Section 4). Our evaluation shows that our proposed algorithm gives a better accuracy for a given energy expenditure. An interesting aspect of our work is that we show that the computation of a good projection requires the consideration both routing and information gain at the same time, which gives rise to cross-layer interaction between the application and network layers. In fact, the computation of a good projection vector can be considered as a routing problem with a non-additive and non-linear metric.

2 Information collection framework

We model the WSN as a graph $G = (\{s\} \cup V, E)$ where s is the sink node, $V = \{1, 2, ..., n\}$ is the set of sensor nodes and E is the set of edges where an edge exists between two sensor nodes if they are within the communication range of each other. (Note that the framework described here can equally be applied to a cluster with n sensor nodes and a cluster head, therefore the method is scalable.) We assume that the sensors are synchronised. We consider a snapshot of the temporal-spatial field where at a particular time t, the sensors make a measurement. Let the noise-free sensor reading of sensor node i (where $i = 1, ..., n$) be x_i. The actual (noisy or measured) sensor reading is assumed to be corrupted by an independent and identically distributed zero mean Gaussian noise of variance σ^2. Let the sensor noise at sensor i be e_i, then the actual sensor reading at sensor node i is $y_i = x_i + e_i$. We will use x to denote the vector $[x_1, x_2, ..., x_n]^T$ where T denotes matrix transpose; the vectors e and y are similarly defined. Our goal is to obtain an approximation $\hat{x} = [\hat{x}_1, \hat{x}_2, ..., \hat{x}_n]^T$ of the true data field x. We will measure the accuracy of the approximate data field by using the relative error $\frac{\|x-\hat{x}\|}{\|x\|}$ where $\|x\| = \sqrt{\sum_{i=1}^{n} x_i^2}$ denotes the $2 - norm$ of x.

We assume that both sensing energy and energy required for computation is negligible, which are fairly typical assumptions in WSNs [10]. The energy consumption in the WSNs is dominated by radio communications in transmitting and receiving data packets. In this paper, we will measure the energy consumption by the total number of transmissions required to collect the information on data field. The reference scenario is when all nodes in the network send the data to the sink which requires a number of transmissions of the order of n^2 in the multi-hop scenario. This work distinguishes itself from other works in energy efficient adaptive sensing, e.g. [12, 7, 16], in that it uses recent works in adaptive compressive sensing [11, 19]. However, existing work in adaptive compressive sensing only takes into consideration the accuracy of the approximate data field. In order to apply adaptive compressive sensing to WSNs, our work in this paper takes both accuracy and energy into consideration. In particular, we will show that there is a cross-layer interaction in using adaptive compressive sensing in WSNs where one needs to take both accuracy (at the application layer) and routing into consideration. To be best of our knowledge, such cross-layer interactions have not been studied before. In order to understand this cross-layer interaction, we will need to understand what a *projection* is in compressive sensing.

2.1 Projections in compressive sensing

A distinctive feature of compressive sensing is that it uses *projections* to collect information. For a snapshot of the noisy data field $\{y_i\}$, the projection of the vector y on a projection vector $p = [p_1, p_2, ..., p_n]^T$ is defined by the inner product $p^T y = \sum_{i=1}^{n} p_i y_i$. Let us illustrate the concept of projection vectors and how projections can be calculated in a WSN with a few examples. Consider the network shown in Figure 2.1 with 4 sensor nodes $\{1, 2, 3, 4\}$ and sink node s.

Example 1 *If the projection vector p is $[0.2, 0.3, 0.4, 0.1]$, then the projected value $p^T y = 0.2y_1 + 0.3y_2 + 0.4y_3 + 0.1y_4$. The sink can obtain this projected value without the sensors*

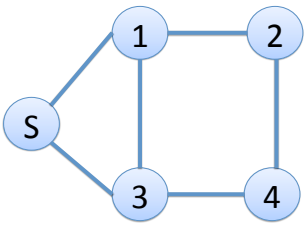

Figure 2.1: Example WSN.

sending their sensor readings to the sink. This can be achieved by the sink passing a message along the tour $S-1-2-3-4-S$ *using source routing in the WSN. The message contains the entire projection vector* p *as well as a field in the message to store the intermediate result of the projection calculation. As the message travels through the tour, each sensor computes its contribution to the projected value and adds it to the intermediate result. After that, the sensor writes the new intermediate result to the message and forwards the message to the next hop. For example, sensor node 2 will receive from sensor node 1 a message with* $0.2y_1$ *as the intermediate result; sensor 2 will compute* $0.3y_2$ *and add this to* $0.2y_1$*, then it will write the sum to the message and then pass it on to the next hop. Note that the computation of this projection requires 5 wireless transmissions.* □

Example 2 *If the projection vector* p *is* $[0.1, 0.2, 0, 0]^T$*, then the projected value* $p^T y = 0.1y_1 + 0.2y_2$*. This can be computed by passing a message along the tour* $S-1-2-1-S$ *since sensor readings from sensors 3 and 4 are not needed to compute this projection. Therefore, the calculation of this projection requires only 4 wireless transmissions. Note that in general, the calculation of a projection only requires a message to be passed along those sensor nodes with a non-zero projection vector coefficient. From an energy efficiency point of view, one would therefore aim to find the shortest tour (or aggregation tree) which passes through all sensor node required.* □

Example 3 *If the projection vector* p *is* $[0, 0, 1, 0]^T$*, then the projected value* $p^T y = y_3$*. Therefore, this projection vector corresponds to collecting the noisy sensor reading from sensor 3. This example aims to show that a projection is a general method of collecting data and collecting a sensor reading from a sensor is in fact a special case of performing a projection.* □

2.2 Adaptive compressive sensing for WSNs

With the above example, we have explained what a projection is and how a projection can be computed in WSN. Assuming that the sink has the projection vectors ϕ_j $(j = 1, ..., k)$ and their corresponding projected values $z_j = \phi_j^T y$ $(j = 1, ..., k)$, then the sink can use compressive sensing to estimate the noise free data field x. Most of the compressive sensing

algorithms proposed to date are non-adapative which means the projection vectors are not chosen according to information collected so far. Recent effort in adaptive compressive sensing [11, 19] shows that by choosing the coefficients in the projection vector to maximise the information content, it is possible to collect more information (or achieve a smaller relative error in estimating x) using a smaller number of projections. In this paper, we will use the generic adaptive compressive sensing algorithm outlined under Algorithm 0.

Algorithm 0 Generic adaptive compressive sensing

1: Each sensor randomly decides whether to send its reading to the sink and if so, it sends its reading to the sink. Note that this can be realised by a node having a probability to send data to the sink and this probability can be determined by the sink and change over time.
2: **while** The sink is not satisfied with the accuracy of the estimated data field **do**
3: The sink determines a projection vector and the corresponding tour to use.
4: The sink sends a message along the tour and waits for the projected value to return.
5: The sink updates the estimate of the unknown data field and determines its accuracy.
6: **end while**

The general idea of the generic adaptive compressive sensing algorithm is as follows. In line 1, a small number of sensors randomly decide to return their sensor readings to the sink. These random samples will allow the sink to estimate the data field and decide whether it needs to use the iteration in lines 2–6 to collect further information from the sensor field. This iteration continues until the sink is satisfied with the accuracy of the estimated data field. Note that the rationale behind the above adaptive compressive sensing is the same as that of optimal experiment design in statistics [6] and active learning in machine learning [14].

A key step in the above algorithm is line 3 where a good projection vector has to be determined. This step is also studied in the adaptive compressive sensing algorithms in [11, 19] where a projection vector which maximises the amount of information gain is determined. However, these earlier works are not suitable for WSNs because they do not take into consideration the energy required to acquire a projection. We show in our earlier examples that the energy required to acquire a projection depends on the length of the tour (or the size of the aggregation tree) needed to obtain the projection, which is in turn a function of the locations of the non-zero coefficients in the projection vector. Therefore, the choice of the coefficient of a projection vector can affect both the information content and the energy expenses (via the choice of route to obtain the projection). This means that a projection vector should be chosen to collect as much information on the unknown data field with as little energy as possible, which is in fact a cross-layer design problem because of the interaction of the application and routing layers. In Section 3, we propose an optimisation problem to find a projection vector which maximises the information gain per energy expenditure and we prove that this optimisation problem is NP-hard. Because of NP-hardness of the problem, we will propose a number of heuristics in Section 3 to solve this problem.

