INTERNATIONAL JOURNAL OF ROBUST AND NONLINEAR CONTROL
*Int. J. Robust Nonlinear Control* 2009; **1**:1–2 *Prepared using* rncauth.cls *[Version: 2002/11/11 v1.00]*

# Cooperative adaptive sampling of random fields with partially known covariance

Rishi Graham[1,*] and Jorge Cortés[2]

[1]*Department of Applied Mathematics and Statistics, University of California, Santa Cruz*
[2]*Department of Mechanical and Aerospace Engineering, University of California, San Diego*

SUMMARY

This paper considers autonomous robotic sensor networks taking measurements of a physical process for predictive purposes. The physical process is modeled as a spatiotemporal random field. The network objective is to take samples at locations that maximize the information content of the data. The combination of information-based optimization and distributed control presents difficult technical challenges as standard measures of information are not distributed in nature. Moreover, the lack of prior knowledge on the statistical structure of the field can make the problem arbitrarily difficult. Assuming the mean of the field is an unknown linear combination of known functions and its covariance structure is determined by a function known up to an unknown parameter, we provide a novel distributed method for performing sequential optimal design by a network comprised of static and mobile devices. We characterize the correctness of the proposed algorithm and examine in detail the time, communication, and space complexities required for its implementation. Copyright © 2009 John Wiley & Sons, Ltd.

KEY WORDS: robotic network, spatial estimation, stochastic process, Bayesian learning

## 1. Introduction

Networks of environmental sensors are playing an increasingly important role in scientific studies, with applications to a variety of scenarios, including detection of chemical pollutants, animal monitoring, and mapping of ocean currents. Among their many advantages, robotic sensor networks can improve

*Correspondence to: Rishi Graham at rishig@ams.ucsc.edu

the efficiency of data collection, adapt to changes in the environment, and provide a robust response to individual failures. The design of coordination algorithms for networks performing these spatially-distributed sensing tasks faces the major challenge of incorporating the complex statistical techniques that come into play in the analysis of environmental processes. 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. Coordination algorithms need to be distributed and scalable to make robotic networks capable of operating in an autonomous and robust fashion. At the same time, these algorithms must be statistically driven to steer the network towards locations that provide the most informative samples about the physical processes. This work is a step forward in bridging the gap between sophisticated statistical modeling and distributed motion coordination.

Consider an experiment in which samples are collected from a dynamic scalar physical process with the goal of estimating its value at unmeasured points in a space-time domain. One motivating example is sampling subsurface ocean temperature with autonomous underwater vehicles. The relative sparsity of samples in such an experiment, see e.g., [1], suggests using stochastic methods rather than deterministic ones, particularly at small scales where ode-driven deterministic models may not take advantage of averaging effects. This can also be a useful approach if the dynamics of the field are not well known, require a high-dimensional parameter space to model, or require extremely accurate specification of initial conditions. Autonomous vehicles may take advantage of a sequential approach, in which new sample locations are chosen based on information gained from past samples. This is known in the statistical literature as *adaptive design*. We use a Bayesian approach to spatiotemporal modeling, and treat the field as a Gaussian Process (GP). A GP is an infinite-dimensional model in which any vector of realizations from the field is treated as jointly normally distributed conditional on the space-time positions. Temperature is one example of the type of correlated spatial field known to be well modeled by a GP (e.g. [2, 3]). In the Bayesian paradigm, a prediction at any point in the field takes the form of a conditional *distribution*, derived from a prior distribution and the sampled data. GP models are fully specified by mean and covariance functions, and provide powerful and flexible tools for modeling under uncertain conditions. The mean provides a smooth "trend" surface, while small scale perturbations are captured by the covariance structure. Prior uncertainty about the mean is commonly handled by treating it as an unknown regression on a set of known basis functions. A less common model, which we make use of here, also allows for an unknown scalar term in the prior

covariance, enabling estimation of fields in which the deviation from the mean surface is not known a priori. From a relatively small number of samples, statistical methods generate an accurate predictive map over the entire space-time domain, complete with a measure of predictive uncertainty. This allows complex surfaces to be estimated with simple models. Any measure of the utility of sample locations in such a model should be based on the uncertainty of the resulting estimation, reflecting deviations in the data as well as the prior uncertainty in the model. This presents a difficult challenge in a distributed setting, because the predictive uncertainty depends on all samples in a nontrivial way.

