Cooperative adaptive sampling via approximate entropy maximization

Rishi Graham Jorge Cortés

***Abstract*— This work deals with a group of mobile sensors sampling a spatiotemporal random field whose mean is unknown and covariance is known up to a scaling parameter. The Bayesian posterior predictive entropy provides a direct mapping between the locations of a new set of point measurements and the uncertainty of the resulting estimate of the model parameters. Since the posterior predictive entropy and its gradient are not amenable to distributed computation, we propose an alternative objective function based on a Taylor series approximation. We present a distributed strategy for sequential design which ensures that measurements at each timestep are taken at local minima of the objective function. The technical approach builds on a novel reformulation of the posterior predictive entropy.**

I. Introduction

Problem motivation: The growing availability and sophistication of robotic sensor technology is enabling a new generation of statistical experimentation. We envision networks of mobile sensing platforms executing intelligent sampling missions, taking advantage of the robust adaptability of multi-agent systems. We are particularly interested in strategies which use modern statistical techniques to give a full accounting of predictive and inferential uncertainty.

This work proposes an entropy-based cooperative strategy for optimal sampling of spatiotemporal processes. In statistical optimal design, the optimality criteria for estimation depend upon the goal of the experiment. Here we focus on learning the parameters of the model and define optimality in terms of entropy as a measure of information about the *model*. Since entropy is inherently linked to the cross correlation of measurements, the main challenge we face is the synthesis of distributed adaptive sampling strategies.

Literature review: Complex statistical techniques exist to model the evolution of physical phenomena. Of particular relevance to this work are [1], [2], [3], regarding statistical models, and [4], [5], regarding optimal design of experiments. The concept of *adaptive design* is generally approached with one of two goals in mind. Minimizing uncertainty in field estimates [6], [7] over new measurements results in A-optimal designs, while maximizing the information gained about the model [8] results in D-optimal designs. Comparisons of these two methods may be found in [9], [10]. In this paper, we concentrate on the latter in the form of entropy maximization.

In cooperative control, various works consider mobile sensor networks performing spatial estimation tasks. [11] introduces performance metrics for oceanographic surveys by autonomous underwater vehicles. [12] considers a robotic sensor network with centralized control estimating a static field from measurements with both sensing and localization error. In [13], a deterministic model is used, where the random elements come in the form of unknown model parameters, and localization error is included. Here we treat the field itself as random with a Gaussian process model. [14] considers optimal sampling trajectories from a parameterized set of paths. In [15], [16] the focus is on estimating deterministic fields when the measurements taken by individual robots are uncorrelated. [17] discusses the tracking of level curves in a noisy scalar field. In previous work we have considered optimal trajectories for minimizing the average error variance of random field estimates using models with known [18] and unknown [19] covariance.

Statement of contributions: We model the physical process of interest as a spatiotemporal random field with mean unknown and covariance known up to a scaling parameter and estimate it from a Bayesian perspective. First, we extend a known univariate representation of the posterior predictive entropy to the multivariate case. From this result, we develop an aggregate objective function that quantifies the goodness of network configurations for the estimation of the spatial field. We show that the proposed objective function is a second-order approximation of the posterior predictive entropy, characterize its smoothness properties, and describe a distributed method to compute it. We employ average consensus and distributed Jacobi overrelaxation algorithms to compute the objective function and its gradient in a distributed way across a network composed of robotic agents and static nodes. Finally, we synthesize a distributed motion coordination for adaptive sampling based on one-step-ahead local optimization of data collection. We conclude illustrating the performance of the algorithm in simulation.

II. Preliminary notions

Let $\mathbb{R}$, $\mathbb{R}_{>0}$, and $\mathbb{R}_{\geq 0}$ denote the set of reals, positive reals and nonnegative reals, respectively. For $p \in \mathbb{R}^d$ and $r \in \mathbb{R}_{>0}$, let $\overline{B}(p,r)$ be the *closed ball* of radius r centered at p. Given two vectors $\underline{u} = (u_1, \ldots, u_a)^T$, and $\underline{v} = (v_1, \ldots, v_b)^T$, $a, b \in \mathbb{Z}_{>0}$, we denote by $(\underline{u}, \underline{v})$ the concatenation $(\underline{u}, \underline{v}) = (u_1, \ldots, u_a, v_1, \ldots, v_b)^T$. We denote by ∂S the boundary of a set S. The *ϵ-contraction* of a set S, with $\epsilon > 0$, is the set $S_\epsilon = \{q \in S \mid \mathrm{d}(q, \partial S) \geq \epsilon\}$. Let $\mathrm{proj}_\Omega : \mathbb{R}^m \to \Omega$ denote the orthogonal projection onto the set Ω, i.e., $\mathrm{proj}_\Omega(x) = \mathrm{argmin}_{y \in \Omega} \|x - y\|$. A *convex polytope* is the convex hull of a finite point set. For a bounded set $S \subset \mathbb{R}^d$, we let $\mathrm{CR}(S)$ denote the *circumradius* of S. We denote by $\mathbb{F}(S)$ the collection of finite subsets of S. With a slight abuse of notation, we use $\mathbf{0}$, respectively $\boldsymbol{I}$ to represent the zero matrix, respectively identity matrix, of appropriate dimension when it is clear from the context. We consider

Rishi Graham is with the Department of Applied Mathematics and Statistics, University of California, Santa Cruz, rishig@ams.ucsc.edu

Jorge Cortés is with the Department of Mechanical and Aerospace Engineering, University of California, San Diego, cortes@ucsd.edu

a compact and connected set $\mathcal{D} \subset \mathbb{R}^d$, $d \in \mathbb{N}$. Since we deal with a process which varies over time, let $\mathcal{D}_e = \mathcal{D} \times \mathbb{R}$ denote the space of points over $\mathcal{D}$ and time. The *Voronoi partition* $\mathcal{V}(\underline{s}) = (V_1(\underline{s}), \ldots, V_n(\underline{s}))$ of $\mathcal{D}$ generated by the points $\underline{s} = (s_1, \ldots, s_n)$ is defined by

$$V_i(\underline{s}) = \{q \in \mathcal{D} \mid \|q - s_i\| \leq \|q - s_j\|, \ \forall j \neq i\}.$$

Each $V_i(\underline{s})$ is called a *Voronoi cell*. Two points s_i and s_j are *Voronoi neighbors* if their Voronoi cells share a boundary.