We will measure the energy consumption in WSNs by counting the number of packet transmissions, e.g. a node which is k hops away from the sink will require k packet transmis-

sions to reach the sink. In line 1 of Algorithm 0, energy is consumed to send packets to the sink, while in lines 2–6, energy is consumed to send packets along the tour. It can be seen that if a good probability can be found in line 1, this can minimise the energy consumption in lines 2–6. A possibility is to determine this probability from past history of the data. In this paper, we assume a reasonable estimate of this probability can be obtained and focus on the design of a projection vector which balances information gain and energy consumption. We also assume that the sink (or cluster head) knows the topology of the network (or cluster).

Remark 1 *We would also like to remark that it is possible to modify line 1 to have a random number of sensors initiating a number of random projections towards the sink, this will improve the information content at the sink and will be studied in the future. See Section 3.4 for further discussion on this.*

3 Adaptive projection vectors

In this section, we show how a projection vector can be computed to provide a high information gain on the unknown data field with small energy expenditure, i.e. line 3 of Algorithm 0. In Section 3.1, we derive an expression on the expected information gain for a projection vector. We then formulate an optimisation problem in Section 3.2 which balances information gain against energy expenditure and show that the problem is NP-hard. In Section 3.3, we will present a number of heuristics to solve the proposed optimisation problem.

3.1 Expected information gain of a projection vector

In this section, we derive an expression for the expected information gain of a projection vector. We assume that k projections have already been measured over the data field. Let $\phi_1, \phi_2, ..., \phi_k \in \mathbb{R}^n$ denote these k projection vectors and $z_1, ..., z_k$ denote the corresponding projected values, i.e. $z_i = \phi_i^T y$. The advantage of compressive sensing is that it can estimate x even if k is less than n (= number of sensors = number of data points in a snapshot) provided that we know that the noise-free data field vector x is *compressible* [3, 1].

A vector x is said to be compressible in a *basis* [18] $B \in \mathbb{R}^{n \times n}$ (where the basis vectors are the columns of B) if the coefficients $w = B^{-1}x$ of x in the basis B has the following property: the ℓ-th largest (in absolute value) coefficient of w decays faster than $\ell^{-\frac{1}{\beta}}$ for some $\beta \in (0, 1)$ [1]. Intuitively, a signal is compressible if the signal contains redundancy. Therefore, for a sensor data field whose data is correlated, it is reasonable to assume that the signal is compressible. In this paper, we will make the assumption that we know a priori the basis in which the signal is compressible. This is by no means a restriction as preliminary experiments can be carried out to determine this basis.

Assuming that the noise-free data field x is compressible in the basis B and $x = Bw$ where w are the coefficients of x in the basis B. We can write the projections using the following

data equation:

$$\underbrace{\begin{bmatrix} z_1 \\ \vdots \\ z_k \end{bmatrix}}_{z} = \underbrace{\begin{bmatrix} \phi_1^T \\ \vdots \\ \phi_k^T \end{bmatrix}}_{\Phi} y = \Phi(x+e) = \Phi B w + \Phi e \tag{3.1}$$

Note that most compressive sensing algorithms assume that the noise on the projected values are independent. However, since a noisy sensor reading may appear in multiple projections in Eq. (3.1), the noise on the projected values are correlated. In order to de-correlate the noise in the projected values, we compute the Cholesky factorisation [9] of $\Phi\Phi^T = R^T R$ (Note: We assume that Φ has full row-rank which is equivalent to assuming that the projection vectors $\{\phi_1, \phi_2, ..., \phi_k\}$ are linearly independent. This is a reasonable assumption because linearly dependent projection vectors do not yield new information.) and pre-multiply Eq. (3.1) with R^{-T} to obtain the revised data equation:

$$\underbrace{R^{-T} z}_{\tilde{z}} \;=\; \underbrace{R^{-T}\Phi B}_{\tilde{\Phi}} w + R^{-T}\Phi e \tag{3.2}$$

We can now input $\tilde{\Phi}$ and $\tilde{z}$ to the Bayesian compressive sensing (BCS) algorithm [11] which returns the posteriori probability distribution of the estimate of unknown vector w and unknown noise variance σ^2. Let us denote the estimate of w by $\hat{w}$. The probability distribution of $\hat{w}$ is Gaussian with mean $\mu \in \mathbb{R}^n$ and covariance matrix $\Sigma \in \mathbb{R}^{n\times n}$. (Note that μ and Σ are the outputs of the BCS algorithm and are therefore functions of $\tilde{\Phi}$ and $\tilde{y}$.) We can now obtain $\hat{x} = B\hat{w}$. Therefore, the estimate of the data field $\hat{x}$ is Gaussian distributed with mean $B\mu$ and covariance $B\Sigma B^T$. Lastly, we will use $\hat{\sigma}^2$ to denote the posteriori estimate of the measurement noise variance σ^2.

Since our goal is to obtain a sufficiently accurate estimate of the unknown data field, this is equivalent to minimising the uncertainty in the estimate $\hat{x}$. Since $\hat{x}$ is given by a continuous probability distribution, we can measure the uncertainty in $\hat{x}$ by using the differential entropy of $\hat{x}$ where a small differential entropy implies a small uncertainty. Thus, we can measure the additional information provided by a new projection vector p by the reduction in differential entropy in the estimate $\hat{x}$ caused by using p together with $\{\phi_i\}_{i=1,...,k}$. The use of the new data equation (3.2) means that the results in [11] on the entropy reduction by an additional projection vector is no longer applicable. We have the following new result.

Lemma 1 *The reduction of differential entropy ΔH in the estimate of the unknown data field by using an additional projection vector $p \in \mathcal{B} = \{p \in \mathbb{R}^{n\times 1} : \|p\| = 1\}$ with $V^T p \neq 0$ is given by*

$$\Delta H(p) = \tfrac{1}{2}\log(1 + \frac{1}{\hat{\sigma}^2}\frac{p^T V V^T B\Sigma B^T V V^T p}{p^T V V^T p}) \tag{3.3}$$

where V satisfy $\Phi V = 0$ and $V^T V = I$ where I denotes the identity matrix. (Note that V is in fact an orthonormal basis of the null space of Φ.)

Proof: See Appendix A.

Remark 2 *Note that there is no loss in generality in considering a projection vector p with $\|p\| = 1$ (i.e. in the ball $\mathcal{B}$), since any non-zero projection vector can be re-scaled to have unit-norm and this does not affect the compressive sensing estimation.*

Remark 3 *If $V^T p = 0$, then it implies that the projection vector p is in the range space of previously chosen projection vectors $\{\phi_1, ..., \phi_k\}$. Note that such projection vectors will not provide new information at all. Therefore, a new projection vector p has to be chosen such that $V^T p \neq 0$.*

Let Θ_1 and Θ_2 denote, respectively, the matrices $VV^T B\Sigma B^T VV^T$ and VV^T in Lemma 1. In the special case that we use the projection operation to collect a measurement from a single sensor, i.e. the projection vector p contains one 1 and all other elements are zeros, then the result in Lemma 1 says that we should collect from the sensor i if the $\frac{\Theta_{1,(i,i)}}{\Theta_{2,(i,i)}}$ is the biggest, where $\Theta_{1,(i,i)}$ and $\Theta_{2,(i,i)}$ are respectively the i-th diagonal elements of Θ_1 and Θ_2. However, the collection of a measurement from a single sensor in general does not give the maximum information gain or reduction in entropy. By using standard results in matrix theory, the projection vector p that maximises entropy reduction is given by the generalised eigenvector of the matrix pencil (Θ_1, Θ_2) that corresponds to the largest generalised eignevalue [15]. Although this optimal projection vector can be readily computed, this optimal projection vector generally contains many non-zero elements which means that the energy cost to acquire this optimal projection can be high. Since a WSN has limited energy reserve, it is therefore important to consider both information gain and energy consumption simultaneously.

3.2 Balancing information gain and energy consumption

In order to strike a balance between information gain and energy consumption, we propose that in each iteration we choose the projection vector $p \in \mathcal{B}$ to maximise the ratio

$$Q(p) \quad = \quad \frac{\Delta H(p)}{E(p)} \tag{3.4}$$

where $E(p)$ is the minimum energy, measured in terms of the number of wireless transmissions, needed to acquire the projection $p^T y$ from the WSN. (Note: Adaptive compressive sensing algorithms in [11, 19] seek to find a p to maximise the information gain $\Delta H(p)$.) In other words, we choose the projection vector that gives the maximum information gain per energy cost needed to acquire it. Although the projection vector that maximises $\Delta H(p)$ can readily be computed, the problem of maximising $Q(p)$ is in general NP-hard.