**Literature review.** Complex statistical techniques allow a detailed account of uncertainty in modeling physical phenomena. Of particular relevance to this work are [4], regarding statistical models, and [5, 6], regarding the application of optimal design techniques to Bayesian models. Optimal design [7, 8] refers to the problem of choosing sample locations which optimize estimation.

In cooperative control, various works consider mobile sensor networks performing spatial estimation tasks. [9] considers a robotic sensor network with centralized control estimating a static field from measurements with both sensing and localization error. Depending on the goal of the experiment, different types of information should be maximized [6, 10, 11]. We focus on the predictive variance as a measure of the accuracy of prediction. An alternative optimality criterion called mutual information [12, 13] is also effective for predictive purposes, but requires that samples and predictions are made on a discrete space (e.g., a grid). Using mutual information, the work [14] addresses the multiple robot path planning problem by greedily choosing way points from a discrete set of possible sensing locations. [15] chooses optimal sampling trajectories from a parameterized set of paths. Here, instead, we are interested in optimizing in a continuous design space and over the set of all possible paths. A different problem is considered in [16], where robots track level curves in a noisy scalar field. [17] develops distributed estimation techniques for predictive inference of a spatiotemporal random field and its gradient. We make use of some of the tools developed in the latter paper for distributed calculations. In [18, 19, 20, 21], the focus is on estimating deterministic fields with random measurement noise and, in some cases, stochastic evolution over discrete timesteps. A fundamental difference with these works is that, in addition to considering uncertainty about the process evolution over time, we consider fields with uncertainty in its covariance structure, i.e., fields in which the deviation from the mean surface is not known a priori and also needs to be estimated. In between the specificity of deterministic modeling and the flexibility of fully probabilistic modeling there are other

alternative methods such as distributed parameter systems [22], which introduce possibly correlated stochastic parameters into an otherwise deterministic model. In this paper, the process itself is treated as random. This allows for a simpler model and more prior uncertainty, and places the focus on estimation as opposed to inference. Complex dynamics and spatial variation are accounted for with space-time correlation instead of explicit partial differential equations. In part to cope with the additional burden of spatial uncertainty, we introduce a hybrid network of static computing nodes and mobile sensors. An alternative approach to the spatiotemporal Gaussian Process is the Kriged Kalman filter [23, 24] approach, which treats the process as a spatial GP with discrete temporal evolution governed by a Kalman filter. Instead, we use an integrated space-time model because it is more general, and because the treatment for our purpose is simpler. Of the above references, those which consider random field models do so under an assumption of known covariance. To our knowledge this is the first work in the cooperative control field which allows for prior uncertainty in the covariance of the spatiotemporal structure as well as the mean. We make use of a model derived in [25], which is the only spatial model we are aware of that makes a direct analytical connection between prior uncertainty in the covariance and the resulting predictive uncertainty. Aside from this model or derivatives, the common practice when confronted with unknown covariance is to either run a separate estimation procedure and then treat the covariance as known, or to use simulation methods such as Markov chain Montecarlo to estimate the posterior distribution. The work [26] addresses a method of choosing sample locations from a discrete space which are robust to misspecification of the covariance. Another method for handling unknown covariance has recently grown out of the exploration-exploitation approach of reinforcement learning (see, e.g. [27]). The work [28] applies this approach to the spatial estimation scenario by breaking up the objective into an exploration component which focuses on learning about the model in a discretized space and an exploitation component in which that knowledge is put to use in optimizing for prediction. Here, we require no discretization and we take full advantage of the mobile capabilities of networks of autonomous sensors. Our work is based in part on previous material presented in [29] and [30].