For a random variable, Y, let $\mathrm{E}[Y]$ denote the expectation of Y and $\mathrm{Var}[Y]$ its variance. We use the normal, inverse gamma, and Student statistical distributions [20]. We let $\mathrm{N}_n(\boldsymbol{a}, B)$ denote the n-variate normal distribution with mean vector $\boldsymbol{a} \in \mathbb{R}^n$ and covariance matrix $B \in \mathbb{R}^{n \times n}$. We let $\Gamma^{-1}(a, b)$ denote the inverse gamma distribution with shape parameter a and scale parameter b. Let $t_n(\boldsymbol{a}, B, c)$ denote an n-variate shifted Student t distribution with mean $\boldsymbol{a}$, covariance matrix $\frac{c}{c-2}B$, and c degrees of freedom.

A. Basic linear algebraic facts

Here we present some basic facts from linear algebra [21] that will be useful throughout the paper. Let $[A]_{ij}$ denote the i,jth element of a matrix A and $\det(A)$ denote its determinant. For a matrix $A \in \mathbb{R}^{n \times m}$ let $\|A\|_F$ denote the Frobenius matrix norm, $\|A\|_F = \sqrt{\mathrm{tr}\,(A^T A)}$, where $\mathrm{tr}(A)$ denotes the trace of A. Given a partitioned symmetric matrix $A = \left[\begin{smallmatrix} B & C \\ C^T & D \end{smallmatrix}\right]$, the Schur complement of B in A is $(B\,|A) = D - C^T B^{-1} C$.

B. Bayesian modeling of space-time processes

Let Z denote a random space-time process taking values on $\mathcal{D}_e$. We deal with both *sampled* (known) and *unsampled* (hypothetical) realizations of Z. To distinguish, we attach the subscripts s and u to associated vectors and matrices. Let $\underline{y}_s = (y_{s:1}, \ldots, y_{s:m})^T \in \mathbb{R}^m$ be $m \in \mathbb{N}$ samples taken from Z at corresponding locations $\underline{x}_s = (x_{s:1}, \ldots, x_{s:m})^T \in \mathcal{D}_e^m$, with $x_{s:i} = (s_{s:i}, t_i)$, $i \in \{1, \ldots, m\}$. Optimal design is the process of choosing locations to take measurements in order to reduce the uncertainty of the resulting statistical inference. Since uncertainty drives the problem, it should be modeled as accurately as possible.

In a Bayesian setting, the prediction takes the form of a distribution, called the posterior predictive [22]. If the field is modeled as a Gaussian process with known covariance, the posterior predictive mean corresponds to the *Best Linear Unbiased Predictor*, and its variance corresponds to the mean-squared prediction error. If the covariance of the field is not known, however, few analytical results exist which take the full uncertainty into account. The model we present here, extending [1], [2], is one of the few which allows for uncertainty in the covariance process and still produces an analytical posterior predictive distribution.

We assume that the samples are distributed as

$$\underline{y}_s \sim \mathrm{N}_m\left(\mathbf{F}_s^T \beta, \sigma^2 \mathbf{K}_s\right). \tag{1}$$

Here $\beta \in \mathbb{R}^p$ is a vector of regression parameters, and $\sigma^2 \in \mathbb{R}_{>0}$ is the variance parameter, both unknown. $\mathbf{K}_s \in \mathbb{R}^{m \times m}$ is the known sample correlation matrix. We assume a finite spatial correlation range, $r \in \mathbb{R}$, i.e., $\|s_i - s_j\| \geq r \implies [\mathbf{K}_s]_{ij} = [\mathbf{K}_s]_{ji} = 0$. The matrix $\mathbf{F}_s \in \mathbb{R}^{p \times m}$ is determined by a set of $p \in \mathbb{N}$ known functions $f_i : \mathcal{D}_e \to \mathbb{R}$ evaluated at the locations $\underline{x}_s$. In order to ensure an analytical form for the predictive distribution, we assume conjugate priors on the parameters,

$$\beta|\sigma^2 \sim \mathrm{N}_p\left(\beta_0, \sigma^2 \mathbf{K}_0\right) \quad \text{and} \quad \sigma^2 \sim \Gamma^{-1}\left(\frac{\nu}{2}, \frac{q\nu}{2}\right). \tag{2}$$

Here $\beta_0 \in \mathbb{R}^p$, $\mathbf{K}_0 \in \mathbb{R}^{p \times p}$, and $q, \nu \in \mathbb{R}_{>0}$ are constants, known as *tuning parameters* for the model.

Given this prior information, consider a hypothetical vector of $n \in \mathbb{N}$ future measurements, which we call *unsampled*. Let $\underline{y}_u \in \mathbb{R}^n$ denote these future measurements at corresponding space-time locations $\underline{x}_u = ((s_{u:1}, t_u), \ldots, (s_{u:n}, t_u))^T \in \mathcal{D}_e^n$, and let $\mathbf{K}_u$, $\mathbf{F}_u$ denote the correlation and basis matrices analogous to the sampled versions. Note that the prior distributions are the same for both sets of measurements. Let $\underline{y} = (\underline{y}_s^T, \underline{y}_u^T)^T$ and $\underline{x} = (\underline{x}_s^T, \underline{x}_u^T)^T$ denote the combined vectors of measurements and locations, and let $\mathbf{K}$ and $\mathbf{F}$ denote the correlation and basis matrices of the combined locations. Let $\mathbf{K}_{su} = \mathbf{K}_{us}^T \in \mathbb{R}^{m \times n}$ denote the cross-covariance between $\underline{y}_s$ and $\underline{y}_u$.

