Cooperative adaptive sampling of random fields with unknown covariance

Rishi Graham Jorge Cortés

Abstract— **This paper considers robotic sensor networks performing spatial estimation tasks. We model a physical process of interest as a spatiotemporal random field with mean unknown and covariance known up to a scaling parameter. We design a distributed coordination algorithm for an heterogeneous network composed of mobile agents that take point measurements of the field and static nodes that fuse the information received from the agents and compute directions of maximum descent of the estimation uncertainty. The technical approach builds on a novel iterative reformulation of Bayesian sequential field estimation, and combines tools from distributed linear iterations, nonlinear programming, and spatial statistics.**

I. INTRODUCTION

Networks of environmental sensors are playing an increasingly important role in scientific studies of the ocean, rivers, and the atmosphere. Envisioned tasks include pollutant detection, fire monitoring, and mapping of ocean currents. Mobile sensing robots can improve the efficiency of data collection, adapt to changes in the environment, and provide a robust response to sensor failures. Complex statistical techniques come into play in the analysis of environmental processes. Consequently, the operation of robotic sensors must be driven by statistically-aware algorithms that make the most of the network capabilities for data collection and fusion. At the same time, such algorithms need to be distributed and scalable to make robotic networks capable of operating in an autonomous and robust fashion. The combination of these two objectives, complex statistical modeling and distributed coordination, presents grand technical challenges: traditional statistical modeling and inference assume full availability of all measurements and central computation. While the availability of data at a central location is certainly a desirable property, the paradigm for motion coordination builds on partial, fragmented information. This work is a step forward in bridging the gap between sophisticated statistical modeling and distributed motion coordination.

Literature review: Complex statistical techniques exist to model the evolution of physical phenomena. Of particular relevance to this work are [1], [2], regarding statistical models, and [3], [4], regarding the application of optimal design techniques to Bayesian models. Under certain conditions on the covariance structure, data taken far from the prediction site have very little impact on the predictor [5]. When the random field does not have a covariance structure with finite spatial correlation, an approximation which does may be generated via covariance tapering [6]. The optimal design literature [7], [8] deals with the problem of determining locations where data should be taken to optimize the estimation.

In the cooperative control arena, various works consider mobile sensor networks performing spatial estimation tasks. [9] introduces performance metrics for oceanographic surveys by autonomous underwater vehicles. [10] considers a robotic sensor network with centralized control estimating a static field from measurements with both sensing and localization error. [11] considers optimal sampling trajectories from a parameterized set of paths. In [12], [13] the focus is on estimating deterministic fields when the measurements taken by individual robots are uncorrelated. [14] discusses the tracking of level curves in a noisy scalar field. In previous work [15], we have considered the estimation of random fields with known covariance. The subject of this paper is the additional complexity in the design of distributed strategies caused by unknown parameters in the field covariance.

Statement of contributions: We take as starting point a widely accepted Bayesian model for the prediction of a spatiotemporal random field with mean unknown and covariance known up to a scaling parameter. The predictive variance of this model can be written as a scaled multiplication of two components, one corresponding to uncertainty about the covariance of the field, the other corresponding to uncertainty of the prediction conditional on the covariance. Our first contribution is the development of a novel iterative procedure amenable to distributed calculation for updating the first component sequentially as new measurements arrive. We also introduce an upper bound for the second component which can be calculated in a distributed way. These two results allow us to identify an objective function for gathering data which minimizes uncertainty in the resulting estimation. Our second contribution is the characterization of the smoothness properties of the objective function and the computation of its gradient. Using consensus and distributed Jacobi overrelaxation algorithms, we show how the objective function and its gradient can be computed in a distributed way across a network composed of robotic agents and static nodes. Our third contribution is the design of a coordination algorithm based on projected gradient descent which guarantees one-step-ahead local optimality of data collection.

Organization: Section II introduces basic notation and describes the statistical model. Section III states the robotic network model and the overall network objective. Section IV presents the results on the objective function, with attention to its smoothness properties. Section V discusses how the network can make the required calculations in a distributed way. Section VI presents the cooperative strategy for optimal data collection. Section VII contains our conclusions.

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

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 $\underline{u} = (u_1, \dots, u_a)^T$, $a \in \mathbb{Z}_{>0}$, and $\underline{v} = (v_1, \dots, v_b)^T$, $b \in \mathbb{Z}_{>0}$, we denote by $(\underline{u}, \underline{v})$ its concatenation $(\underline{u}, \underline{v}) = (u_1, \dots, u_a, v_1, \dots, v_b)^T$. We denote by ∂S the boundary of a set S. The *ϵ-contraction* of a set S, with $\epsilon > 0$, is the set $S_\epsilon = \{q \in S \mid \mathrm{d}(q, \partial S) \geq \epsilon\}$. 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, that is, the radius of the smallest-radius d-sphere enclosing S. We denote by $\mathbb{F}(S)$ the collection of finite subsets of S.

We consider 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* $\mathcal{V}(\underline{s}) = (V_1(\underline{s}), \dots, V_n(\underline{s}))$ of $\mathcal{D}$ generated by the points $\underline{s} = (s_1, \dots, s_n)$ 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.

A. Bayesian modeling of space-time processes

Let Z denote a random space-time process taking values on $\mathcal{D}_e$. Let $\underline{y} = (y_1, \dots, y_m)^T \in \mathbb{R}^m$ be $m \in \mathbb{N}$ measurements taken from Z at corresponding locations $\underline{x} = (x_1, \dots, x_m)^T \in \mathcal{D}_e^m$, with $x_i = (s_i, t_i)$, $i \in \{1, \dots, m\}$. Given these data, various models allow for prediction of the Z at any point in $\mathcal{D}_e$, with associated uncertainty. Optimal design is the process of choosing locations to take measurements in order to reduce the uncertainty of the resulting statistical prediction. Since prediction 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 [16]. If the field is modeled as a Gaussian process with known covariance, the posterior predictive mean corresponds to the *Best Linear Unbiased Predictor*, or *Linear Unbiased Minimum Variance Estimator*, 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 [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 measurements are distributed as

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

Here $\beta \in \mathbb{R}^p$ is a vector of unknown regression parameters, $\sigma^2 \in \mathbb{R}_{>0}$ is the unknown variance parameter, and $\mathbf{K}$ is a correlation matrix whose i,jth element is the correlation between y_i and y_j. We assume a finite correlation range in space, $r \in \mathbb{R}$, such that if $\|s_i - s_j\| \geq r$, then $\mathbf{K}_{ij} = \mathbf{K}_{ji} = 0$. The matrix $\mathbf{F}$ is determined by a set of $p \in \mathbb{N}$ known basis functions $f_i : \mathcal{D}_e \to \mathbb{R}$ evaluated at the locations $\underline{x}$, i.e.,

$$\mathbf{F} = \begin{bmatrix} f_1(x_1) & \dots & f_1(x_m) \\ \vdots & \ddots & \vdots \\ f_p(x_1) & \dots & f_p(x_m) \end{bmatrix}.$$

In order to ensure an analytical form for the posterior predictive distribution, we assume the following conjugate prior distributions on the parameters,

$$\beta | \sigma^2 \sim \mathrm{N}_p\left(\beta_0, \sigma^2 \mathbf{K}_0\right) \tag{2a}$$

$$\sigma^2 \sim \Gamma^{-1}\left(\frac{\nu}{2}, \frac{q\nu}{2}\right). \tag{2b}$$

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, and $\Gamma^{-1}(a,b)$ denotes the inverse gamma distribution with shape parameter a and scale parameter b (see, e.g. [17]).