**Statement of contributions.** We begin with a widely accepted Bayesian model for the prediction of a spatiotemporal random field, designed to handle various degrees of knowledge about the mean and covariance. The predictive variance of this model can be written as a scaled product 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 an approximate predictive variance which may be calculated efficiently in a sequential and distributed manner. This includes introducing a scheduled update of the estimated covariance parameter based on uncorrelated clusters of samples. We introduce three possible approximation methods which trade off accuracy for computational burden. To our knowledge, none of these approximations have been examined for this particular optimization approach, although one of the methods is similar to the exploration-exploitation approach used by [27] to optimize for a different information criterion with different model assumptions. 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. This hybrid network architecture is motivated in part by the heavier computational capabilities of static agents and in part by the spatial structure of the problem. Our third contribution is the design of a coordination algorithm based on projected gradient descent which guarantees one-step-ahead locally optimal data collection. Due to the nature of the solution, optimality here takes into account both the unknown parameter in the covariance and the (conditional) uncertainty in the prediction. Finally, our fourth contribution is the characterization of the communication, time, and space complexities of the proposed algorithm. For reference, we compare these complexities against the ones of a centralized algorithm in which all sample information is broadcast throughout the network at each step of the optimization.

**Organization.** Section 2 introduces basic notation and describes the statistical model. Section 3 states the robotic network model and the overall network objective. The following two sections present the main results of the paper. Section 4 introduces the objective function, with attention to its smoothness properties, and discusses how the network can make the required calculations in a distributed way. Section 5 presents the cooperative strategy for optimal data collection along with correctness and complexity results. Section 6 contains our conclusions.

## 2. Preliminary notions

Let $\mathbb{R}$, $\mathbb{R}_{>0}$, and $\mathbb{R}_{\geq 0}$ denote the set of reals, positive reals and nonnegative reals, respectively, and let $\mathbb{Z}_{>0}$ and $\mathbb{Z}_{\geq 0}$ denote the sets of positive and nonnegative integers. We denote by $\lfloor x \rfloor$ and $\lceil x \rceil$ the floor

and ceiling of $x \in \mathbb{R}$, 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 $p$. We will generally use upper case to denote vectors and matrices, with the exception of the correlation vector, $\mathbf{k}$, and basis function vector, $\mathbf{f}$, which are lower case to distinguish them from the matrices $\mathbf{K}$ and $\mathbf{F}$ (see Section 2.2). Given $U = (u_1, \ldots, u_a)^T$, $a \in \mathbb{Z}_{>0}$, and $V = (v_1, \ldots, v_b)^T$, $b \in \mathbb{Z}_{>0}$, we denote by $(U,V) = (u_1, \ldots, u_a, v_1, \ldots, v_b)^T$ its concatenation. We denote by $\mathbb{F}(\Omega)$ the collection of finite subsets of $\Omega$. Let $i_{\mathbb{F}} : (\mathbb{R}^d)^n \to \mathbb{F}(\mathbb{R}^d)$ be the natural immersion, i.e., $i_{\mathbb{F}}(P)$ contains only the distinct points in $P = (p_1, \ldots, p_n)$. Note that $i_{\mathbb{F}}$ is invariant under permutations of its arguments and that the cardinality of $i_{\mathbb{F}}(p_1, \ldots, p_n)$ is in general less than or equal to $n$.

We consider a convex region $\mathcal{D} \subset \mathbb{R}^d$, $d \in \mathbb{Z}_{>0}$. The assumption of convexity is a requirement for projected gradient descent (see Section 2.1). Let $\mathcal{D}_e = \mathcal{D} \times \mathbb{R}_{\geq 0}$ denote the space of points over $\mathcal{D}$ and time. Let $\mathrm{proj}_\Omega : \mathbb{R}^m \to \Omega$ denote the orthogonal projection onto the set $\Omega$,