From the joint distribution of $\underline{y}$ and the prior information, we may compute the posterior predictive distribution, $p(\underline{y}_u|\underline{y}_s)$, of $\underline{y}_u$ once the samples $\underline{y}_s$ have been measured. Here, the notation $p(a|b)$ denotes the conditional probability of random variable or vector a given b. An explicit form for $p(\underline{y}_u|\underline{y}_s)$ will be described in Section IV.

III. Problem statement

In Section III-A we introduce the robotic network model and in Section III-B we detail the overall network objective.

A. Robotic sensor network model

Consider a group $\{S_1, \ldots, S_m\}$ of $m \in \mathbb{N}$ static nodes deployed in a convex polytope $\mathcal{D} \subset \mathbb{R}^d$. Let $Q = (q_1, \ldots, q_m) \in \mathcal{D}^m$ denote the positions of the static nodes. Assume that each node has a limited communication radius, $R \in \mathbb{R}_{>0}$, and that they are positioned so that each one can communicate with its Voronoi neighbors.

In addition to the static nodes, consider a group $\{R_1, \ldots, R_n\}$ of n robotic sensor agents. The position of robot $i \in \{1, \ldots, n\}$ at time $t \in \mathbb{R}$ is denoted by $p_i(t) \in \mathcal{D}$. The robots take point measurements of the spatial field at discrete instants of time in $\mathbb{Z}_{\geq 0}$. Between measurement instants, each robot moves according to the discrete dynamics,

$$p_i(k+1) = p_i(k) + u_i(k),$$

where $\|u_i\| \leq u_{\max}$ for some $u_{\max} \in \mathbb{R}_{>0}$. The communication radius of the robotic agents is also R. Each node needs to be able to communicate with any robot which may be in its Voronoi region at the next timestep. To that end, assume

$$R \geq \max_{i \in \{1,\ldots,m\}} \{\mathrm{CR}(V_i(Q))\} + u_{\max}. \tag{3}$$

The robots have some limited capability of sensing each other, so that a robot knows the positions of other robots within a distance of $2u_{\max}$. At discrete timesteps, each robot sends the measurement and location to static nodes within

communication range, along with the current locations of any other sensed robots. The nodes cooperatively compute control vectors, then relay them back to the robots. The implementation does not require direct communication between robotic agents. We refer to this network model as $\mathcal{N}$.

To avoid agent collision, we further restrict the motion of the robotic agents as follows. Consider the locations $P^{(k)} = (p_1(k), \ldots, p_n(k))^T$. Between timestep k and timestep $k+1$, robot i moves within the region, $\Omega_i^{(k)} \subset \mathcal{D}$ defined by,

$$\Omega_i^{(k)} = (V_i(P^{(k)}))_{\omega/2} \cap \overline{B}(p_i(k), u_{\max}),$$

where $(V_i(P^{(k)}))_{\omega/2}$ denotes the $\omega/2$-contraction of $V_i(P^{(k)})$. This requirement combines the restriction imposed by $u_{\max}$ with a minimum distance requirement such that any two robots are always at least ω away from each other [18]. Let $\Omega^{(k)} = \prod_{i=1}^n \Omega_i^{(k)} \subset \mathcal{D}^n$ denote the region of allowed movement of all the robotic agents at timestep $k \in \mathbb{N}$.

B. Network objective

Between measurement instants, we would like to move the robots to those locations which ensure a maximum gain in information appropriate to the goal of the experiment. When the goal is to make inference about model parameters, we would like an objective function which maximizes gain in information about the model. A generally accepted practice [3], [8], [9] is to choose a set of measurement locations which maximize the entropy of the joint posterior predictive distribution. Intuitively, to maximize the gain in information we choose to measure those locations about which we currently know the least. The entropy of an arbitrary continuous distribution with pdf $p(Y)$ can be written as $\mathcal{E} = -\operatorname{E}\left[\log \frac{p(Y)}{h(Y)}\right]$, where $h(Y)$ is a reference measure chosen to ensure invariance under affine transformations of Y. When the data come from a multivariate Student t distribution, $\underline{y} \sim t_n(\mu, \Psi, \delta)$, then the entropy is [3],

$$\mathcal{E} = \frac{1}{2} \log\det\left((\delta - n + 1)\Psi\right). \tag{4}$$

IV. A distributed criterion for one-step-ahead data collection

In this section, we derive an expression for the entropy of the joint posterior predictive distribution given the model (1). As this function is not amenable to distributed computations, we propose instead an alternative which is. We finish with important smoothness properties of our proposed objective function, including an expression for its gradient.

A. Entropy of the random field estimation

We begin by extending the univariate results of [2] to provide a multivariate posterior predictive distribution with a particularly useful form for the variance.

Proposition IV.1 (Posterior predictive distribution) *Under the Bayesian model* (1)*, the posterior predictive distribution at a vector of unsampled locations $\underline{x}_u \in \mathcal{D}_e^n$ from samples $\underline{y}_s$ is a shifted Student t distribution with $\nu + m$ degrees of freedom and variance given by*

$$\operatorname{Var}[\underline{y}_u | \underline{y}_s] = \varphi(\underline{y}_s, \underline{x}_s)\phi(\underline{x}),$$

where $\varphi(\underline{y}_s, \underline{x}_s) \in \mathbb{R}$ depends on the number *of unsampled measurements, but not on their locations or values, and $\phi(\underline{x}) \in \mathbb{R}^{(n+m)^2}$ admits the expression*

$$\log\det(\phi(\underline{x})) = \log\det(\Upsilon) + \log\det(\mathbf{K}) - \\ - \log\det(\Upsilon_s) - \log\det(\mathbf{K}_s). \tag{5}$$

Here Υ, respectively Υ_s, is the inverse of the posterior covariance matrix for the parameter vector β, given the all the data, respectively the sampled data. Explicitly,

$$\Upsilon = \mathbf{K}_0^{-1} + \mathbf{F}\mathbf{K}^{-1}\mathbf{F}^T \text{ and } \Upsilon_s = \mathbf{K}_0^{-1} + \mathbf{F}_s\mathbf{K}_s^{-1}\mathbf{F}_s^T.$$

The first multiplicand in the posterior predictive variance, $\varphi(\underline{y}_s, \underline{x}_s) \in \mathbb{R}$, informs the posterior distribution of σ^2. The second, $\phi(\underline{x}) \in \mathbb{R}^{n \times n}$, is the covariance matrix of the posterior predictive distribution, conditional on σ^2.