Proposition II.1 (Posterior predictive distribution [2]) *Under the Bayesian model* (1)*, the posterior predictive at location $x_0 \in \mathcal{D}_e$ is a shifted Students t distribution (see, e.g. [17]) with $\gamma = \nu + m + 1$ degrees of freedom, which takes the form, for $Z = Z(x_0)$,*

$$Z | \underline{y}, \underline{x} \propto \left|\mathrm{Var}[Z|\underline{y}, \underline{x}]\right| \left(1 + \frac{\left(Z - \mathrm{E}[Z|\underline{y}, \underline{x}]\right)^2}{\frac{\gamma - 1}{\gamma} \mathrm{Var}[Z|\underline{y}, \underline{x}]}\right)^{-\frac{\gamma+1}{2}}.$$

Here the expectation is given by

$$\mathrm{E}[Z|\underline{y}, \underline{x}] = \left(f(x_0) - \mathbf{F}\mathbf{K}^{-1}\mathbf{k}\right)^T \tilde{\beta} + \mathbf{k}^T \mathbf{K}^{-1} \underline{y}$$
$$\tilde{\beta} = ((E + \mathbf{K}_0^{-1})^{-1} E)\hat{\beta} + $$
$$+ (I - (E + \mathbf{K}_0^{-1})^{-1} E)\beta_0,$$

where $\hat{\beta} = E^{-1}\mathbf{F}\mathbf{K}^{-1}\underline{y}$, $E = \mathbf{F}\mathbf{K}^{-1}\mathbf{F}^T$, and $\mathbf{k} =$ $\mathrm{Cor}[\underline{y}, Z] \in \mathbb{R}^m$. The variance is given by

$$\mathrm{Var}[Z|\underline{y}, \underline{x}] = \frac{\varphi(\underline{y}, \underline{x})}{\gamma} \phi(x_0; \underline{x})$$
$$\phi(x_0; \underline{x}) = \mathrm{Cor}[Z, Z] - \mathbf{k}^T \mathbf{K}^{-1} \mathbf{k} + \xi_0^T \left(\mathbf{K}_0^{-1} + E\right)^{-1} \xi_0$$
$$\xi_0 = f(x_0) - \mathbf{F}\mathbf{K}^{-1}\mathbf{k}$$
$$\varphi(\underline{y}, \underline{x}) = q\nu + \frac{1}{2}\left(\underline{y} - \mathbf{F}^T \hat{\beta}\right)^T \mathbf{K}^{-1} \left(\underline{y} - \mathbf{F}^T \hat{\beta}\right) +$$
$$+ \frac{1}{2}\left(\hat{\beta} - \beta_0\right)^T \left(\mathbf{K}_0 + E^{-1}\right)^{-1} \left(\hat{\beta} - \beta_0\right).$$

The posterior predictive variance in Proposition II.1 can be separated into three multiplicands. The first, $\frac{1}{\gamma}$, decreases linearly with m. The second, $\varphi(\underline{y}, \underline{x})$, results from the uncertainty in σ^2. The third, $\phi(x_0; \underline{x})$, is the posterior predictive variance conditional on σ^2.

B. Distributed computational tools

Here we briefly describe some tools for distributed computations. Consider a network of n agents with limited communication capabilities. Assume that the communication graph G is connected (i.e., there is a path connecting any two agents). Let $a_{ij} \in \{0, 1\}$, $i, j \in \{1, \dots, n\}$ be 1 if agents i and j are neighbors in G, 0 otherwise. Let $b = (b_1, \dots, b_n)^T \in \mathbb{R}^n$, $C = [c_{ij}] \in \mathbb{R}^{n \times n}$ and $D = [d_{ij}] \in \mathbb{R}^{n \times m}$, and assume agent i knows b_i and the ith rows of C and D. Additionally assume that $c_{ii} \neq 0$ and for $i \neq j$, $c_{ij} \neq 0$ iff i and j are communication neighbors. Then the following results hold.

JOR: The network can compute the vector $y = C^{-1}b$ via a *distributed Jacobi overrelaxation* algorithm [18], formulated as the discrete-time dynamical system,

$$y_i(l+1) = (1-h)y_i(l) - \frac{h}{c_{ii}}\left(\sum_{j\neq i} c_{ij}y_j(l) - b_i\right),$$

for $l \in \mathbb{Z}_{\geq 0}$ and $i \in \{1,\dots,n\}$, where $y(0) \in \mathbb{R}^n$ and $h \in (0,1)$. At the end of the algorithm, agent i knows the ith element of the result.

dynamic consensus: Consider a time-varying signal, $t \mapsto u(t) = (u_1(t),\dots,u_n(t))^T \in (\mathbb{R}^m)^n$ represented as an n-dimensional vector of m-dimensional vectors. Assume that agent i has access to $u_i(t)$. For each $i \in \{1,\dots,n\}$, consider the dynamical system,

$$\frac{dv_i}{dt} = \gamma\left(u_i(t) - v_i(t)\right) - \sum_{j\neq i} a_{ij}\left(v_i(t) - v_j(t)\right) + \sum_{j\neq i} a_{ij}\left(w_i(t) - w_j(t)\right),$$

$$\frac{dw_i}{dt} = -\sum_{j\neq i} a_{ij}\left(v_i(t) - v_j(t)\right),$$

where $\gamma \in \mathbb{R}_{>0}$, and $v, w \in (\mathbb{R}^m)^n$. For any constant input, $t \mapsto u(t) = u \in (\mathbb{R}^m)^n$, and initial states, $v(0), w(0) \in (\mathbb{R}^m)^n$, this algorithm satisfies,

$$\lim_{t\to\infty} v_i(t) - \frac{1}{n}\sum_{i=1}^{n} u_i(t) = 0,$$

exponentially fast, for $i \in \{1,\dots,n\}$. If u varies slowly, small steady-state errors are guaranteed [19].

WLS: The network can compute the quantity, $D^T C^{-1} b$ by concurrently following the JOR algorithm to calculate $y = C^{-1}b$ and the dynamic average consensus algorithm to track $D^T y$. At the end of this algorithm, known as *weighted least squares* [20], agent i knows the ith element of $C^{-1}b$ and the entire vector $D^T C^{-1} b$.

The results above are only exact asymptotically, but convergence is very fast. The JOR algorithm converges linearly with time, while the dynamic consensus algorithm converges exponentially. Next, we present another tool which we will find useful for multiplying matrices across the network.

Lemma II.2 (One-hop triple matrix multiplication) *Let C, D as described above, and assume that only the ith row and column of C are nonzero. Then agent i can compute the matrix $D^T C D$ with one-hop communication.*

III. Problem statement

Here we introduce the model for the group of robotic agents and static nodes, and detail the overall objective.