Theorem 1 *Assuming that we measure $E(p)$ by the number of packet transmissions needed to compute the projection p in a WSN, then the optimisation problem $\max_{p\in\mathcal{B}} \frac{\Delta H(p)}{E(p)}$ is NP-hard.*

Proof: We will show that the problem is NP-hard for a particular instance of the problem. Consider the case where the WSN is fully connected, then for a given projection vector p, the minimum energy $E(p)$ needed to acquire this projection is to use the shortest tour to connect

all the sensors whose corresponding projection vector coefficients are non-zero. The energy cost needed is equal to the length of the tour and since the network is fully connected, it is in turn equal to the number of non-zero elements in the projection vector p. Let $card(p)$ denote the number of non-zero elements in p. The optimisation problem can equivalently be written as $\max_{k=1,..,n} \frac{1}{k} \max_{p\in\mathcal{B};card(p)=k} \Delta H(p)$. Since log is a strictly increasing function, maximising $\Delta H(p)$ is equivalent to maximising $\frac{p^T\Theta_1 p}{p^T\Theta_2 p}$. The optimisation problem $\max_{p\in\mathcal{B};card(p)=k} \frac{p^T\Theta_1 p}{p^T\Theta_2 p}$ is an instance of the sparse Linear Discriminant Analysis which is known to be NP-hard [15]. Hence the theorem. □

Since the problem of choosing a projection vector p to maximise $Q(p)$ is NP-hard, we propose a number of heuristics to solve this problem.

3.3 Heuristics

In this section, we present a number of heuristics, of increasing complexity, to solve the optimisation problem $\max_{p\in\mathcal{B}} \frac{\Delta H(p)}{E(p)}$. In order to simplify the design of the heuristics, we will approximate $\Delta H(p)$ by $\frac{1}{2}\log(1+\frac{1}{\delta^2}p^T\Theta_1 p)$. Note that this is not expected to affect the result much since Θ_2 is an idempotent matrix [9]. Note that the problem remains NP-hard even with this approximation.

Algorithm 1 (Sensor with best information gain per energy): In this algorithm, the sink will use a projection to obtain a sensor measurement from a sensor. In other words, the projection vector p is limited to have only one non-zero element. The optimisation problem can be written as $\max_{i=1,...,n} \max_{p_i=1;p\in\mathcal{B}} \frac{\Delta H(p)}{E(p)}$ which can be solved by computing the value of $\frac{\Delta H(p)}{E(p)}$ for each sensor, where for the i-th sensor, $\Delta H(p) = \frac{1}{2}\log(1+\Theta_{1,(i,i)})$ where $\Theta_{1,(i,i)}$ is the (i,i)-th element in the matrix Θ_1 and $E(p)$ is the number of hops to sensor i via the least hop path times 2.

Algorithm 1 can be implemented by the sink sending a query packet to the chosen sensor which will then return its measurement to the sink. However, this algorithm only uses the sensor measurement of the end-point of the path but it does not use any information from the sensors along the path. It can readily be shown that a projection vector that uses all sensors along a path will always give a higher entropy reduction than a projection vector that uses only a subset of sensors along the same path; moreover, the energy consumption of these two situations are equal, so it will be an advantage to use all the sensors along the path too. Based on this discussion, we propose the following algorithm.

Algorithm 2 (Shortest path with best information gain per energy): Consider a sensor node $i \in V$, let $\mathcal{P}_i$ be the set of sensor nodes along the least hop path from the sink to sensor i excluding the sink. We abuse the notation and write $p \in \mathcal{P}_i$ if the non-zero elements of the projection vector p correspond only to the sensors in $\mathcal{P}_i$. i.e. $p \in \mathcal{P}_i$ iff $p_v = 0 \; \forall \; v \notin \mathcal{P}_i$. Algorithm 2 chooses the projection vector which maximises $\max_{i=1,...,n} \frac{\max_{p\in\mathcal{P}_i;p\in\mathcal{B}} \Delta H(p)}{2|\mathcal{P}_i|}$.

A shortcoming of Algorithm 2 is that the path calculation does not take the entropy reduction into consideration when it decides which nodes are to be included in the path. In Algorithms 3 and 4, we present two heuristics where the choice of nodes is determined by entropy reduction.

Algorithm 3: Greedy Path The key idea of Algorithm 3 is to find a path with a good ratio of entropy reduction to path length. The following description refers to the symbolic description of the algorithm. This algorithm loops n times where in the i-th iteration, we determine a path P_i from node v_i to the sink. After all n P_i's have been determined, we choose the projection vector p that maximises $\max_{i=1,...,n} \frac{\max_{p\in P_i;p\in\mathcal{B}} \Delta H(p)}{2|P_i|}$. The heart of the algorithm is in the **for**-loop in lines 5–10 where P_i is determined. The variable P_i, which is the set of nodes that are to be included in the path, is initialised to contain the node v_i. Given that v_i is h_i hops away from the sink, in the first iteration in lines 5–10, the algorithm finds all neighbours of v_i which are $h_i - 1$ hops away from the sink. These neighbours are put into the set N_j (line 6). The algorithm then runs through all the combinations $\{v_i, v\}$ with $v \in N_j$ to see which of them gives the largest reduction in entropy (line 7) and this node is then included in P_i (line 8). The process then repeats and in each iteration, a node is added to P_i and the nodes in P_i become a hop closer to the sink. The output of the algorithm (line 15) is the path to be used where the nodes on the path are stored in $P_{\hat{i}}$ and the corresponding projection vector $\hat{p}$.

Algorithm 3 Algorithm 3

1: **Input:** $[s, \Delta H(p), G = (V, E)]$
2: **for** $i \in V = \{1, ..., n\}$ **do**
3: current_node $= v_i$; $P_i = \{v_i\}$
4: $h_i =$ number of hops v_i is from the sink
5: **for** $j = h_i - 1, h_i - 2, ..., 1$ **do**
6: $N_j =$ the set of neighbours of current_node that are j hops away from the sink
7: $u_m = \arg\max_{v\in N_j} \max_{p\in P_i\cup\{v\};p\in\mathcal{B}} \Delta H(p)$
8: $P_i \leftarrow P_i \cup \{u_m\}$
9: current_node $= u_m$
10: **end for**
11: $Q_i = \frac{\max_{p\in P_i;p\in\mathcal{B}} \Delta H(p)}{2h}$
12: **end for**
13: $\hat{i} = \arg\max_{i=1,..,n} Q_i$
14: $\hat{p} = \arg\max_{p\in P_{\hat{i}};p\in\mathcal{B}} \Delta H(p)$
15: **Output:** $P_{\hat{i}}$ and $\hat{p}$

Algorithm 4: Greedy Tour A weakness of Algorithm 3 is that the projection is computed along a path from the sink to v_i, which means the message will go through the intermediate nodes in the path twice. It is possible to achieve greater reduction in entropy if the nodes in the forward path from the sink to v_i are different from those in the reverse path from v_i to

the sink. Therefore, Algorithm 4 finds a tour through the node v_i $(i = 1, ..., n)$ and chooses the tour that gives the best entropy reduction to energy expenditure ratio. The following exposition refers to the symbolic description of Algorithm 10. Algorithm 4 aims to find two node-disjoint paths from v_i to the sink if possible. It finds the first path from v_i to the sink using Algorithm 3 (line 5 of Algorithm 4). It then runs another **for**-loop (lines 6-12) to find a node-disjoint path from v_i to the sink. Note that this **for**-loop is essentially identical to that in lines 5-10 in Algorithm 3 except that it considers only nodes that have not been chosen already in order to achieve node-disjointness (line 7). The rest of the Algorithm 4 is essentially the same as Algorithm 3.

Algorithm 4 Algorithm 4

1: **Input:** $[s, \Delta H(p), G = (V, E)]$
2: **for** $i \in V = \{1, ..., n\}$ **do**
3: current_node $= v_i$; $P_i = \{v_i\}$
4: $h_i =$ number of hops v_i is from the sink
5: Run lines 5–10 of Algorithm 3
6: **for** $j = h_i - 1, h_i - 2, ..., 1$ **do**
7: $N_j =$ the set of neighbours of current_node that are j hops away from the sink and not in P_i
8: **if** $N_j = \emptyset$, break the inner **for**-loop;**else** continue
9: $u_m = \arg\max_{v \in N_j} \max_{p \in P_i \cup \{v\}; p \in \mathcal{B}} \Delta H(p)$
10: $P_i \leftarrow P_i \cup \{u_m\}$
11: current_node $= u_m$
12: **end for**
13: $Q_i = \frac{\max_{p \in P_i; p \in \mathcal{B}} \Delta H(p)}{2h}$
14: **end for**
15: $\hat{i} = \arg\max_{i=1,..,n} Q_i$
16: $\hat{p} = \arg\max_{p \in P_{\hat{i}}; p \in \mathcal{B}} \Delta H(p)$
17: **Output:** $P_{\hat{i}}$ and $\hat{p}$

Complexity analysis of algorithms The aim of this section is to study the complexity of Algorithms 1–4. In the complexity analysis below, we assume that

1. The matrix Θ_1 has already been computed. Therefore, the following analysis does not include the Bayesian compressive sensing algorithm calculations and the computation of Θ_1. Note that the computation of Θ_1 requires the calculating the matrix V in Lemma 1. This can be done by performing a QR factorisation [9] on Φ^T, which has a complexity $\mathcal{O}(n^3)$.