Consider now the situation at timestep k, in which n samples have already been taken at each of $k-1$ previous timesteps, and we wish to choose the best locations for the next set of measurements. In the notation of Section II-B, the number of *unsampled* measurements is n, while the number of *sampled* measurements is $m = n(k-1)$. Identifying terms in (4), the posterior predictive entropy can be written, $\mathcal{E} = \frac{1}{2}\log\det(\phi(\underline{x})) + \mathcal{M}(\underline{y}_s, \underline{x}_s)$, where $\mathcal{M}(\underline{y}_s, \underline{x}_s)$ does not depend on the locations or values of the new samples. Given past measurements at locations $\underline{x}_s$, it is desirable to take the next measurements at locations $\underline{x}_u$ which maximize $\log\det(\phi(\underline{x}))$. This function, which we call the *conditional entropy*, is invariant under permutations of $\underline{x}_u$, so we are free to choose any ordering to facilitate computation.

Note from (5) that $\log\det(\phi(\underline{x}))$ does not depend on the values of the measurements, *only* on their locations, and that the last two terms do not depend on the new locations at all. Thus we are interested in maximizing

$$\tilde{\mathcal{E}} = \log\det(\Upsilon) + \log\det(\mathbf{K}), \tag{6}$$

over potential measurement sites. However, the full distributed computation of these terms or their gradients over the robotic sensor network is not straightforward. We show later that the term $\log\det(\Upsilon)$ can be handled using known distributed computation tools. To deal with the term $\log\det(\mathbf{K})$, we follow the route presented next.

B. Alternative criterion for adaptive design

In this section, we propose an alternative aggregate objective function to maximize the posterior predictive entropy at each timestep. Let $\mathcal{H}^{(k)} : \mathcal{D}^n \to \mathbb{R}$ be defined by

$$\mathcal{H}^{(k)}(P) = \log\det(\Upsilon) - \frac{1}{2}\operatorname{tr}\big((\mathbf{K} - \boldsymbol{I})^2\big), \tag{7}$$

where the matrices $\Upsilon = \Upsilon^{(k)}(P)$ and $\mathbf{K} = \mathbf{K}^{(k)}(P)$ are calculated using the spatial positions $P \in \mathcal{D}^n$ at time $k+1$ for unsampled locations. We avoid the explicit functional notation for ease of exposition.

Proposition IV.2 ($\mathcal{H}^{(k)}(P)$ is a second order approximation of $\tilde{\mathcal{E}}$) *The function $\mathcal{H}^{(k)}$ is a second order approximation of $\tilde{\mathcal{E}}$ over the region $\mathcal{T}^{(k)} = \{P \in \mathcal{D}^n \mid \|\mathbf{K}\|_F < 2\}$ in*

the sense that $-\frac{1}{2}tr\big((\mathbf{K}-\boldsymbol{I})^2\big)$ *is the second order Taylor approximation of* $\log\det(\mathbf{K})$.

Throughout the sequel, we assume $P \in \mathcal{T}^{(k)}$. As informal justification, we note that it can be shown that under the $\frac{(n*(k-1))^2}{2}$-dimensional gradient flow of the map $A \mapsto \log\det(A)$ on correlation matrices, the level sets of the Frobenius norm are positively invariant.

C. Smoothness properties of $\mathcal{H}^{(k)}$

Here we study the smoothness properties of $\mathcal{H}^{(k)}$ and provide an expression for its gradient. Let $p_{i:l}$, $l \in \{1,\dots,d\}$ denote the lth coordinate of p_i. For notational simplicity, let $\nabla_{i:l}$ denote the partial derivative operator with respect to $p_{i:l}$, i.e., $\nabla_{i:l} = \frac{\partial}{\partial p_{i:l}}$, and let ∇_i denote the gradient operator with respect to p_i, i.e., $\nabla_i = (\nabla_{i:1},\dots,\nabla_{i:d})^T$. The following result establishes the smoothness of $\mathcal{H}^{(k)}$.

Proposition IV.3 *Assume that $f_1,\dots,f_p$ and the covariance of Z are C^1 with respect to the spatial positions of their arguments. Then $\mathcal{H}^{(k)}$ is C^1 on $\Omega^{(k)}$. Furthermore, the gradient, $\nabla\mathcal{H}^{(k)}$ at P may be written as the nd-dimensional vector, $\nabla\mathcal{H}^{(k)}|_P = \left((\nabla_1\mathcal{H}^{(k)}(P))^T,\dots,(\nabla_n\mathcal{H}^{(k)}(P))^T\right)^T$, where the partial derivatives take the form,*

$$\begin{aligned}\nabla_{i:l}\mathcal{H}^{(k)}(P) &= tr\left(\Upsilon^{-1}\nabla_{i:l}\Upsilon\right) - \\ &\quad - \frac{1}{2}\mathrm{row}_i\left(\mathbf{K}-\boldsymbol{I}\right)\mathrm{col}_i\left(\nabla_{i:l}\mathbf{K}\right), \text{ where}\\ \nabla_{i:l}\Upsilon &= \mathbf{F}\mathbf{K}^{-1}\nabla_{i:l}\mathbf{F}^T + \left(\mathbf{F}\mathbf{K}^{-1}\nabla_{i:l}\mathbf{F}^T\right)^T - \\ &\quad - \mathbf{F}\mathbf{K}^{-1}\left(\nabla_{i:l}\mathbf{K}\right)\mathbf{K}^{-1}\mathbf{F}^T.\end{aligned}$$

Here the matrix partials are taken component-wise.