A. Robotic sensor network model

Consider a group $\{S_1,\dots,S_m\}$ of $m \in \mathbb{N}$ static nodes deployed in a convex polytope $\mathcal{D} \subset \mathbb{R}^d$. Let $Q = (q_1,\dots,q_m) \in \mathcal{D}^m$ 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 $\{R_1,\dots,R_n\}$ of n robotic sensor agents. The position of robot $i \in \{1,\dots,n\}$ 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 will need to be able to communicate with any robot which may be within covariance range of the points in its Voronoi region at the following timestep. To that end, we assume that

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

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 communicates the measurement and location to static nodes within communication range, along with the locations of any other sensed robots. The nodes then compute control vectors, and relay them back to robots within communication range. The implementation does not require direct communication between robots. 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 $P^{(k)} = (p_1(k),\dots,p_n(k))^T$. 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 [15]. 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. The average variance as objective function

For predictions over a region in space and time, the average variance is a natural measure of uncertainty, corresponding to A-optimality. We consider the average over the spatiotemporal region of the posterior predictive variance. Here we motivate a sequential formulation of this objective. Assume that the experiment has been run for $k_{\max} \in \mathbb{Z}_{>0}$ timesteps. A sequence of measurements taken at discrete times $\{1,\dots,k_{\max}\}$, at space-time locations $\underline{x} \in (\mathcal{D}_e^n)^{k_{\max}}$ is available. Given the distribution in Proposition II.1, the average posterior predictive variance over $\mathcal{D}$ and T is,

$$\mathcal{A} = \frac{1}{\gamma}\varphi(\underline{y}, \underline{x}) \int_{\mathcal{D}} \int_{T} \phi((y_0, t_0); \underline{x})\, dt_0\, dy_0. \tag{4}$$

One would like to choose the measurement locations that minimize $\mathcal{A}$. Since measurements are taken sequentially, and each new set is restricted to a region nearby the previous, one cannot simply optimize over $(\mathcal{D}_e^n)^k_{\max}$. Additionally, since the quantity $\varphi(\underline{y}, \underline{x})$ depends on the actual values of the measurements, it cannot be calculated a priori.

Consider, instead, a greedy approach in which we use past measurements to choose the positions for the next ones. At each timestep the next locations are chosen to minimize the average posterior variance of the predictor given the information known so far. This process is known as sequential optimal or adaptive design. In the next section, we develop a sequential formulation of the average posterior predictive variance and discuss its amenability to distributed implementation over the network $\mathcal{N}$.

IV. Distributed criterion for adaptive design

The objective of this section is to develop an optimality criterion to maximally reduce the average variance of the prediction made at each timestep. We begin by introducing a novel iterative formulation of the posterior predictive variance, which allows for estimation based on previous measurement values and update based on new ones. Given centralized computing capabilities, this equation can be used to perform sequential optimal design, but is not amenable to distributed computation. We therefore provide an upper bound whose computation is distributed over $\mathcal{N}$.

A. Sequential formulation of φ

At timestep k, assume that measurements, $\underline{y}_s \in \mathbb{R}^{nk}$ have already been taken at *sampled* locations $\underline{x}_s \in \mathcal{D}_e^{nk}$. We are interested in choosing *unsampled* locations, $\underline{x}_u \in \mathcal{D}_e^n$ at which to take the next set of measurements, $\underline{y}_u \in \mathbb{R}^n$. Let $\underline{x} = (\underline{x}_u^T, \underline{x}_s^T)^T \in \mathcal{D}_e^{n(k+1)}$ denote the full set of measurement locations at timestep $k+1$, and let $\underline{y} = (\underline{y}_u^T, \underline{y}_s^T)^T \in \mathbb{R}^{n(k+1)}$ denote the full set of measurements. Let $\mathbf{K}_s$ denote the correlation matrix of the vector $\underline{y}_s$, and let $\mathbf{K}_{us} = \mathbf{K}_{su}^T$ denote the matrix whose (i,j)th element is the correlation between $y_{u:i}$ and $y_{s:j}$. Once all measurements have been taken, the posterior predictive variance is

$$\mathrm{Var}[Z(x_0)|\underline{y}, \underline{x}] = \frac{\varphi(\underline{y}, \underline{x})}{\gamma} \phi(x_0; \underline{x}).$$

However, since the measurements $\underline{y}_u$ have not been taken yet at time step k, we can not calculate $\varphi(\underline{y}, \underline{x})$. Our approach is to use the generalized least squares estimate, and compute the induced errors in the approximation.

Proposition IV.1 *Let $\underline{\hat{y}}_{LS} = \mathbf{K}_{us}\mathbf{K}_s^{-1}\underline{y}_s$ be the generalized least squares estimate of $\underline{y}_u$ based on the sampled measurements (conditional on all parameters) and let $\overline{y}_{LS} = \underline{y}_u - \underline{\hat{y}}_{LS}$. Then, φ can be reformulated as*

$$\varphi(\underline{y}, \underline{x}) = \varphi(\underline{y}_s, \underline{x}_s) + \hat{\varphi}(\underline{y}_s, \underline{x}_s, \underline{x}_u) + \tilde{\varphi}(\underline{y}_s, \underline{x}_s, \overline{y}_{LS}, \underline{x}_u).$$

where $\hat{\varphi}(\underline{y}_s, \underline{x}_s, \underline{x}_u) = \hat{\varphi}$ and $\tilde{\varphi}(\underline{y}_s, \underline{x}_s, \overline{y}_{LS}, \underline{x}_u) = \tilde{\varphi}$ are

$$\begin{aligned}\hat{\varphi} = \frac{1}{2}\left(\beta^\dagger - \beta_0\right)^T \left(\mathbf{K}_0 + E^{-1}\right)^{-1} \left(\beta^\dagger - \beta_0\right) - \\ - \frac{1}{2}\left(\hat{\beta}_s - \beta_0\right)^T \left(\mathbf{K}_0 + E_s^{-1}\right)^{-1} \left(\hat{\beta}_s - \beta_0\right) + \\ + \underline{y}_s^T \mathbf{K}_s^{-1} \mathbf{F}_s^T \left(E^{-1} - E_s^{-1}\right) \mathbf{F}_s \mathbf{K}_s^{-1} \underline{y}_s,\end{aligned}$$

$$\begin{aligned}\tilde{\varphi} = \frac{1}{2}\overline{y}_{LS}^T \left(\mathbf{K}_s|\mathbf{K}\right)^{-1} \xi_u^T E^{-1} \left(\mathbf{K}_0 + E^{-1}\right)^{-1} \times \\ \times E^{-1} \xi_u \left(\mathbf{K}_s|\mathbf{K}\right)^{-1} \overline{y}_{LS} - \\ - \overline{y}_{LS}^T \left(\mathbf{K}_s|\mathbf{K}\right)^{-1} \xi_u^T \beta^\dagger + \frac{1}{2}\overline{y}_{LS}^T \left(\mathbf{K}_s|\mathbf{K}\right)^{-1} \overline{y}_{LS} - \\ - \frac{1}{2}\overline{y}_{LS}^T \left(\mathbf{K}_s|\mathbf{K}\right)^{-1} \xi_u^T E^{-1} \xi_u \left(\mathbf{K}_s|\mathbf{K}\right)^{-1} \overline{y}_{LS} + \\ + \left(\beta^\dagger - \beta_0\right)^T \left(\mathbf{K}_0 + E^{-1}\right)^{-1} E^{-1} \xi_u \left(\mathbf{K}_s|\mathbf{K}\right)^{-1} \overline{y}_{LS},\end{aligned}$$

with $\beta^\dagger = E^{-1}\mathbf{F}_s\mathbf{K}_s^{-1}\underline{y}_s$, $\hat{\beta}_s = E_s^{-1}\mathbf{F}_s\mathbf{K}_s^{-1}\underline{y}_s$ and $(\mathbf{K}_s|\mathbf{K}) = \mathbf{K}_u - \mathbf{K}_{us}\mathbf{K}_s^{-1}\mathbf{K}_{su}$.