2. The sink (or cluster head) has already computed the shortest path tree of the network (or cluster).

Based on the above assumptions, the analysis below counts only the operations in Algorithms 1–4.

For Algorithm 1, it can readily be shown that the complexity is $\mathcal{O}(n)$.

Algorithms 2–4 require computing the largest eigenvalue and the corresponding eigenvectors of some sub-matrices of Θ_1. There are a number of algorithms to do this. For example, the Power method [2] return the largest eigenvalue and the corresponding eigenvector without having to compute all the other eigenvalues. However, the convergence of Power method depends on the ratio between the largest and the second largest eigenvalue, which is not known beforehand generally. Instead, we have chosen to use algorithms which compute the entire set of eignevalues and eigenvectors, e.g. base on Q-R iteration. For a matrix of size ℓ-by-ℓ, these algorithms have a complexity of $\mathcal{O}(\ell^3)$. Let us assume that sensor node i is h_i hops away from the sink, then the complexity of Algorithm 2 is $\sum_{i\in V}\mathcal{O}(h_i^3)$. Let κ be an upper bound on the node degree, then the complexity of algorithms 3 and 4 are both $O(\kappa\sum_{i\in V}\sum_{j=2}^{h_i} j^3)$.

3.4 Discussion

Note that typically choices of projection vectors in the compressive sensing literature are dense, e.g. Rademacher projection with $\{+1,-1\}$ with equal probability [8] or Gaussian distributed elements [4]. There is some recent effort in [20] to use sparse projection vector. We can view collecting raw sample values and computing projection over a short path as a sparse projection vector.

Note that whether direct sampling of the data is good or not depends on the coherence between the standard vector basis and the basis B. In [4], the coherence between a projection basis matrix Φ and a basis B is defined as

$$c(\Phi, B) \quad = \quad \sqrt{n}\ \ max_{1\le k,j\le n}|\phi_i^T b_j| \tag{3.5}$$

where ϕ_i and b_j are respectively the i-th column of Φ and j-column of B. It can readily be shown that the coherence between two orthonormal bases is between 1 and $\sqrt{n}$. If two bases have small coherence, then compressive sensing is likely to give good results. For example, for Φ being the identity matrix (which corresponds to sampling the data) and B the Discrete Cosine Transform (DCT) basis, the coherence between them can be shown to be $\sqrt{2}$, which is a small coherence if n is sufficiently large. Thus random sampling should work well in the DCT basis.

4 Performance evaluation

We will illustrate the performance of our algorithms by using simulation as well as data collected from an outdoor WSN.

4.1 Simulation

We consider a WSN with 256 sensors arranged in a regular square grid in a two-dimensional plane. We assume the sensors are located at $(i-8-\frac{1}{2}, j-8-\frac{1}{2})$ for $i,j\in\{1,2,...,16\}$ on the (x,y)-plane. The sink is located at the centre of the square grid at (0,0). Two nodes in the WSN are connected if they are within a distance of 1.5 units of each other. Therefore a

node has a maximum of 8 neighbours. We assume that a shortest path tree has already been built in the network.

We generate correlated data on the sensor network using an algorithm similar to the one used in [16]. Let d_{ij} be the distance between sensors i and j, then we assume that the correlation of the data between them is given by $\exp(-0.5d_{ij})$. Let C be the resulting correlation matrix. We compute the Cholesky factorisation of $C = LL^T$ and generate a realisation of the noise-free data by Lz where z is a vector of *i.i.d.* random numbers. Sensor noise with variance of 0.005 is then added to the noise-free data.

We first show that the data set generated is compressible in the Discrete Cosine Transform (DCT) [18] basis. Figure 4.1 plots the DCT coefficients of the data (shown in 'x' in the figure). The two straight lines in the graph show the decay in the magnitude of coefficients if it obeys the power law with exponent $-\frac{1}{2}$ and -1. Given that the coefficients has a decay according to the power law, the noise free data is compressible.

We apply the generic adaptive compressive sensing algorithm (Algorithm 0) to the data. We assume that in the initial part of the algorithm (in line 1), 80 random sensors return their noisy readings to the sink and 80 additional projections are performed in the loop between lines 2–6. Algorithms 1–4, which are described in Section 3.3, are used to determine a good projection vector (line 3). In addition, we use the following three algorithms as references:

- Algorithm MaxEnt: The sink uses the projection vector that gives the maximum reduction in entropy, i.e solution of $\max_{p\in\mathcal{B}} \Delta H(p)$. Note that this p is non-sparse, so we expect high energy consumption but good accuracy. Note that for this algorithm, $\Delta H(p)$ uses the exact entropy reduction formula given in Lemma 1 rather than the approximation formula in Section 3.3.

- Algorithm MaxEntNode: The sink asks the sensor that gives maximum reduction in entropy to return its reading.

- Algorithm Random: The sink randomly asks one of the sensors that has not been queried before to return its reading.

We choose two different metrics to measure the performance. For accuracy, we use the relative reconstruction error $\frac{\|x-\hat{x}\|}{\|x\|}$ defined in Section 2. For energy, we count the total number of transmissions used till each iteration. For example, if 50 additional projections have been used, then the total number of transmissions include those coming from the 80 initial sensor readings and the 50 additional projections. Let J represent the number of transmissions then we express our results as $\frac{J}{J_{\text{ref}}}$ where J_{ref} is the total number of transmissions required by the sensors if they all send their data to the sink, i.e. if $\frac{J}{J_{\text{ref}}} \geq 1$ then the method does no better than the sensor nodes simply return all their data to the sink.

Figure 4.2 compares the performance of Algorithms 1–4 against that of the reference algorithms MaxEntNode and Random. The results are obtained from the average of 100 simulation experiments. It can be seen that Algorithms 3 and 4 — which make use of a well chosen path or tour to balance accuracy and energy consumption — give the best result in the sense that, for a given energy consumption, these two algorithms give the least relative error.

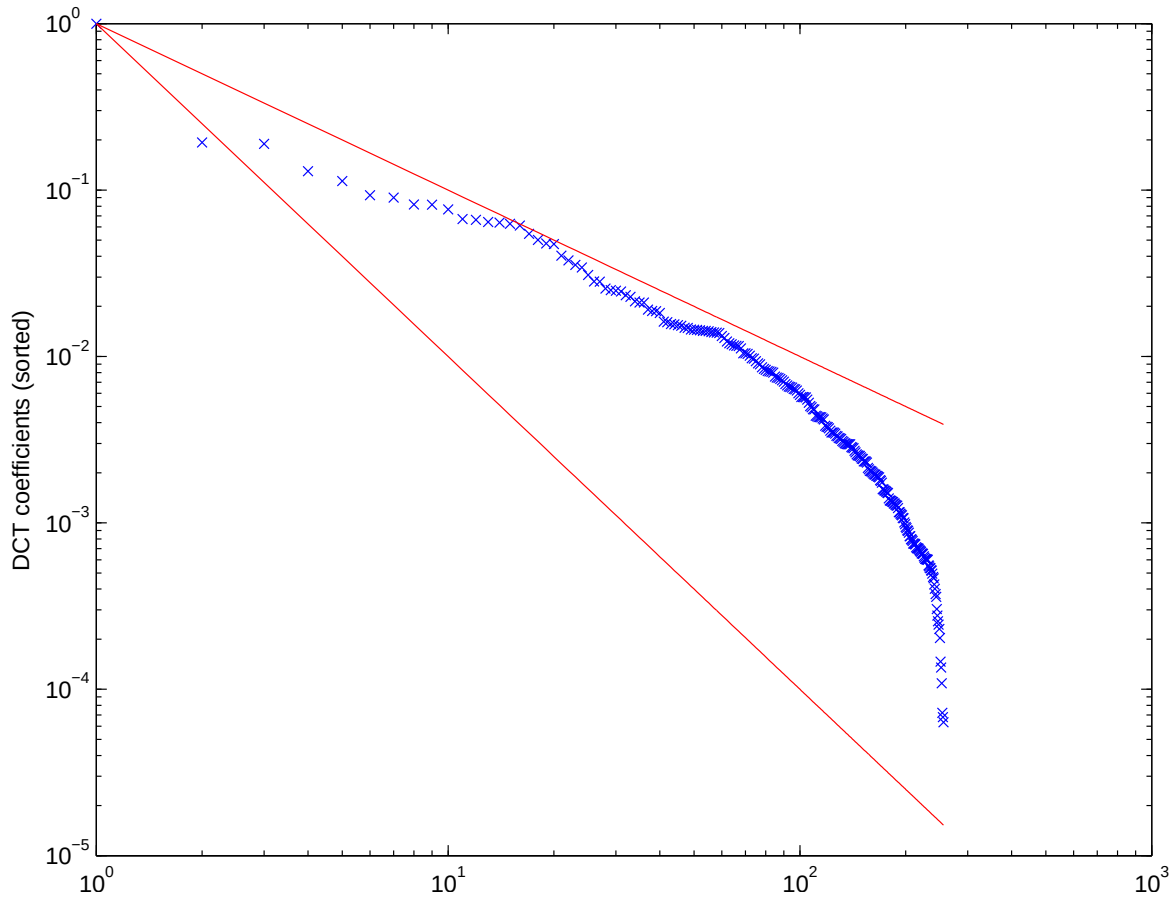

Figure 4.1: Compressibility of data.