Lemma IV.4 *Under the assumptions of Proposition IV.3, assume, in addition, that the partial derivatives of $f_1,\dots,f_p$ and the covariance of Z are C^1 with respect to the spatial positions of their arguments. Then the map $P \to \nabla\mathcal{H}^{(k)}|_P$ is globally Lipschitz on $\Omega^{(k)}$.*

V. Adaptive sampling via distributed entropy optimization

The function $\mathcal{H}^{(k)}$ depends on all of its arguments as well as all of the past measurement locations ($\underline{x}_s$) in a nontrivial and nonlinear way. In this section, we show how both $\mathcal{H}^{(k)}$ and $\nabla\mathcal{H}^{(k)}$ may be calculated in a distributed way over $\mathcal{N}$. This allows us to propose a distributed projected gradient descent algorithm which ensures that measurements are taken at local minima of $\mathcal{H}^{(k)}$ over $\Omega^{(k)}$.

A. Distributed calculations

Here, we describe a distributed method for calculating $\mathcal{H}^{(k)}$ and its gradient. In general, the matrices involved in the calculation depend on samples and locations known to multiple nodes. Furthermore, multiple samples and locations are known to each node. Distributed consensus algorithms may be performed in a similar manner whether each node knows one element or multiple elements, as long as the network is connected and each element is known by exactly one node. Since $\mathcal{V}(Q)$ describes a partition of the physical space, we may partition all measurement locations by region. Thus for each $(s,t) \in i_{\mathbb{F}}(\underline{x}_s)$, there is exactly one $j \in \{1,\dots,m\}$ such that $s \in V_j(Q)$. Let $R_{\text{in}}^{(1:k)} : \mathbb{N} \to \mathbb{F}(\mathbb{N})$ and $R_{\text{in}}^{(k+1)} : \mathbb{N}\times\mathcal{D}^n \to \mathbb{F}(\mathbb{N})$ be defined as follows,

$$\begin{aligned}R_{\text{in}}^{(1:k)}(j) &= \{i \in \{1,\dots,nk\} \mid x_{s:i} = (s,t),\ s \in V_j(Q)\}\\ R_{\text{in}}^{(k+1)}(j,P) &= \{i+nk \mid i \in \{1,\dots,n\} \text{ and } p_i \in V_j(Q)\}.\end{aligned}$$

These index sets list columns of the matrices $\mathbf{K}$ and $\mathbf{F}$ which correspond to past ($R_{\text{in}}^{(1:k)}$) and hypothetical future ($R_{\text{in}}^{(k+1)}$) sample locations in the jth Voronoi cell. With a slight abuse of notation, define $R_{\text{in}}^{(1:k+1)} : \mathbb{N}\times\mathcal{D}^n \to \mathbb{F}(\mathbb{N})$ as the union of the two sets, $R_{\text{in}}^{(1:k+1)}(j,P) = R_{\text{in}}^{(1:k)}(j) \cup R_{\text{in}}^{(k+1)}(j,P)$. The following result shows how pieces of $\mathcal{H}^{(k)}$ can be calculated.

Lemma V.1 *Let $P \in \mathcal{D}^n$ be a potential set of sites for the next measurement. Assume that S_j for each $j \in \{1,\dots,m\}$ knows the following quantities,*

- $\{x_{s:i} = (s,t) \in i_{\mathbb{F}}(\underline{x}_s) \mid \mathrm{d}(s, V_j(Q)) < r\}$
- $\{p_i \in i_{\mathbb{F}}(P) \mid \mathrm{d}(p_i, V_j(Q)) < r\}$*;*
- $\mathbf{K}_0 \in \mathbb{R}^{p\times p}$.

Using consensus and distributed JOR [23] algorithms, the network can calculate the matrices $\mathbf{F}\mathbf{K}^{-1}$ and Υ. After running the algorithms, S_j has access to the quantities, Υ, and $\mathrm{col}_i\left(\mathbf{F}\mathbf{K}^{-1}\right) \in \mathbb{R}^p,\ i \in R_{in}^{(1:k+1)}(j,P)$.

Next we present our main distributed computation result.

Proposition V.2 *For any $j_1 \neq j_2 \in \{1,\dots,m\}$, and any $i_1 \in R_{in}^{(1:k+1)}(j_1,P)$ and $i_2 \in R_{in}^{(1:k+1)}(j_2,P)$, assume that if $[\mathbf{K}]_{i_1 i_2} \neq 0$ then S_{j_1} can communicate with S_{j_2}. Then, under the assumptions of Lemma V.1, $\mathcal{H}^{(k)}$ and its gradient at $P \in \mathcal{D}^n$ can be calculated in a distributed manner by $\mathcal{N}$.*

B. Distributed gradient descent algorithm

Here we outline a distributed version of the projected gradient descent algorithm (see, e.g. [24]), which is guaranteed to converge to a stationary point of $\mathcal{H}^{(k)}$ on $\Omega^{(k)}$. Let $\kappa_j^{(k)} : \mathcal{D}^n \to \mathbb{R}$ denote the partial sum,

$$\kappa_j^{(k)}(P) = \sum_{i \in R_{\text{in}}^{(1:k+1)}(j,P)} \mathrm{row}_i(\mathbf{K}^{(k)}(P) - \boldsymbol{I})\mathrm{col}_i(\mathbf{K}^{(k)}(P) - \boldsymbol{I}).$$

Then $\kappa_j^{(k)}(P)$ may be calculated by S_j, and $\mathrm{tr}\left(\left(\mathbf{K}^{(k)}(P) - \boldsymbol{I}\right)^2\right) = \sum_{j=1}^m \kappa_j^{(k)}(P)$. Table I describes a distributed line search with a starting position of $P \in \Omega$. The maximum stepsize, $\alpha_{\max}$, ensures that all robots with nonzero partial derivatives can move the maximum distance.