In Proposition IV.1, the function $\hat{\varphi}$, which does not depend on the new data, signifies the change in uncertainty about σ^2 which may be predicted by assuming the generalized least squares estimate $\underline{\hat{y}}_{\mathrm{LS}}$. On the other hand, the function $\tilde{\varphi}$ denotes the extra uncertainty induced by having made that prediction, once the data $\underline{y}_u$ have been measured.

Remark IV.2 Each term in $\tilde{\varphi}$ is multiplied by the least squares estimation error, $\overline{y}_{\mathrm{LS}}$. Thus, the closer the least squares estimation is, the closer $\tilde{\varphi}$ is to zero. •

Using Proposition IV.1, we can rewrite the one-step ahead average prediction variance as follows. Let $\varphi^{(1:k)}$ and $\tilde{\varphi}^{(k)}$ denote the value of φ and $\tilde{\varphi}$ calculated with information up to timestep k, respectively. Let $\hat{\varphi}^{(k)} : \Omega^{(k)} \to \mathbb{R}$ map the location of the next set of measurements to the value of $\hat{\varphi}$ at timestep k. Let $\gamma^{(k)} = \nu + n * (k+1) + 1$, and let $(P, k+1)$ denote the space-time locations at spatial positions $P = (p_1, \ldots, p_n) \in \mathcal{D}^n$ and time $k+1$. In order to optimize the average posterior predictive variance at the $k+1$st timestep, we should then choose the locations P which minimize the function $\mathcal{A}^{(k)} : \mathcal{D}^n \to \mathbb{R}$ defined by

$$\begin{aligned}\mathcal{A}^{(k)}(P) = \frac{\varphi^{(1:k)} + \hat{\varphi}^{(k)}(P)}{\gamma^{(k)}} \times \\ \times \int_{\mathcal{D}} \int_T \phi\left((s,t); (\underline{x}^{(1:k)}, (P, k+1))\right) dt\, ds. \quad (5)\end{aligned}$$

Once locations have been fixed at P^*, and new measurements have been taken, we can update φ with the equation,

$$\varphi^{(1:k+1)} = \varphi^{(1:k)} + \hat{\varphi}^{(k)}(P^*) + \tilde{\varphi}^{(k)}. \quad (6)$$

In Section V we will show how the parts of $\varphi^{(1:k+1)}$ can be calculated in a distributed way by $\mathcal{N}$ using consensus and the distributed Jacobi overrelaxation algorithm. However, due to dependence on the quantity $\mathbf{k}^T\mathbf{K}^{-1}\mathbf{k}$, the conditional variance, ϕ, can not. In the next section, we detail an upper bound for ϕ, which may be computed locally by each node.

B. Upper bound of the average posterior predictive variance

Here we compute an upper bound of $\mathcal{A}^{(k)}$ defined in (5). We begin by providing a result that isolates the effect of a subset of measurements on ϕ.

Lemma IV.3 (Upper bound on conditional variance) *Let $\underline{y} = (\underline{y}_1, \underline{y}_2) \in \mathbb{R}^n$ denote a full set of measurements at distinct locations $\underline{x} = (\underline{x}_1, \underline{x}_2) \in \mathcal{D}_e^n$, with $\underline{x}_1 = (x_1, \ldots, x_{n_1}) \in \mathcal{D}_e^{n_1}$ and $\underline{x}_2 = (x_{n_1+1}, \ldots, x_n) \in \mathcal{D}_e^{n_2}$. Let $\mathbf{F}_1 \in \mathbb{R}^{p \times n_1}$, respectively $\mathbf{F}_2 \in \mathbb{R}^{p \times n_2}$ denotes the matrix whose jth row is $f_j(\underline{x}_1)$, respectively $f_j(\underline{x}_2)$, and define,*

$$
\begin{aligned}
\phi(\underline{x}_2;\underline{x}_1) &= \mathbf{K}_2 - \mathbf{K}_{21}\mathbf{K}_1^{-1}\mathbf{K}_{12} + \xi_2^T \left(\mathbf{K}_0^{-1} + E_1\right)^{-1} \xi_2 \\
\Gamma_1 &= \mathbf{k}_{20} - \mathbf{K}_{21}\mathbf{K}_1^{-1}\mathbf{k}_{10} + \xi_2^T E_1^{-1}\xi_{01} \\
\xi_2 &= \mathbf{F}_2 - \mathbf{F}_1\mathbf{K}_1^{-1}\mathbf{K}_{12} \in \mathbb{R}^{p\times n_2} \\
\xi_{01} &= f(x_0) - \mathbf{F}_1\mathbf{K}_1^{-1}\mathbf{k}_{10} \in \mathbb{R}^{p} \\
\mathbf{K}_{21} &= \mathbf{K}_{12}^T = \text{Cor}[\underline{y}_2, \underline{y}_1] \in \mathbb{R}^{n_2\times n_1} \\
\mathbf{K}_j &= \text{Cor}[\underline{y}_j, \underline{y}_j] \in \mathbb{R}^{n_j\times n_j},\ j \in \{1,2\} \\
\mathbf{k}_{j0} &= \text{Cor}[\underline{y}_j, Z] \in \mathbb{R}^{n_j},\ j \in \{1,2\}.
\end{aligned}
$$

Then we have the upper bound,

$$
\phi(x_0;\underline{x}) = \phi(x_0;\underline{x}_1) - \Gamma_1^T\phi(\underline{x}_2;\underline{x}_1)^{-1}\Gamma_1 \le \phi(x_0;\underline{x}_1),
$$

with equality if $\mathbf{k}_{20} = \mathbf{0}_{n_2}$, the n_2-dimensional zero vector, and $\mathbf{K}_{21} = \mathbf{0}_{n_2\times n_1}$, a $n_2 \times n_1$ matrix of zeros.

Lemma IV.3 implies that the conditional variance can be upper bounded using only a subset of the measurements. Using this result, $\mathcal{A}^{(k)}$ can be upper bounded as follows.