Figure 4.3 compares the performance of Algorithm 4 with the reference algorithm MaxEnt. It shows that MaxEnt can give a lower relative error after the same number of projections but it also requires much more energy because the optimal projection vector for MaxEnt is non-sparse. In fact, Figure 4.3 shows that the energy consumption of MaxEnt is worse than simply having the sensors return the data to the sink. This demonstrates that the existing adaptive compressive sensing algorithms cannot be applied directly to WSNs because they do not take energy expenditure to acquire projections into consideration. Note that we have elected to plot only the results for Algorithm 4 in Figure 4.3 because of the horizontal scale needed to plot the result of MaxEnt means that the results of Algorithms 1-4 will all be cluttered near the left hand side of the graph which renders them indistinguishable from each other.

4.2 Outdoor WSN

We evaluate the performance of our algorithms using the sensor readings collected from the CSIRO sensor deployment in Belmont, Queensland, Australia. As shown in Fig. 4.4 that

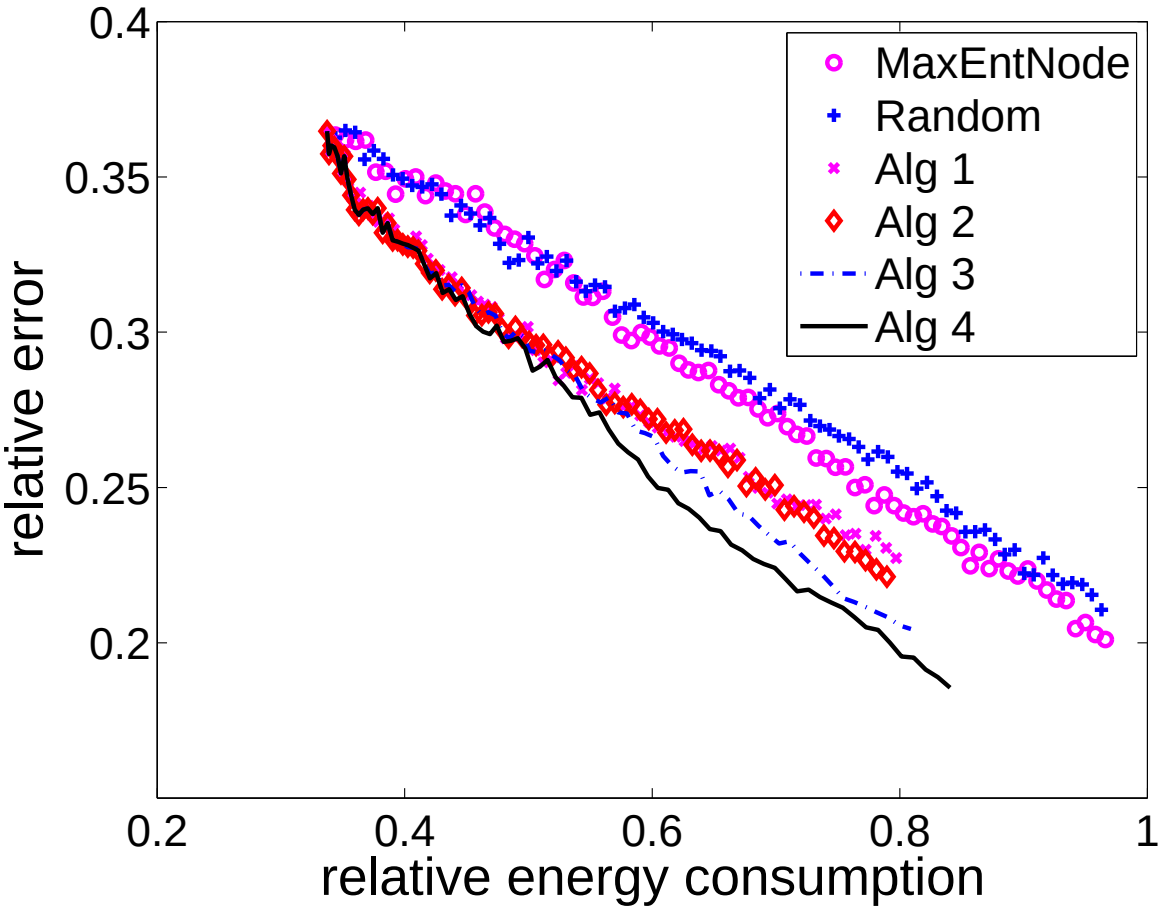

Figure 4.2: Performance of Algorithms 1–4 and reference algorithms MaxEntNode and Random.

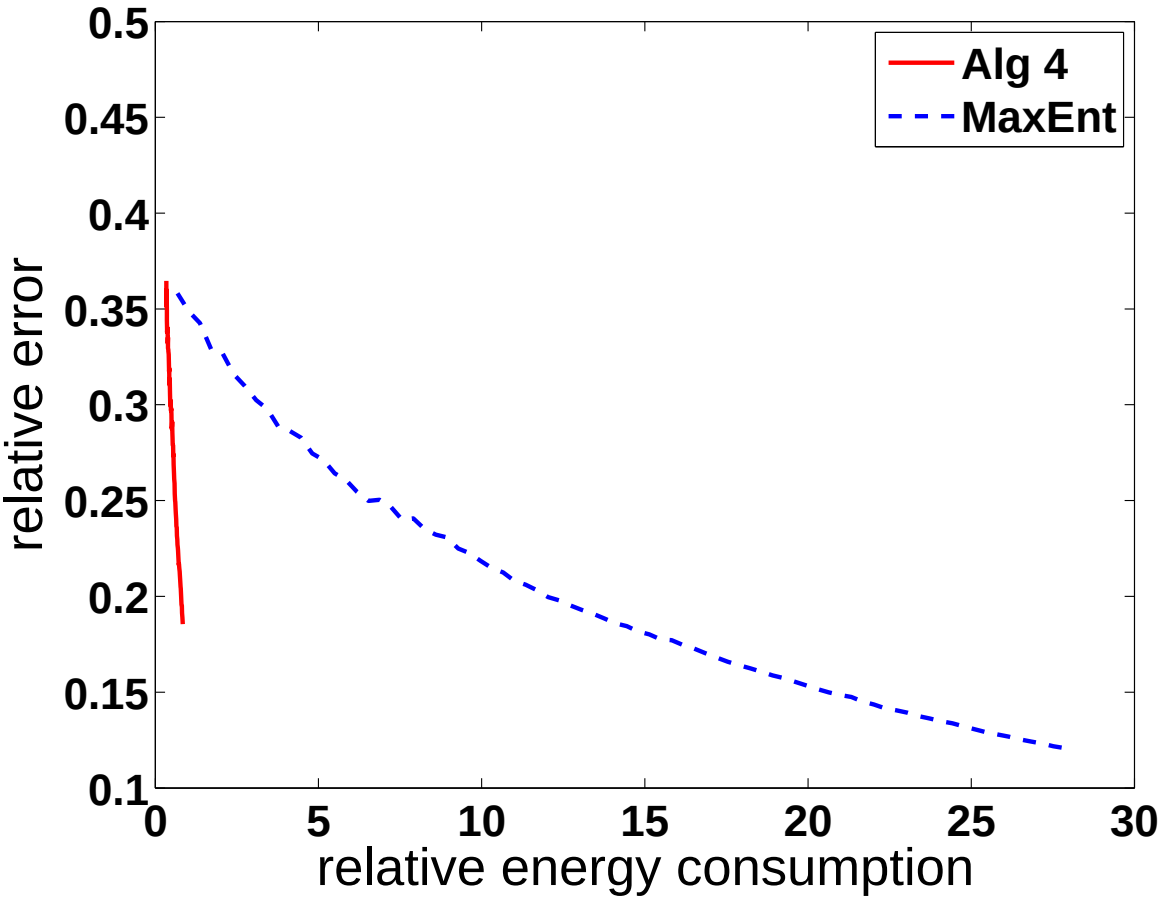

Figure 4.3: Performance of Algorithms 3–4 and reference algorithms MaxEnt.

there are 32 nodes in the Belmont deployment which are arranged in a 4×8 grid. The sink node is located to the left of this grid. Nodes in this deployment have similar connectivity to those discussed in Sect. 4.1, i.e. each of them can have maximum eight neighbors. We synchronized the nodes and collected one month of temperature data with a sampling interval of one minute. The number of snapshots used is 42,646.