We are ready to present our technique for a greedy optimization algorithm. At timestep k, the nodes follow a gradient descent algorithm to define a sequence of configurations, $\{P_l^\dagger\}$, $l \in \mathbb{N}$, such that $P_1^\dagger$ is $P^{(k)} \in \mathcal{D}^n$, the vector of current spatial locations of the robotic agents and

$$P_{l+1}^\dagger = \mathrm{proj}_\Omega\left(P_l^\dagger - \alpha\nabla\mathcal{H}^{(k)}|_{P_l^\dagger}\right),\ \alpha \in \mathbb{R}_{\geq 0},$$

where α is chosen via Distributed Line Search Algorithm. When $|\mathcal{H}^{(k)}(P_{l+1}^\dagger) - \mathcal{H}^{(k)}(P_l^\dagger)| = 0$, the algorithm

Name: DISTRIBUTED LINE SEARCH ALGORITHM
Goal: Compute step size for gradient descent of $\mathcal{H}^{(k)}$
Input: Configuration, $P = (p_1, \ldots, p_n) \in \mathcal{D}^n$
Assumes: (i) Connected network of static nodes
(ii) S_j knows $\mathcal{H}^{(k)}(P)$, as well as p_i, $\nabla_i \mathcal{H}^{(k)}(P)$, $\text{row}_i(\mathbf{K}-\boldsymbol{I})$ and $\Omega_i^{(k)}$ for each robot within communication range
(iii) Shrinkage factor τ, tolerance $\theta \in (0,1)$, and prior β-correlation matrix, $\mathbf{K}_0$ known a priori
Uses: (i) $p_i'(\alpha, P) = \text{proj}_{\Omega_i^{(k)}}(p_i + \alpha \nabla_i \mathcal{H}^{(k)}(P))$
(ii) Square distance of robots entering $V_j(Q)$,

$$\mathrm{d}_j(\alpha, P) = \sum_{\substack{i \in \{1,\ldots,n\} \text{ such that} \\ p_i'(\alpha,P) \in V_j(Q)}} \|p_i'(\alpha, P) - p_i\|^2$$

Output: Step size $\alpha \in \mathbb{R}$, next configuration $P'(\alpha, P) = (p_1'(\alpha, P), \ldots, p_n'(\alpha, P))^T$, and $\mathcal{H}^{(k)}(P'(\alpha, P))$.

Initialization
1: $S_1, \ldots, S_m$ use consensus to calculate the quantity

$$\alpha_{\max} = \max\left\{\|\nabla_i \mathcal{H}^{(k)}(P)\|^{-1} \middle| \nabla_i \mathcal{H}^{(k)}(P) \neq 0\right\} u_{\max}$$

For $j \in \{1, \ldots, m\}$, node S_j executes concurrently
1: $\alpha = \alpha_{\max}$
2: **repeat**
3: calculates $\mathrm{d}_j(\alpha, P)^2$
4: calculates Υ according to Lemma V.1
5: calculates $\kappa_j^{(k)}(P'(\alpha, P))$
6: executes consensus algorithm to calculate the following:

$$\text{tr}\left((\mathbf{K} - \boldsymbol{I})^2\right) = \sum_{j=1}^{m} \kappa_j^{(k)}(P'(\alpha, P))$$

$$\left\|P - P'(\alpha, P)\right\|^2 = \sum_{j=1}^{m} \mathrm{d}_j(\alpha, P)^2$$

7: $\mathcal{H}^{(k)}(P'(\alpha, P)) = \log\det(\Upsilon) + \text{tr}\left((\mathbf{K} - \boldsymbol{I})^2\right)$
8: $\nu = \frac{\theta}{\alpha}\|P - P'(\alpha, P)\|^2 - \mathcal{H}^{(k)}(P'(\alpha, P)) + \mathcal{H}^{(k)}(P)$
9: **if** $\nu > 0$ **then**
10: $\alpha = \alpha\tau$
11: **until** $\nu \leq 0$

TABLE I
DISTRIBUTED LINE SEARCH ALGORITHM.

terminates, and the nodes set $P^{(k+1)} = P_{l+1}^{\dagger}$. By the end of this calculation, each node knows the identity of robotic agents in its Voronoi cell at timestep $k+1$. Node S_j transmits $p_i(k+1)$ to robot R_i, which then moves to that location between timesteps. The overall algorithm is in Table II.

Proposition V.3 *The* DISTRIBUTED PROJECTED GRADIENT DESCENT ALGORITHM *is distributed over the network $\mathcal{N}$. Moreover, under the assumptions of Lemma IV.4, any execution is such that the robots do not collide and, at each timestep after the first, measurements are taken at stationary configurations of $P \mapsto \mathcal{H}^{(k)}(P)$ over $\Omega^{(k)}$.*

Name: DISTRIBUTED PROJECTED GRADIENT DESCENT ALGORITHM
Goal: Find a local minimum of $\mathcal{H}^{(k)}$ within $\Omega^{(k)}$.
Assumes: (i) Connected network of nodes and robots
(ii) Static nodes deployed over $\mathcal{D}$ such that $R \geq \max_{i \in \{1,\ldots,m\}} \{\text{CR}(V_i(Q))\} + u_{\max}$, initial configuration $P^{(1)} \in \mathcal{D}^n$
(iii) Line search shrinkage factor τ, tolerance $\theta \in (0,1)$, and prior β-correlation matrix, $\mathbf{K}_0$ known a priori by all nodes.