Proposition IV.4 (Spatial approximation for distributed implementation) *Define $\mathit{CS}^{(1:k+1)} : \mathbb{Z}_{\ge 0} \times \mathcal{D}^n \to \mathbb{F}(\mathcal{D}_e)$ by*

$$
\begin{aligned}
&\mathit{CS}^{(1:k+1)}(j,P) = \\
&\quad \left\{(s,t) \in i_{\mathbb{F}}\left(\underline{x}^{(1:k)}, (P,k+1)\right) \mid \mathrm{d}(s, V_j(Q)) \le r\right\},
\end{aligned}
$$

i.e., the subset of measurement locations up to time $k+1$ which are correlated in space to the Voronoi cell of S_j. Let $\tilde{\mathcal{A}}_j^{(k)} : \mathcal{D}^n \to \mathbb{R}$ be defined by

$$
\begin{aligned}
\tilde{\mathcal{A}}_j^{(k)}(P) = &\frac{\varphi^{(1:k)} + \hat{\varphi}^{(k)}(P)}{\gamma^{(k)}}\times \\
&\times \int_{V_j(Q)}\int_T \phi\left((s,t); \mathit{CS}^{(1:k+1)}(j,P)\right)\, dt\, ds.
\end{aligned}
$$

Then $\mathcal{A}^{(k)} \le \tilde{\mathcal{A}}^{(k)} = \sum_{j=1}^{m} \tilde{\mathcal{A}}_j^{(k)}$. In addition, equality holds if, for all $j \in \{1,\dots,m\}$, the points in $\mathit{CS}^{(1:k+1)}(j,P)$ are not correlated to other measurement locations outside it.

We refer to $\tilde{\mathcal{A}}^{(k)}$ as the *aggregate average prediction variance*. Unlike $\mathcal{A}^{(k)}$, the function $\tilde{\mathcal{A}}^{(k)}$ may be computed in a distributed manner over $\mathcal{N}$.

C. Smoothness of the aggregate average prediction variance

Next, we characterize the smoothness properties of $\tilde{\mathcal{A}}^{(k)}$ by examining the ith partial derivative. For simplicity, let ∇_i denote $\frac{\partial}{\partial p_i}$. Given matrix, A, we denote by $\nabla_i A$ the component-wise partial derivative of A. Here we drop the superscript notation and assume that the timestep k is fixed, where $\underline{x} = ((P,k+1), \underline{x}^{(1:k)}) \in \mathbb{R}^{n*(k+1)}$, so that p_i is the spatial location of x_i. So the ith row and column of $\mathbf{K}$, e.g., with $i \le n$, are the correlations between $(p_i, k+1)$ and $\underline{x}$.

Lemma IV.5 *Assume that $f_1,\dots,f_p$ and the covariance of Z are C^1 with respect to the spatial position of their arguments. Then the map $P \mapsto \phi(x_0; \mathit{CS}^{(1:k+1)}(j,P))$ is C^1 on $\Omega^{(k)}$ and the ith component of its gradient is*

$$
\begin{aligned}
\nabla_i\phi = &-2\mathbf{k}^T\mathbf{K}^{-1}\nabla_i\mathbf{k} + \mathbf{k}^T\mathbf{K}^{-1}\nabla_i\mathbf{K}\mathbf{K}^{-1}\mathbf{k} - \\
&- \xi_0^T\left(\mathbf{K}_0^{-1} + E\right)^{-1}\nabla_i E\left(\mathbf{K}_0^{-1} + E\right)^{-1}\xi_0 + \\
&+ 2\xi_0^T\left(\mathbf{K}_0^{-1} + E\right)^{-1}\nabla_i\xi_0,\ \textit{where} \\
\nabla_i\xi_0 = &-\nabla_i\mathbf{F}\mathbf{K}^{-1}\mathbf{k} - \mathbf{F}\mathbf{K}^{-1}\nabla_i\mathbf{k} + \mathbf{F}\mathbf{K}^{-1}\nabla_i\mathbf{K}\mathbf{K}^{-1}\mathbf{k} \\
\nabla_i E = &\nabla_i\mathbf{F}\mathbf{K}^{-1}\mathbf{F}^T + \mathbf{F}\mathbf{K}^{-1}\nabla_i\mathbf{F}^T - \mathbf{F}\mathbf{K}^{-1}\nabla_i\mathbf{K}\mathbf{K}^{-1}\mathbf{F},
\end{aligned}
$$

where the matrices are built with a location vector comprised of an ordering of the elements of $\mathit{CS}^{(1:k+1)}(j,P)$.

These matrix partial derivatives have some sparsity structure which is worth noting. The matrix $\nabla_i\mathbf{F} \in \mathbb{R}^{p\times n(k+1)}$ is nonzero only in the column corresponding to $f((p_i, k+1))$. The matrix $\nabla_i\mathbf{K} \in \mathbb{R}^{n(k+1)\times n(k+1)}$ is nonzero only in the row and column corresponding to the location $(p_i, k+1)$. Additionally, due to the finite correlation range, only those elements corresponding to correlation with other measurement locations $x = (s,t)$ which satisfy $\|p_i - s\| \le r$ are nonzero.

Lemma IV.6 *Under the assumptions of Lemma IV.5, 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 position of their arguments. Then the map $P \mapsto \nabla_i\phi(x_0; \mathit{CS}^{(1:k+1)}(j,P))$ is globally Lipschitz on $\Omega^{(k)}$.*

Note that the value of $\hat{\varphi}^{(k)}(P)$ depends on P only through the matrix E, whose partial derivative is given in Lemma IV.5. This leads us to the following continuity results.

Lemma IV.7 *Under the assumptions of Lemma IV.5, $\hat{\varphi}^{(k)}$ is C^1 on $\Omega^{(k)}$ and the ith component of its gradient is*

$$
\begin{aligned}
\nabla_i\hat{\varphi}^{(k)}(P) = &-\left(\beta^\dagger - \beta_0\right)^T\left(\mathbf{K}_0 + E^{-1}\right)^{-1}E^{-1}\nabla_i E\beta^\dagger + \\
&+ \frac{1}{2}\left(\beta^\dagger - \beta_0\right)^T\left(\mathbf{K}_0 + E^{-1}\right)^{-1}E^{-1}\nabla_i E\times \\
&\times E^{-1}\left(\mathbf{K}_0 + E^{-1}\right)^{-1}\left(\beta^\dagger - \beta_0\right) - \left(\beta^\dagger\right)^T\nabla_i E\beta^\dagger.
\end{aligned}
$$

Additionally, under the assumptions of Lemma IV.6, $\hat{\varphi}^{(k)}$ is globally Lipschitz on $\Omega^{(k)}$.

We are finally ready to state the smoothness properties of $\tilde{\mathcal{A}}^{(k)}$ and provide an explicit expression for its gradient.

Proposition IV.8 *Under the assumptions of Lemma IV.5, $\tilde{\mathcal{A}}^{(k)}$ is C^1 on $\Omega^{(k)}$ and the ith component of its gradient is*

$$
\begin{aligned}
\nabla_i\tilde{\mathcal{A}}^{(k)}(P) = &\frac{\varphi^{(1:k)} + \hat{\varphi}^{(k)}(P)}{\gamma^{(k)}}\times \\
&\times \int_{V_j(Q)}\int_T \nabla_i\phi\big((s,t); \mathit{CS}^{(1:k+1)}(j,P)\big)\, dt\, ds + \\
&+ \frac{\nabla_i\hat{\varphi}^{(k)}(P)}{\gamma^{(k)}}\int_{V_j(Q)}\int_T \phi\big((s,t); \mathit{CS}^{(1:k+1)}(j,P)\big)\, dt\, ds.
\end{aligned}
$$

Additionally, under the assumptions of Lemma IV.6, $\tilde{\mathcal{A}}^{(k)}$ is globally Lipschitz on $\Omega^{(k)}$.