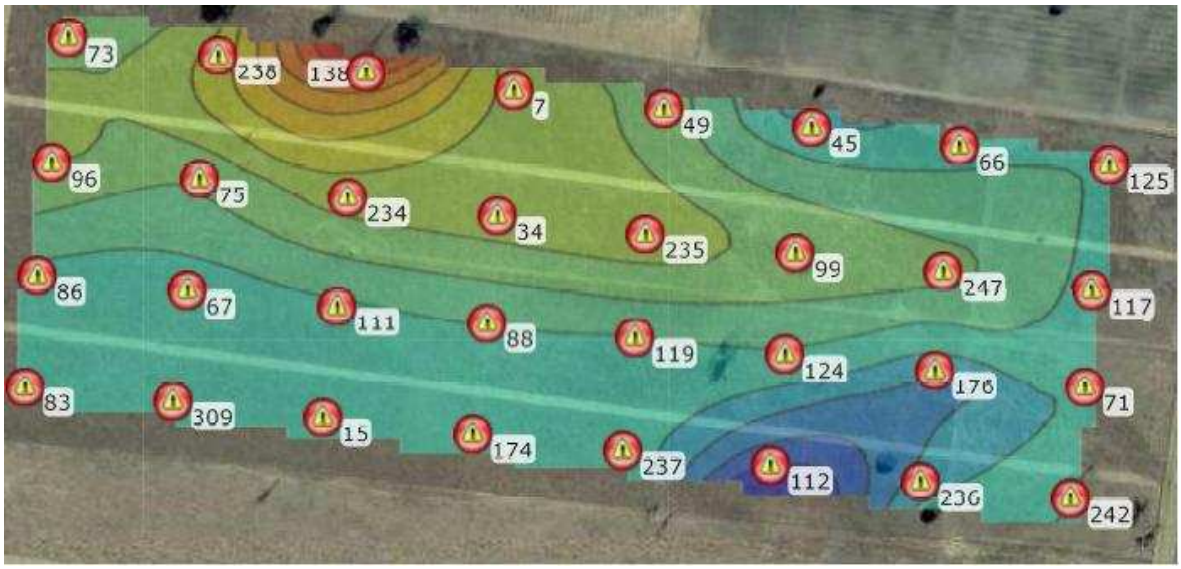

Figure 4.4: Sensor arrangement in Belmont, The sink node is neighbor to node 96,86 and 83.

Basis selection for microclimate data

In order to find an appropriate basis for the temperature signal, we compute its representation in a number of bases, including DCT, Fourier and different wavelets such as Haar, Daubechies (D4), Symlets, Coiflets, and Splines etc. For a given spatial signal and for a given basis, we compute the relative reconstruction error between the original signal and its approximation by retaining only the largest k ($k = 1, 2, ...$) coefficients in that basis. Repeating this process for all the bases, we make a comparison among all the bases in terms of percentage of largest coefficients and corresponding reconstruction error. We generalize the result by conducting this comparison for each of the spatial signals of the one month period. Fig. 4.5 shows this comparison for DCT, Fourier and D4 basis (Note that, we have presented result using only best three bases to avoid cluttering of images). We observe that, for the same percentage of coefficients, the representation in DCT gives a lower error compared to the other bases.

Performing the above comparison, we identify that the temperature signal has the most compressible representation in DCT. However, a question may be posed here that whether the representation of the temperature signal in DCT satisfies the requirements of a compressible signal. To answer this question we plot (see Fig. 4.6) the DCT coefficients (absolute values) in descending order of magnitude along with $\ell^{-1/\beta}$ (As discussed in Sect. 2). We find that for $\beta = 1$, each of the ordered coefficients decays quicker than $\ell^{-1/\beta}$. This result implies that the temperature signal is sufficiently compressible in DCT. Compressibility of the temperature