$$\mathrm{proj}_\Omega(x) = \operatorname*{argmin}_{y \in \Omega} \|x - y\|.$$

Let $\mathrm{dist} : \mathbb{R}^d \times \mathbb{F}(\mathbb{R}^d) \to \mathbb{R}_{\geq 0}$ denote the (minimum) distance between a point and a set, i.e., $\mathrm{dist}(s, \Omega) = \min_{q \in \Omega} \|q - s\|$. With a slight abuse of notation, for two convex sets, $\Omega_1, \Omega_2$, we will write the minimum distance between points in the two sets as, $\mathrm{dist}(\Omega_1, \Omega_2) = \min_{p \in \Omega_1} \mathrm{dist}(p, \Omega_2)$. The *$\epsilon$-contraction* of a set $\Omega$, for $\epsilon > 0$, is the set $\Omega_{\text{ctn}:\epsilon} = \{q \in \Omega \mid \mathrm{dist}(q, \mathrm{bnd}\,\Omega) \geq \epsilon\}$, where $\mathrm{bnd}\,\Omega$ denotes the boundary of $\Omega$. For a bounded set $\Omega \subset \mathbb{R}^d$, we let $\mathrm{CR}(\Omega)$ denote the *circumradius* of $\Omega$, that is, the radius of the smallest $d$-sphere enclosing $\Omega$. The *Voronoi partition* $\mathcal{V}(S) = (V_1(S), \ldots, V_n(S))$ of $\mathcal{D}$ generated by the points $S = (s_1, \ldots, s_n)$ is defined by $V_i(S) = \{q \in \mathcal{D} \mid \|q - s_i\| \leq \|q - s_j\|, \ \forall j \neq i\}$. Each $V_i(S)$ is called a *Voronoi cell*. Two points $s_i$ and $s_j$ are *Voronoi neighbors* if their Voronoi cells share a boundary.

We use $\lambda_{\min}(A)$ and $\lambda_{\max}(A)$ to denote the smallest and largest eigenvalue of the square matrix $A$, respectively. We let $\rho(A)$ denote the spectral radius of $A$, $\det A$ the determinant of $A$, $[A]_{ij}$ the $(i,j)$th element of $A$, and $\mathrm{col}_i(A)$ denote the $i$th column of $A$. Let $\mathbf{0}_{i \times j}$ denote the $i \times j$ zero matrix (or vector if either $i$ or $j$ is 1). If the dimensions are clear from the context we may omit the subscripts and use $\mathbf{0}$. Given a partitioned matrix, $A = \left[\begin{smallmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{smallmatrix}\right]$, we denote by $(A_{11}\,|A)$, respectively $(A_{22}\,|A)$ the Schur complement of $A_{11}$, respectively $A_{22}$ in $A$, i.e.,

$$(A_{11}\,|A) = A_{22} - A_{21}A_{11}^{-1}A_{12} \qquad \text{and} \qquad (A_{22}\,|A) = A_{11} - A_{12}A_{22}^{-1}A_{21}.$$

Table I provides an overview of notational conventions introduced in this section.

| Notation | Description |
|---|---|
| $d$ | Spatial dimension of experiment region |
| $\mathcal{D}$ | Convex spatial region where the experiment takes place |
| $\mathcal{D}_e$ | Space-time domain of $\mathcal{D}$ over the entire experiment |
| $\lfloor a \rfloor, \lceil a \rceil$ | Floor and ceiling of $a$ |
| $\overline{B}(p,r)$ | Closed $d$-dimensional ball of radius $r$ centered at $p$ |
| $\mathrm{proj}_\Omega(s)$ | Orthogonal projection of $s$ onto $\Omega$ |
| $\mathrm{dist}(p,\Omega), \mathrm{dist}(\Omega_1,\Omega_2)$ | Minimum point-to-set and set-to-set distance |
| $i_{\mathbb{F}}(P)$ | Natural immersion of vector $P$ (set of distinct points) |
| $\mathbb{F}(\Omega)$ | Collection of finite subsets of $\Omega$ |
| $\Omega_{\mathrm{ctn}:\epsilon}$ | The $\epsilon$-contraction of $\Omega$ |
| $\mathrm{CR}(\Omega)$ | Radius of *smallest $d$-sphere containing* $\Omega$ |
| $\mathcal{V}(S)$ | Voronoi partition generated by $S$ |
| $V_i(S)$ | The $i$th cell in the Voronoi partition, $\mathcal{V}(S)$ |
| $\lambda_{\min}(A)$, $\lambda_{\max}(A)$ | Extremal eigenvalues of square matrix, $A$ |
| $\rho(A)$ | Spectral radius of $A$ |
| $\det A$ | Determinant of $A$ |
| $[A]_{ij}$ | Element $i,j$ of matrix $A$ |
| $(B\,\vert A)$ | Schur complement of submatrix $B$ in matrix $A$ |

Table I. Notational conventions.