V. Distributed computation of aggregate average prediction variance and its gradient

In this section, we substantiate our assertion that the aggregate average prediction variance and its gradient are distributed over the network $\mathcal{N}$. We structure our discussion in three parts motivated by the expressions of the function $\tilde{\mathcal{A}}^{(k)}$ and its gradient in Propositions IV.4 and IV.8, respectively. First, in Section V-A, we discuss how each node S_j can calculate the quantities $\hat{\varphi}^{(k)}$ and $\nabla_i\hat{\varphi}^{(k)}$, for some appropriate $i \in \{1,\dots,n\}$. Second, in Section V-B, we specify how S_j can compute $\tilde{\mathcal{A}}_j^{(k)}$ and its gradient (assuming it knows $\varphi^{(1:k)}$), and then the network $\mathcal{N}$ can fuse this information. Finally, in Section V-C, we discuss how S_j can calculate $\tilde{\varphi}^{(k)}$ (and hence $\varphi^{(1:k+1)}$).

A. Distributed calculation of $\hat{\varphi}^{(k)}$ and its gradient

In general, the matrices involved in the calculation of φ depend on measurements and locations known to multiple nodes. Furthermore, multiple measurements and locations are known to each node. The type of distributed algorithms discussed in Section II-B may be performed in a similar manner whether each node knows one element or multiple elements, as long as the network is connected and all of the information is known in a non-overlapping way by some node in the network. Since $\mathcal{V}(Q)$ describes a partition of the physical space, we may partition all measurement locations by region. Thus for each $(s,t) \in i_{\mathbb{F}}(\underline{x}^{(1:k)})$, there is exactly one $j \in \{1,\dots,m\}$ such that $s \in V_j(Q)$. Let $R_{\text{in}}^{(1:k)} : \mathbb{N} \to \mathbb{F}(\mathbb{N})$ map the index of the node to the set of indices of past locations whose spatial position lies inside its Voronoi cell,

$$R_{\text{in}}^{(1:k)}(j) = \{i \in \{1,\dots,nk\} \mid x_i = (s,t) \text{ and } s \in V_j(Q)\}\,.$$

With a slight abuse of notation, define $R_{\text{in}}^{(1:k+1)} : \mathbb{N} \times \mathcal{D}^n \to \mathbb{F}(\mathbb{N})$ to be the extension of this set to future measurements,

$$\begin{aligned} R_{\text{in}}^{(1:k+1)}(j,P) = R_{\text{in}}^{(1:k)}(j) \cup \\ \{i \in \{nk+1,\dots,n(k+1)\} \mid p_{i-nk} \in V_j(Q)\}\,. \end{aligned}$$

We are now ready to provide an iterative solution for the network to calculate the quantity $\hat{\varphi}^{(k)}$ and its gradient. The first result illustrates the parts of the equation which do not include the locations P, and may therefore be calculated once before the gradient descent algorithm. We use the notation $\text{col}_i(M)$ to denote the ith column of the matrix M.

Proposition V.1 *Assume that S_j for each $j \in \{1,\dots,m\}$ knows x_i, y_i for each $i \in R_{in}^{(1:k)}(j)$. After p executions of the WLS algorithm, S_j has access to,*

#1: $col_i\left(\mathbf{F}_s\mathbf{K}_s^{-1}\right) \in \mathbb{R}^p,\ i \in R_{in}^{(1:k)}(j)$ via JOR;
#2: $\mathbf{F}_s\mathbf{K}_s^{-1}\underline{y}_s \in \mathbb{R}^p$ via consensus;
#3: $E_s \in \mathbb{R}^{p\times p}$ via consensus;
#4: $\hat{\beta}_s \in \mathbb{R}^p$ from items #2 and #3.

Furthermore, if $k=1$, then the initial value of $\varphi^{(1)}$ can be calculated with the information above.

Next, we describe calculations which may be done at each step of the gradient descent algorithm for set of locations P.

Proposition V.2 *Given a set of locations, $P \in \Omega^{(k)}$, assume that S_j for each $j \in \{1,\dots,m\}$ knows x_i for each $i \in R_{in}^{(1:k+1)}(j,P)$ and the results of Proposition V.1. After p executions of WLS, S_j has access to,*

#5: $col_i\left(\mathbf{F}\mathbf{K}^{-1}\right) \in \mathbb{R}^p,\ i \in R_{in}^{(1:k+1)}(j,P)$ via JOR;
#6: $E \in \mathbb{R}^{p\times p}$ via consensus;
#7: $\beta^\dagger \in \mathbb{R}^p$ via consensus;

Using the results of Lemma II.2, S_j has access to,

#8: $\nabla_i E$ for $i \in \{i \in \{1,\dots,n\} \mid p_i \in V_j(Q)\}$;
#9: $\nabla_i \beta^\dagger$ for $\{i \in \{1,\dots,n\} \mid p_i \in V_j(Q)\}$.

After these computations, S_j can calculate $\hat{\varphi}^{(k)}$ and $\nabla_i\hat{\varphi}^{(k)}$ at P for each robot in $\{i \in \{1,\dots,n\} \mid p_i \in V_j(Q)\}$.

B. Distributed calculation of $\tilde{\mathcal{A}}^{(k)}$ and its gradient

In the definition of $\tilde{\mathcal{A}}_j^{(k)}$ in Proposition IV.4, the integrand depends only on measurements known to S_j. This is also the case for the gradient of $\tilde{\mathcal{A}}_j^{(k)}$ in Proposition IV.8. The next result combines this information with the steps above to arrive at distributed computation of $\tilde{\mathcal{A}}^{(k)}$ and its gradient.

Proposition V.3 *Under the assumptions of Proposition V.2, assuming S_j knows $\varphi^{(1:k)}$, the following hold:*

#10: For $i \in \{1,\dots,n\}$, node S_j can calculate

$$\int_{V_j(Q)}\int_T \phi\left((s,t); CS^{(1:k+1)}(j,P)\right)\,dt\,ds,$$

$$\int_{V_j(Q)}\int_T \nabla_i\phi\left(x_0; CS^{(1:k+1)}(j,P)\right)\,dt\,ds;$$

#11: With the results of #10 and Proposition V.2, S_j can calculate $\tilde{\mathcal{A}}_j^{(k)}$ and its gradient at P.