At time $k \in \mathbb{Z}_{\geq 0}$, robot R_i executes:
1: takes measurement at $p_i(k)$
2: sends position to S_j, where $p_i(k) \in V_j(Q)$
3: receives next location $p_i(k+1)$
4: moves to $p_i(k+1)$.

At time $k \in \mathbb{Z}_{\geq 0}$, node S_j executes:
1: collects location from each R_i with $\mathrm{d}(p_i(k), V_j(Q)) < u_{\max}$ as well as locations of nearby agents
2: updates $R_{\text{in}}^{(k+1)}(j, P)$ and $R_{\text{in}}^{(1:k+1)}(j)$
3: calculates Υ (cf. Lemma V.1)
4: computes $\kappa_j^{(k)}\left(P^{(k)}\right)$, and then $\mathcal{H}^{(k)}\left(P^{(k)}\right)$ via consensus
5: sets $P_{\text{next}} = P^{(k)}$
6: **repeat**
7: stores $P_{\text{cur}} = P_{\text{next}}$ and $\mathcal{H}^{(k)}(P_{\text{cur}}) = \mathcal{H}^{(k)}(P_{\text{next}})$
8: calculates $-\nabla_i \mathcal{H}^{(k)}(P_{\text{cur}})$ for each $i \in R_{\text{in}}^{(k+1)}(j, P_{\text{cur}})$ (cf. Prop. V.2)
9: runs DISTRIBUTED LINE SEARCH ALGORITHM at P_{cur} to get α, P_{next}, and $\mathcal{H}^{(k)}(P_{\text{next}})$
10: **until** $|\mathcal{H}^{(k)}(P_{\text{next}}) - \mathcal{H}^{(k)}(P_{\text{cur}})| = 0$
11: sets $P^{(k+1)} = P_{\text{next}}$
12: conveys $p_i(k+1)$ to R_i for each $i \in R_{\text{in}}^{(k+1)}(j, P_{\text{cur}})$

TABLE II
DISTRIBUTED PROJECTED GRADIENT DESCENT ALGORITHM.

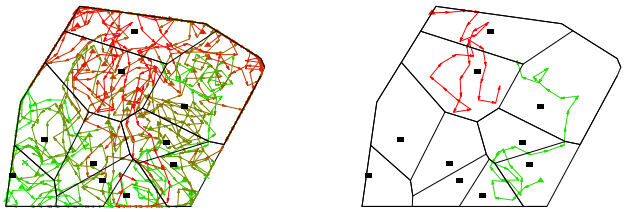
C. Simulations

We implemented the DISTRIBUTED PROJECTED GRADIENT DESCENT ALGORITHM in several simulations. The one presented here was run with $d = 2$ spatial dimensions, $m = 10$ static nodes, $n = 30$ robotic agents, and the domain $\mathcal{D} = \{(0, .1), (2.5, .1), (3.45, 1.6), (3.5, 1.7), (3.45, 1.8), (2.7, 2.2), (1, 2.4), (0.2, 1.3)\}$. We used the separable covariance function defined by $\text{Cov}[Z(s_1, t_1), Z(s_2, t_2)] = C_{\text{tap}}(\|s_1 - s_2\|, 0.5) C_{\text{tap}}(|t_1 - t_2|, 6.5)$, where

$$C_{\text{tap}}(\delta, r) = \begin{cases} e^{-\frac{\delta}{10r}}\left(1 - \frac{3\delta}{2r} + \frac{\delta^3}{2r^3}\right) \text{ if } \delta \leq r, \\ 0 \text{ otherwise.} \end{cases}$$

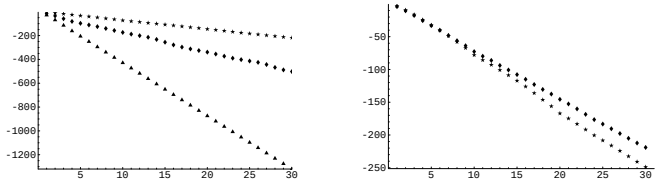
This is a tapered exponential function belonging to the class of covariance functions suggested in [25]. We used $\omega = 0.02$, and $u_{\max} = 0.2$. For the mean regression functions f_i, we used $f((x, y), t) = (1, \sin(2\pi x), \sin(2\pi y))^T$.

Fig. 1 shows the trajectories taken by the robots. We compare in Fig. 2 the performance of our algorithm against two algorithms that pre-plan agent trajectories. The first is a static approach in which the agents spread out around the region and remain in place. The second is a lawnmower-type algorithm in which the agents march back and forth across the region in evenly space (horizontal) lines. In all cases, two agents lost contact part way through. Note that both dynamic algorithms perform much better than the static one, but the gradient descent algorithm performs better than



(a) (b)

Fig. 1. (a) Trajectories of all robots, and (b) two representative trajectories, both from the same run of the distributed projected gradient descent algorithm. The filled squares represent the (static) positions of the nodes, and the filled triangles show the starting positions of the robots. Two agents were lost in progress, their final position designated by X's in plot (a).



(a) (b)

Fig. 2. Plot (a) shows the progression of $\mathcal{H}^{(k)}$ as k increases, resulting from the static (triangle), lawnmower (diamond), and gradient descent (star) approaches. For the gradient descent algorithm only, plot (b) compares the value of $\tilde{\mathcal{E}}$ (stars) against the approximation, $\mathcal{H}^{(k)}$ (diamonds).

the lawnmower. This is due to the facts that the gradient algorithm reacts to the basis functions of the model and that the lawnmower does not compensate for the dropped agents.