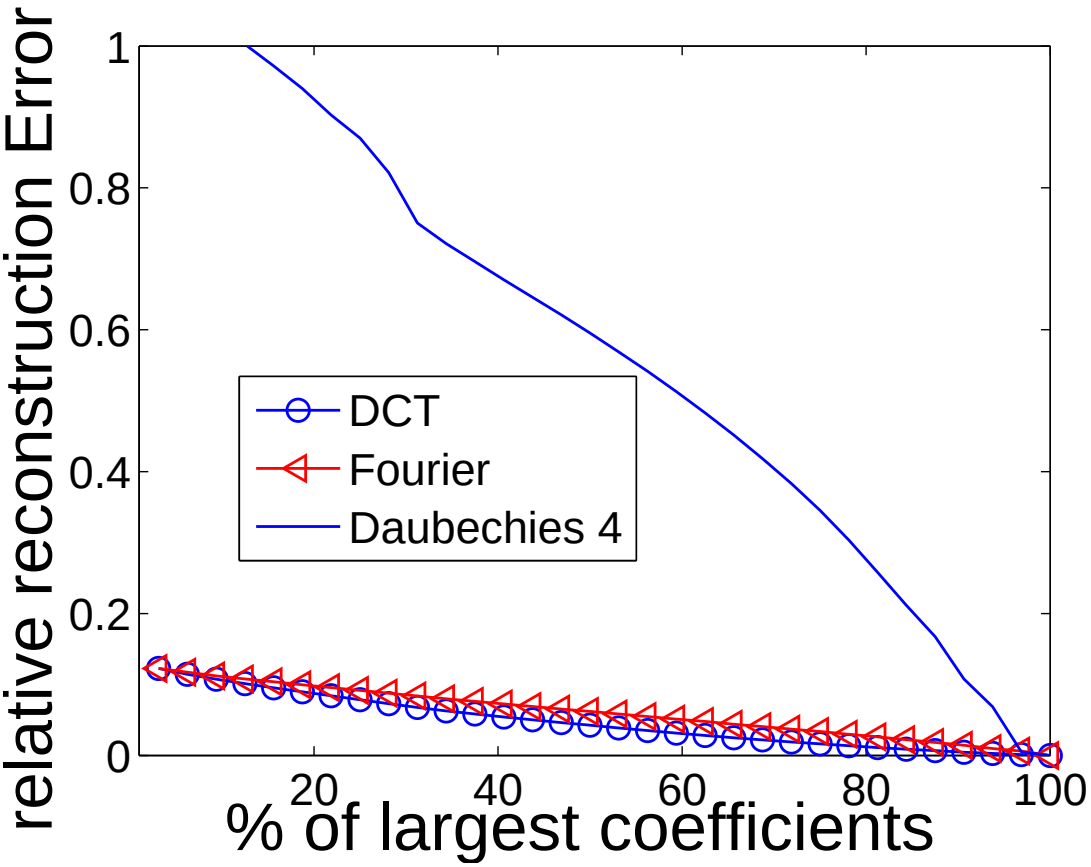

Figure 4.5: Basis selection: temperature

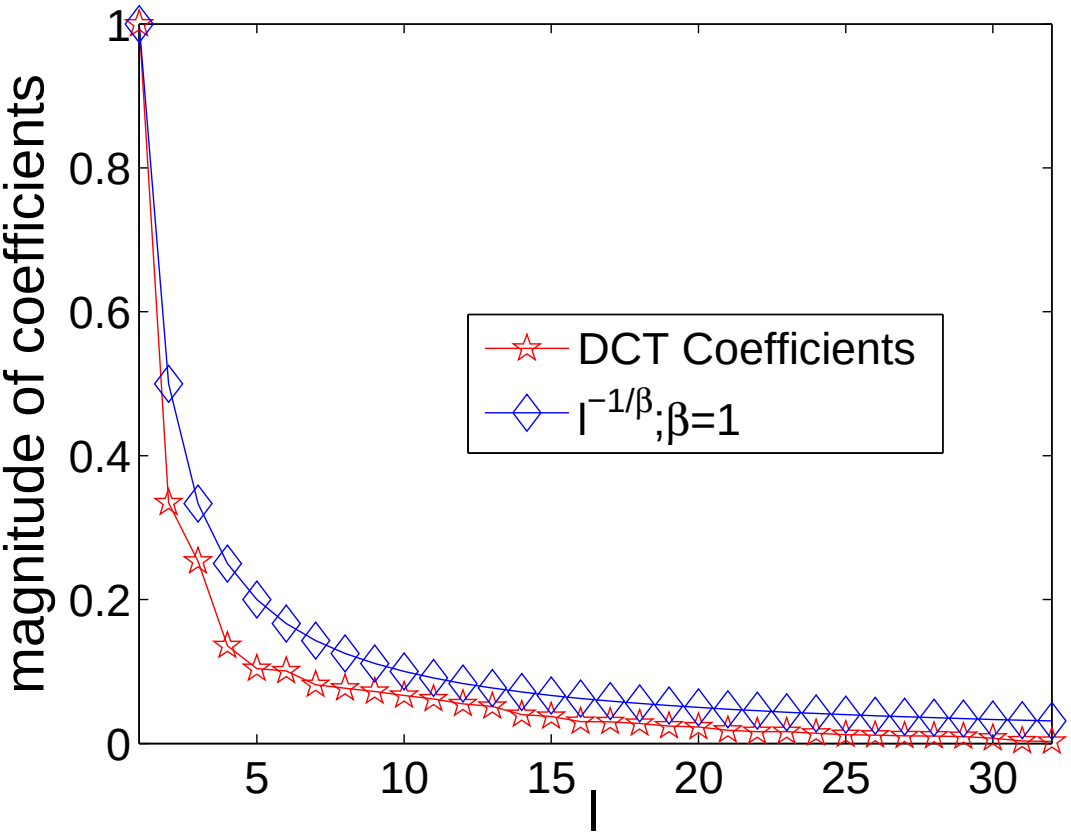

Figure 4.6: Decay of DCT coefficents

signal in DCT can also be found intuitively form Fig. 4.5. This figure shows that it requires only 10% of the DCT coefficients to keep the relative reconstruction error within 0.1.

Performance of adaptive sensing algorithms

We apply the generic adaptive compressive sensing algorithm (Algorithm 0) to the 42,646 snapshots of temperature data using DCT as the basis. We assume that in the initial part of the algorithm (line 1), 10 random sensors return their readings to the sink and 10 additional projections are performed in the loop between lines $2-6$. We use Algorithms 1–4 and MaxEntNode to determine a projection vector. Fig. 4.7 compares the performance of Algorithms $1-4$ against that of the reference algorithms MaxEntNode where the average relative error and average energy consumption is plotted. Similar to the results in Section 4.1, Algorithms 3 and 4 give the least relative error for a given energy expenditure. These two algorithms give 11% relative error in the estimated data field for an energy saving of 40%.

5 Related work

There is a rich literature on adaptive sampling for WSNs, see e.g. [13, 12, 7, 16]. Our work distinguishes from these earlier works in that it uses adaptive compressive sensing. This means that our focus is on computing a good projection, which is a generalised form of measurements. However, the earlier works use the traditional form of measurement which means that they decide whether to collect sensing data from a specific sensor at a specific time or not. The computation of projection also introduces a new cross-layer routing problem that has not been studied before.

The paper [1] proposes to compute projections in WSNs by using an additive MAC channel. This work is complementary to ours in the sense that it proposes an alternative method to compute projections. However, [1] does not use adaptation in determining its projection. We expect that the work in [1] can benefit from adaptation. The paper [17] proposes to obtain projections in WSNs via gossiping; however, the paper does not study energy consumption requirement.

6 Conclusions and future works

This paper has proposed a framework to adaptively collect information from a wireless sensor networks using adaptive compressive sensing taking into account both energy consumption and the amount of information in the sensing data. We show that the problem of computing a projection in adaptive compressive sensing that maximises information gain to energy expenditure is NP-hard and we propose a number of heuristics to solve this problem. Our performance evaluation with simulated data and real data shows that our algorithms gives accurate estimation of the unknown data for a given energy expenditure. Our work currently considers only snapshots of temporal-spatial data. An extension is to consider the full temporal-spatial data taking both temporal and spatial correlation into consideration. In ad-

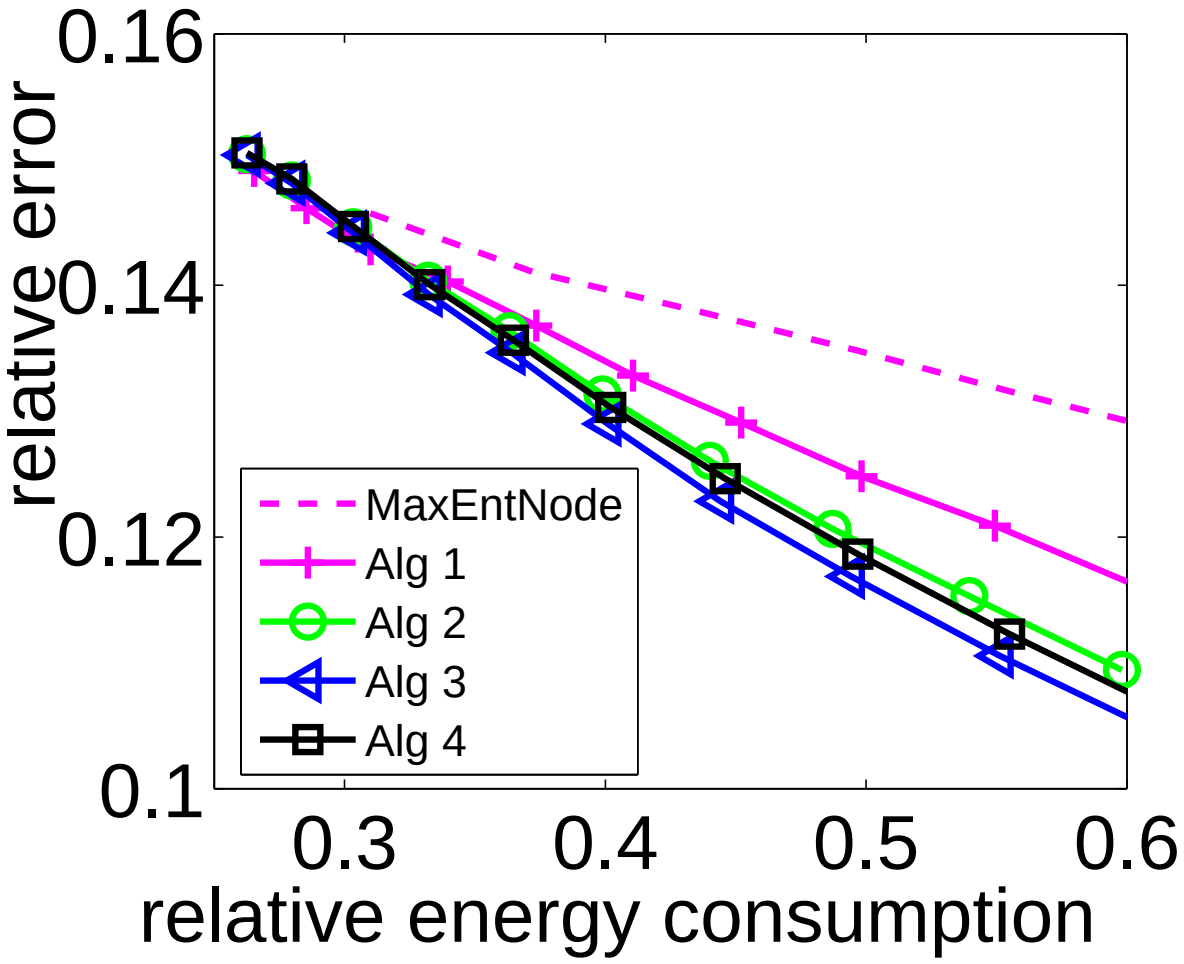

Figure 4.7: Reconstruction accuracy and energy consumption trade-off while reconstructing temperature signal

dition, current work measures energy expenditure using the number of wireless transmissions assuming no packet loss. An extension is to consider wireless environment with packet loss.

Acknowledgment

The authors thank CSIRO for providing the data used in Section 4.2. This research is partially supported by the Australian Research Council Discovery Project DP0770523.