C. Distributed calculation of $\tilde{\varphi}^{(k)}$

Finally, we detail the computations to calculate $\tilde{\varphi}^{(k)}$ and thus update $\varphi^{(1:k+1)}$, once new measurements are available. For the purposes of distributed calculation, we write

$$\begin{aligned} \tilde{\varphi}^{(k)} = \frac{1}{2}\left(\underline{y}^T\mathbf{K}^{-1}\underline{y} - \underline{y}_s^T\mathbf{K}_s^{-1}\underline{y}_s\right) - \frac{1}{2}\zeta^T\left(\mathbf{K}_0^{-1}+E\right)^{-1}\zeta \\ - \zeta^T\left(\mathbf{K}_0^{-1}+E\right)^{-1}\mathbf{F}_s\mathbf{K}_s^{-1}\underline{y}_s - \beta_0^T\left(\mathbf{K}_0+E^{-1}\right)^{-1}E^{-1}\zeta, \end{aligned}$$

where $\zeta = \mathbf{F}\mathbf{K}^{-1}\underline{y} - \mathbf{F}_s\mathbf{K}_s^{-1}\underline{y}_s$.

Proposition V.4 *Given full vectors of measurements $\underline{y}$ and locations $\underline{x}$ up to timestep $k+1$, with previous vectors $\underline{y}_s$ and $\underline{x}_s$ up to timestep k, assume that S_j for each $j \in \{1,\dots,m\}$ knows x_i, y_i for each $i \in R_{in}^{(1:k+1)}(j)$, and the result of #2. After p executions of WLS, S_j has access to,*

#12: $\mathbf{K}^{-1}\underline{y} \in \mathbb{R}^{n(k+1)}$ - element $i \in R_{in}^{(1:k+1)}(j)$ via JOR;
#13: $\underline{y}^T\mathbf{K}^{-1}\underline{y} \in \mathbb{R}$ via consensus;
#14: $\mathbf{F}\mathbf{K}^{-1}\underline{y} \in \mathbb{R}^p$ via consensus;
#15: $\zeta \in \mathbb{R}^p$ with items #2 and #14.

After an additional execution of WLS, S_j has access to,

#16: $\mathbf{K}_s^{-1}\underline{y}_s \in \mathbb{R}^{nk}$ - element $i \in R_{in}^{(1:k+1)}(j)$ via JOR;
#17: $\underline{y}_s^T\mathbf{K}_s^{-1}\underline{y}_s \in \mathbb{R}$ via consensus.

Furthermore, with these calculations and the results of #2, S_j can calculate $\tilde{\varphi}^{(k)}$, and thereby compute $\varphi^{(1:k+1)}$.

Remark V.5 Note that the number of executions of the WLS algorithm required is independent of the network size. •

VI. Distributed optimization of the aggregate average predictive variance

Here we outline a distributed version of the projected gradient descent algorithm (see, e.g. [21]), which is guaranteed to converge to a stationary point of $\tilde{\mathcal{A}}^{(k)}$ on $\Omega^{(k)}$. For convenience, let $P'_j : \mathbb{R} \times \mathcal{D}^n \to \mathbb{F}(\mathcal{D})$ map a step size and configuration to the set of next locations calculated by S_j,

$$P'_j(\alpha, P) = \Big\{ \mathrm{proj}_{\Omega_i} \left(p_i + \alpha \nabla_i \tilde{\mathcal{A}}(P) \right), \\ \text{foreach } i \text{ s.t. } \mathrm{d}\left(p_i, V_j(Q)\right) \leq r + u_{\text{max}} + \omega \Big\}.$$

Let $\mathrm{d}_j : \mathbb{R} \times \mathcal{D}^n \to \mathbb{R}_{\geq 0}$ denote the total distance traveled by robots entering $V_j(Q)$, i.e.,

$$\mathrm{d}_j\left(\alpha, P\right) = \sum_{\substack{i \in \{1,\dots,n\} \text{ such that} \\ \mathrm{proj}_{\Omega_i}\left(p_i + \alpha \nabla_i \tilde{\mathcal{A}}(P)\right) \in V_j(Q)}} \| \mathrm{proj}_{\Omega_i} \left(p_i + \alpha \nabla_i \tilde{\mathcal{A}}(P) \right) - p_i \|^2.$$

Globally, let $P' : \mathbb{R} \times \mathcal{D}^n \to \mathcal{D}^n$, $P'(\alpha, P) = \mathrm{proj}_{\Omega}(P + \alpha \nabla \tilde{\mathcal{A}}(P))$. Table I describes a distributed line search with a starting position of $P \in \Omega$. The line search starts with a factor α_{max} which scales the smallest nonzero partial to u_{max}, ensuring all robots with nonzero partial derivatives can move the maximum distance,

$$\alpha_{\text{max}} = \frac{u_{\text{max}}}{\min\{\|\nabla_i \tilde{\mathcal{A}}(P)\| \mid \nabla_i \tilde{\mathcal{A}}(P) \neq 0\}}. \tag{7}$$

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

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

where α is chosen via Distributed Line Search Algorithm. When $|\tilde{\mathcal{A}}^{(k)}(P_{l+1}^\dagger) - \tilde{\mathcal{A}}^{(k)}(P_l^\dagger)| = 0$, the 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 the location between timesteps. The overall algorithm is in Table II.

Proposition VI.1 *The* Distributed Projected Gradient Descent Algorithm *is distributed over the network $\mathcal{N}$. Moreover, under the assumptions of Lemma IV.6, any execution is such that the robots do not collide and, at each timestep after the first, measurements are taken at stationary configurations of $P \mapsto \tilde{\mathcal{A}}^{(k)}(P)$ over $\Omega^{(k)}$.*

The proposed algorithm is robust to agent failures. If an agent stops sending position information, it ceases to receive new control vectors. The rest of the network continues operating with the available resources and will eventually sample the areas previously covered by the failing agents.

```
Name:      Distributed Line Search Algorithm
Goal:      Compute step size for gradient descent of $\tilde{\mathcal{A}}^{(k)}$
Input:     Configuration, $P = (p_1, \dots, p_n) \in \mathcal{D}^n$
Assumes:   (i) Connected network of static nodes
           (ii) $S_j$ knows $p_i$, $\tilde{\mathcal{A}}_j^{(k)}(P)$, $\nabla_i \tilde{\mathcal{A}}^{(k)}(P)$ and $\Omega_i$
           for each robot within communication range
           (iii) $S_j$ knows items #1-#4 from Proposition V.1,
           $\varphi^{(1:k)}$, and $\gamma^{(k)}$
           (iv) Shrinkage factor $\tau$ and tolerance $\theta \in (0,1)$
           known a priori by all static nodes
Output:    Step size $\tau \in \mathbb{R}$.
-----------------------------------------------------------------------
Initialization
  $S_1, \dots, S_m$ calculate $\alpha_{\text{max}}$, cf. (7) via a consensus algorithm
For $j \in \{1, \dots, m\}$, node $S_j$ executes concurrently
  $\alpha = \alpha_{\text{max}}$
  repeat
    calculates $\mathrm{d}_j\left(\alpha, P\right)^2$
    calculates $\hat{\varphi}^{(k)}\left(P'_j(\alpha, P)\right)$ according to Proposition V.2
    calculates $\tilde{\mathcal{A}}_j^{(k)}\left(P'_j(\alpha, P)\right)$
    execute consensus algorithm to calculate the following:
        $\tilde{\mathcal{A}}^{(k)}\left(P'(\alpha, P)\right) = \sum_{j=1}^{m} \tilde{\mathcal{A}}_j^{(k)}\left(P'_j(\alpha, P)\right)$
        $\left\|P - P'(\alpha, P)\right\|^2 = \sum_{j=1}^{m} \mathrm{d}_j\left(\alpha, P\right)^2$
    $\nu = \frac{\theta}{\alpha} \left\|P - P'(\alpha, P)\right\|^2 + \tilde{\mathcal{A}}^{(k)}(P'(\alpha, P)) - \tilde{\mathcal{A}}^{(k)}(P)$
    if $\nu > 0$ then
      $\alpha = \alpha\tau$
  until $\nu \leq 0$
```

TABLE I

Distributed Line Search Algorithm.