VI. Conclusions and Future Work

We have designed a distributed algorithm for adaptive sampling of spatiotemporal processes with unknown mean and covariance known up to a scaling parameter. At each time step, an heterogeneous network composed of static nodes and mobile agents optimizes an aggregate objective function to maximize the information provided by future data. We have shown that the objective function is a second-order approximation of the conditional entropy, defined as the posterior predictive entropy conditional on the covariance scaling parameter. We have characterized the correctness of the proposed coordination algorithm and provided several simulations of its performance. Immediate future work will investigate the invariance of the region $\mathcal{T}^{(k)}$ under the gradient of $\mathcal{H}^{(k)}$, comparison against a smarter, self-adjusting lawnmower algorithm, and quantification of the communication and computational complexity of the algorithms. From the preliminary experiments reported here, we found that the JOR algorithm seems to account for the vast majority of the communication complexity. Ongoing work will explore the possibility of reducing the overhead of this special case of the JOR algorithm. In the longer term, we plan to continue exploring methods to cooperatively estimate stochastic processes considering statistical models with increasing generality.

Acknowledgements

This research was partially supported by NSF CAREER Award ECS-0546871. We would like to thank Professors B. Sansó and H. Lee for advice on Bayesian spatial statistics.

References

[1] P. K. Kitanidis, "Parameter uncertainty in estimation of spatial functions: Bayesian analysis," *Water Resources Research*, vol. 22, pp. 449–507, 1986.

[2] M. Gaudard, M. Karvson, E. Linder, and D. Sinha, "Bayesian spatial prediction," *Environmental and Ecological Statistics*, vol. 6, pp. 147–171, 1999.

[3] N. D. Lee and J. V. Zidek, *Statistical Analysis of Environmental Space-Time Processes*. Springer Series in Statistics, New York: Springer, 2006.

[4] F. Pukelsheim, *Optimal Design of Experiments*, vol. 50 of *Classics in Applied Mathematics*. Philadelphia, PA: SIAM, 2006.

[5] E. P. Liski, N. K. Mandal, K. R. Shah, and B. K. Sinha, *Topics in Optimal Design*, vol. 163 of *Lecture Notes in Statistics*. New York: Springer, 2002.

[6] D. A. Cohn, "Neural network exploration using optimal experiment design," in *Neural Networks*, pp. 679–686, Morgan Kaufmann, 1994.

[7] R. B. Gramacy, *Bayesian Treed Gaussian Process Models*. PhD thesis, University of California, Santa Cruz, CA 95064, December 2005. Department of Applied Math & Statistics.

[8] D. J. C. MacKay, "Information-based objective functions for active data selection," *Neural Computation*, vol. 4, no. 4, pp. 590–604, 1992.

[9] K. Chaloner and I. Verdinelli, "Bayesian experimental design, a review," *Statistical Science*, vol. 10, no. 3, pp. 273–304, 1995.

[10] S. Seo, M. Wallat, T. Graepel, and K. Obermayer, "Gaussian process regression: Active data selection and test point rejection," in *In Proceedings of the International Joint Conference on Neural Networks (IJCNN*, pp. 241–246, IEEE, 2000.

[11] J. S. Willcox, J. G. Bellingham, Y. Zhang, and A. B. Baggeroer, "Performance metrics for oceanographic surveys with autonomous underwater vehicles," *IEEE Journal of Oceanic Engineering*, vol. 26, no. 4, pp. 711–725, 2001.

[12] D. O. Popa, K. Sreenath, and F. L. Lewis, "Robotic deployment for environmental sampling applications," in *International Conference on Control and Automation*, (Budapest, Hungary), pp. 197–202, June 2005.

[13] M. F. Mysorewala, *Simultaneous robot localization and mapping of parameterized spatio-temporal fields using multi-scale adaptive sampling*. PhD thesis, University of Texas at Arlington, 2008.

[14] N. E. Leonard, D. Paley, F. Lekien, R. Sepulchre, D. M. Fratantoni, and R. Davis, "Collective motion, sensor networks and ocean sampling," *Proceedings of the IEEE*, vol. 95, no. 1, pp. 48–74, 2007.

[15] K. M. Lynch, I. B. Schwartz, P. Yang, and R. A. Freeman, "Decentralized environmental modeling by mobile sensor networks," *IEEE Transactions on Robotics*, vol. 24, no. 3, pp. 710–724, 2008.

[16] S. Meyn, *Control Techniques for Complex Networks*. Cambridge University Press, 2007.

[17] F. Zhang, E. Fiorelli, and N. E. Leonard, "Exploring scalar fields using multiple sensor platforms: tracking level curves," in *IEEE Conf. on Decision and Control*, (New Orleans, LA), pp. 3579–3584, Dec. 2007.

[18] R. Graham and J. Cortés, "A cooperative deployment strategy for optimal sampling in spatiotemporal estimation," in *IEEE Conf. on Decision and Control*, (Cancun, Mexico), pp. 2432–2437, Dec. 2008.

[19] R. Graham and J. Cortés, "Cooperative adaptive sampling of random fields with unknown covariance," in *American Control Conference*, (St. Louis, MI), 2009. 4543-4548.

[20] C. P. Robert and G. Casella, *Monte Carlo statistical methods*. Springer texts in statistics, New York: Springer, 2004.

[21] D. S. Bernstein, *Matrix Mathematics*. Princeton, NJ: Princeton University Press, 2005.

[22] T. Leonard and J. S. J. Hsu, *Bayesian Methods*, vol. 1 of *Cambridge series in statistical and probabilistic mathematics*. Cambridge, UK: Cambridge University Press, 1999.

[23] D. P. Bertsekas and J. N. Tsitsiklis, *Parallel and Distributed Computation: Numerical Methods*. Athena Scientific, 1997.

[24] D. P. Bertsekas, "On the Goldstein-Levitin-Polyak gradient projection method," *IEEE Transactions on Automatic Control*, vol. 21, no. 2, pp. 174–184, 1976.

[25] R. Furrer, M. G. Genton, and D. Nychka, "Covariance tapering for interpolation of large spatial datasets," *Journal of Computational and Graphical Statistics*, vol. 15, no. 3, pp. 502–523, 2006.