A Proof of Lemma 1

Using the same method in [11], it can be shown that the reduction of entropy by using an additional projection vector p when the data equation is given as in equation (3.2) is:

$$\begin{aligned}\Delta H(p) = & \tfrac{1}{2}\log\det(A+\frac{1}{\hat{\sigma}^2}B^T\breve{\Phi}^T(\breve{\Phi}\breve{\Phi}^T)^{-1}\breve{\Phi}B) - \\ & \tfrac{1}{2}\log\det(A+\frac{1}{\hat{\sigma}^2}B^T\Phi^T(\Phi\Phi^T)^{-1}\Phi B)\end{aligned} \tag{A.1}$$

where det is the determinant of a matrix and

$$\breve{\Phi} \;=\; \left[\begin{array}{c}\Phi\\ p^T\end{array}\right]. \tag{A.2}$$

The matrix A contains the hyper-parameters of Bayesian estimation and is related to the posteriori covariance Σ by

$$\Sigma^{-1} \;=\; A+\frac{1}{\hat{\sigma}^2}B^T\Phi^T(\Phi\Phi^T)^{-1}\Phi B \tag{A.3}$$

Define

$$\delta \;=\; 1-p^T\Phi^T(\Phi\Phi^T)^{-1}\Phi p \tag{A.4}$$

$$\Upsilon \;=\; \Phi^T(\Phi\Phi^T)^{-1}\Phi \tag{A.5}$$

and assume that $\delta \neq 0$.

By applying the matrix inversion lemma [9, p.18] to the block-structured matrix $(\breve{\Phi}\breve{\Phi}^T)^{-1}$, we have

$$\breve{\Phi}^T(\breve{\Phi}\breve{\Phi}^T)^{-1}\breve{\Phi} \tag{A.6}$$

$$= \left[\begin{array}{cc}\Phi^T & p\end{array}\right]\left[\begin{array}{cc}\Phi\Phi^T & \Phi p\\ p^T\Phi & 1\end{array}\right]^{-1}\left[\begin{array}{c}\Phi\\ p^T\end{array}\right] \tag{A.7}$$

$$= \left[\begin{array}{cc}\Phi^T & p\end{array}\right]\left[\begin{array}{cc}(\Phi\Phi^T)^{-1}+\frac{1}{\delta}(\Phi\Phi^T)^{-1}\Phi pp^T\Phi^T(\Phi\Phi^T)^{-1} & -\frac{1}{\delta}(\Phi\Phi^T)^{-1}\Phi p\\ -\frac{1}{\delta}p^T\Phi^T(\Phi\Phi^T)^{-1} & \frac{1}{\delta}\end{array}\right]\left[\begin{array}{c}\Phi\\ p^T\end{array}\right] \tag{A.8}$$

$$= \Upsilon+\frac{1}{\delta}(I-\Upsilon)pp^T(I-\Upsilon) \tag{A.9}$$

Equation (A.1) can now be written as

$$\begin{aligned}
& \Delta H(p) && \text{(A.10)}\\
= \; & \tfrac{1}{2}\log\det(A+\frac{1}{\hat{\sigma}^2}B^T(\Upsilon+\frac{1}{\delta}(I-\Upsilon)pp^T(I-\Upsilon)B)) - \tfrac{1}{2}\log\det(A+\frac{1}{\hat{\sigma}^2}B^T\Upsilon B) && \text{(A.11)}\\
= \; & \tfrac{1}{2}\log\det(\Sigma^{-1}+\frac{1}{\hat{\sigma}^2}\frac{1}{\delta}B^T(I-\Upsilon)pp^T(I-\Upsilon)B) - \tfrac{1}{2}\log\det(\Sigma^{-1}) && \text{(A.12)}\\
= \; & \tfrac{1}{2}\log\det(I+\frac{1}{\hat{\sigma}^2}\frac{1}{\delta}\Sigma^{\frac{1}{2}}B^T(I-\Upsilon)pp^T(I-\Upsilon)B\Sigma^{\frac{1}{2}}) && \text{(A.13)}\\
= \; & \tfrac{1}{2}\log(1+\frac{1}{\hat{\sigma}^2}\frac{1}{\delta}p^T(I-\Upsilon)B\Sigma B^T(I-\Upsilon)p) && \text{(A.14)}
\end{aligned}$$

where I denotes the identity matrix. Note that in the above derivations, we have made used of the following results from matrix theory: (1) $\det(XY)=\det(X)\det(Y)$ for square matrices X and Y of the same dimension. (2) $\det(I+ab^T)=1+b^Ta$ for any two vectors a and b of the same dimension.

Note that Φ is in general a fat matrix (i.e. has more columns than rows). Let the singular value decomposition (SVD) [9] of Φ be

$$\Upsilon \;=\; U\begin{bmatrix} S & 0 \end{bmatrix}\begin{bmatrix} \bar{V}^T \\ V^T \end{bmatrix} \qquad \text{(A.15)}$$

where U and $\begin{bmatrix} \bar{V}^T \\ V^T \end{bmatrix}$ are orthronormal matrices, and S is a diagonal matrix containing the singular values. Note that the matrices in the above SVD have been partitioned into sub-matrices with compatible dimension. Assuming that Φ has full row-rank, then the matrix V is in fact an orthronormal basis of the null space of Φ. Given the above SVD, we can readily show that

$$I-\Upsilon \;=\; VV^T \qquad \text{(A.16)}$$

By substituting equation (A.16) into equation (A.14), we have

$$\Delta H(p) \;=\; \tfrac{1}{2}\log(1+\frac{1}{\hat{\sigma}^2}\frac{p^TVV^TB\Sigma B^TVV^Tp}{p^TVV^Tp}) \qquad \text{(A.17)}$$

which is the desired result. Note that we have used

$$\delta = 1-p^T\Phi^T(\Phi\Phi^T)^{-1}\Phi p = p^T(I-\Upsilon)p = p^TVV^Tp \qquad \text{(A.18)}$$

and the fact that p has unit norm.

Bibliography

[1] W. Bajwa et al., Joint Source-Channel Communication for Distributed Estimation in Sensor Networks. IEEE Trans Information Theory, 2007.

[2] R. L. Burden and J. D. Faires, Numerical Analysis. Brooks-Cole Publishing, 2000.

[3] E. J. Candes and M. Wakin. An introduction to compressive sampling. IEEE Signal Processing Magazine, March 2008.

[4] E. J. Candes. Compressive sampling. Proceedings of the International Congress of Mathematicians, Madrid, Spain, 2006.

[5] E. J. Candes et al., Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information, IEEE Trans. Inf. Theory, 2006.

[6] V.V. Fedorov. Theory of Optimal Experiments. Academic Press, 1972.

[7] H. Gupta, V. Navda, S. Das and V. Chowdhary. Efficient gathering of correlated data in sensor networks. In ACM Trans. on Sensor Networks, 2008.

[8] J. Haupt, R. D. Nowak: Signal Reconstruction From Noisy Random Projections. IEEE Transactions on Information Theory 52(9): 4036-4048 (2006)

[9] R.A. Horn and C.R. Johnson. *Matrix Analysis*, Cambridge University Press, 1988.

[10] W. Hu, C. T. Chou, S. Jha, N. Bulusu: Deploying long-lived and cost-effective hybrid sensor networks. Ad Hoc Networks 4(6): 749-767 (2006)

[11] S. Ji, Y. Xue, and L. Carin. Bayesian Compressive Sensing. IEEE Trans. Signal Processing, 2008.

[12] C. Liu, K. Wu and J. Pei. An energy-efficient data collection framework for wireless sensor networks by exploiting spatialtemporal correlation. IEEE Trans. on Parallel and Distributed Systems, 2007.

[13] J. Kho, A. Rogers, N.R. Jennings. Decentralised control of adaptive sampling in wireless sensor networks. ACM Trans. on Sensor Networks, 2009.

[14] MacKay, D. J. C. 1991. Bayesian interpolation. Neural Comp. 4.

[15] B. Moghaddam, Y. Weiss, and S. Avidan. Generalized spectral bounds for sparse LDA. In Proc. ICML, 2006.

[16] A. Muttreja, A. Raghunathan, S. Ravi and N.K. Jha, Active Learning Driven Data Acquisition for Sensor Networks. Proc. of the 11th IEEE Sym. on Computers and Communications (ISCC'06), 2006.

[17] M. Rabbat et al., Decentralized compression on predistribution via randomized gossiping. Proc. IPSN 2006.

[18] K. Sayood. Introduction to Data Compression, Morgan Kaufmann, 2000.

[19] M.W. Seeger, Bayesian Inference and Optimal Design for he Sparse Linear Model. Journal of Machine Learning Research, 2008.

[20] Wei Wang, Minos Garofalakis and Kannan Ramchandran. Distributed Sparse Random Projections for Renable Approximation. IPSN 2007.