VII. Conclusions and future work

We have considered a network of static computing nodes and mobile robotic sensing platforms taking measurements of a time-varying random process with covariance known up to a scaling parameter. We have used a Bayesian approach, treating the field as a spatiotemporal Gaussian random process, and developed a novel iterative approach to calculating the variance of the posterior predictive distribution. Using this sequential formulation, we have developed a projected gradient descent algorithm which is distributed over the network of nodes and robots. Future work will focus on the investigation of theoretical guarantees on the accuracy of the approximation $\tilde{\mathcal{A}}^{(k)}$ and on the robustness to failure of the proposed coordination algorithm, the quantification of the communication requirements of the proposed strategy, and software implementation of the approach in several illustrative scenarios.

VIII. Acknowledgments

The authors would like to thank Bruno Sansó and Herbie Lee for multiple conversations 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.

Name: DISTRIBUTED PROJECTED GRADIENT DESCENT ALGORITHM

Goal: Find a local minimum of $\tilde{\mathcal{A}}^{(k)}$ within $\Omega^{(k)}$.

Assumes: (i) Connected network of static computing nodes and mobile robotic sensing agents
(ii) Static nodes deployed over $\mathcal{D}$ such that $R \geq \max_{i \in \{1,\dots,m\}} \{\mathrm{CR}(V_i(Q))\} + r + u_{\max}$, robotic agents in initial configuration $P^{(1)} \in \mathcal{D}^n$
(iii) Line search shrinkage factor τ and tolerance value $\theta \in (0,1)$ known a priori by all nodes
(iv) A termination marker known to all nodes and robots which may be sent to mark the end of a gradient descent loop.

Uses: (i) Each node uses the temporary vectors P_{cur}, respectively P_{next} to hold the configuration at the current, respectively next step of the gradient projection algorithm. For ease of exposition, we use global notation although S_j only calculates and uses the parts of these vectors which correspond to agents currently within communication range.

At time $k \in \mathbb{Z}_{\geq 0}$, node S_j executes:

```
sets $R_{\text{cov}}(j) = \{R_i \mid \mathrm{d}(p_i(k), V_j(Q)) \leq r\}$
collects initial samples and locations from $R_i$ for each $i \in R_{\text{cov}}(j)$.
**if** $k = 1$ **then**
    calculates $\varphi^{(1)}$ (cf. Prop. V.1)
**else**
    calculates $\tilde{\varphi}^{(k-1)}$ (cf Prop. V.4)
    sets $\varphi^{(1:k)} = \varphi^{(1:k-1)} + \hat{\varphi}^{(k-1)} + \tilde{\varphi}^{(k-1)}$
computes $\tilde{\mathcal{A}}_j^{(k)}\big(P^{(k)}\big)$, and then $\tilde{\mathcal{A}}^{(k)}\big(P^{(k)}\big)$ via consensus
sets $P_{\text{next}} = P^{(k)}$
**repeat**
    sets $P_{\text{cur}} = P_{\text{next}}(j)$ and calculates $-\nabla \tilde{\mathcal{A}}_j^{(k)}(P_{\text{cur}})$ (cf. Prop. V.3)
    transmits vector $\nabla_i \tilde{\mathcal{A}}_j^{(k)}(P_{\text{cur}})$ to all robots in $R_{\text{cov}}(j)$
    collects sum $\nabla_i \tilde{\mathcal{A}}^{(k)}(P_{\text{cur}})$ from all robots in $R_{\text{cov}}(j)$
    runs DISTRIBUTED LINE SEARCH ALGORITHM at $P_{\text{cur}}$ to get $\alpha$
    sets $P_{\text{next}} = P_{\text{cur}} + \alpha \nabla \tilde{\mathcal{A}}^{(k)}(P_{\text{cur}})$
    calculates $|\tilde{\mathcal{A}}^{(k)}(P_{\text{next}}) - \tilde{\mathcal{A}}^{(k)}(P_{\text{cur}})|$ from known information
**until** $|\tilde{\mathcal{A}}^{(k)}(P_{\text{next}}) - \tilde{\mathcal{A}}^{(k)}(P_{\text{cur}})| = 0$
sets $P^{(k+1)} = P_{\text{next}}$
sends a termination marker to all robots currently in $V_j(Q)$
conveys $p_i(k+1)$ to robots that currently belong to $V_j(Q)$
```

At time $k \in \mathbb{Z}_{\geq 0}$, robot R_i executes:

```
takes measurement at $p_i(k)$
sets $S_{\text{cov}}(i) = \{S_j \mid \mathrm{d}(p_i(k), V_j(Q)) \leq r\}$
sends measurement and position to all nodes in $S_{\text{cov}}(i)$
**repeat**
    receives $\nabla_i \tilde{\mathcal{A}}_j^{(k)}(P^{(k)})$ from nodes in $S_{\text{cov}}(i)$
    calculates sum $\nabla_i \tilde{\mathcal{A}}^{(k)}(P^{(k)})$
    sends $\nabla_i \tilde{\mathcal{A}}^{(k)}(P^{(k)})$ to all nodes in $S_{\text{cov}}(i)$
**until** receives termination marker from any node
receives next location $p_i(k+1)$
moves to $p_i(k+1)$.
```

TABLE II

DISTRIBUTED PROJECTED GRADIENT DESCENT ALGORITHM.

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

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

[5] M. L. Stein, "The screening effect in kriging," *The Annals of Statistics*, vol. 30, pp. 298–323, Feb. 2002.

[6] 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.

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

[8] 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 Verlag, 2002.

[9] 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.

[10] 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.

[11] 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. 45, pp. 48–74, Jan. 2007. Special Issue on Networked Control Systems.

[12] 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.

[13] S. Martínez, "Distributed representation of spatial fields through an adaptive interpolation scheme," in *American Control Conference*, (New York City, NY), pp. 2750–2755, 2007.

[14] 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.

[15] R. Graham and J. Cortés, "A cooperative deployment strategy for optimal sampling in spatiotemporal estimation," in *IEEE Conf. on Decision and Control*, (Cancun, Mexico), Dec. 2008. Accepted. To appear.

[16] 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.

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

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

[19] R. A. Freeman, P. Yang, and K. M. Lynch, "Stability and convergence properties of dynamic average consensus estimators," in *IEEE Conf. on Decision and Control*, (San Diego, CA), pp. 398–403, Dec. 2006.

[20] J. Cortés, "Distributed Kriged Kalman filter for spatial estimation," *IEEE Transactions on Automatic Control*, 2007. Submitted.

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