### *2.1. Projected gradient descent*

We describe here the constrained optimization technique known as projected gradient descent [31] to iteratively find the minima of an objective function $F : \mathbb{R}^m \to \mathbb{R}_{\geq 0}$. Let $\Omega$ denote a nonempty, closed, and convex subset of $\mathbb{R}^m$, $m \in \mathbb{Z}_{>0}$. Assume that the gradient $\nabla F$ is globally Lipschitz on $\Omega$. Consider a sequence $\{x_k\} \in \Omega$, $k \in \mathbb{Z}_{>0}$, which satisfies

$$x_{k+1} = \mathrm{proj}_\Omega\left(x_k - \alpha_k \nabla F(x_k)\right),\ x_1 \in \Omega, \tag{1}$$

where the step size, $\alpha_k$, is chosen according to the LINE SEARCH ALGORITHM described in Table II, evaluated at $x = x_k$.

| | |
|---|---|
| **Name:** | LINE SEARCH ALGORITHM |
| **Goal:** | Determine step size for the Sequence (1) |
| **Input:** | $x \in \Omega$ |
| **Assumes:** | $\tau, \theta \in (0,1)$, max step $\alpha_{\max} \in \mathbb{R}_{>0}$ |
| **Output:** | $\alpha \in \mathbb{R}_{>0}$ |

1: $\alpha := \alpha_{\max}$
2: **repeat**
3: $\quad x_{\text{new}} := \text{proj}_{\Omega}\left(x - \alpha \nabla F(x)\right)$
4: $\quad \varpi := \frac{\theta}{\alpha}\|x - x_{\text{new}}\|^2 + F(x_{\text{new}}) - F(x)$
5: $\quad$ **if** $\varpi > 0$ **then**
6: $\quad\quad \alpha := \alpha\tau$
7: **until** $\varpi \leq 0$

Table II. LINE SEARCH ALGORITHM.

In Table II, the grid size $\tau$ determines the granularity of the line search. The tolerance $\theta$ may be adjusted for a more (larger $\theta$) or less (smaller $\theta$) strict gradient descent. With $\theta > 0$, the LINE SEARCH ALGORITHM must terminate in finite time. The Armijo condition (step 7) ensures that the decrease in $F$ is commensurate with the magnitude of its gradient. A sequence $\{x_k\}_{k=1}^{\infty}$ obtained according to (1) and Table II converges in the limit [31] as $k \to \infty$ to stationary points of $F$.

### *2.2. Bayesian modeling of space-time processes*

Let $Z$ denote a random space-time process taking values on $\mathcal{D}_e$. Let $Y = (y_1, \ldots, y_n)^T \in \mathbb{R}^n$ be $n \in \mathbb{Z}_{>0}$ measurements taken from $Z$ at corresponding space-time positions $X = (x_1, \ldots, x_n)^T \in \mathcal{D}_e^n$, with $x_i = (s_i, t_i),\ i \in \{1, \ldots, n\}$. Given these data, various models allow for prediction of $Z$ at any point in $\mathcal{D}_e$, with associated uncertainty. Optimal design is the procedure of choosing where to take measurements in order to reduce the uncertainty of the resulting statistical prediction. Since prediction uncertainty drives the design procedure, it should be modeled as accurately as possible.

In a Bayesian setting, the prediction takes the form of a distribution, called the posterior predictive [32]. One advantage of a Bayesian approach is that parameters such as the mean and variance of the field may be treated as random variables, carrying forward uncertainty which informs the predictive distribution. We assume that $Z$ is a Gaussian Process. This means that any vector of realizations is treated as jointly normally distributed conditional on unknown parameters, with mean

vector and covariance matrix dictated by the mean and covariance functions of the field. Thus a prediction made after samples have been taken is the result of conditioning the posterior predictive distribution on the sampled data. When this conditional posterior distribution is analytically tractable, as in the case of the models presented here, the resulting approach provides two powerful advantages to optimal design. First, there is a direct and (under certain technical conditions) continuous map from the space-time coordinates of realizations (data and prediction) to the predictive uncertainty. Second, the joint distribution of predictions and samples allow conditioning on subsets of samples to see the effect, for instance, of optimizing over a single timestep.

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. Predictive modeling in this context is often referred to in geostatistics as *simple kriging* if the mean is also known, or *universal kriging* if the mean is treated as an unknown linear combination of known basis functions. If the covariance of the field is not known, however, few analytical results exist which take the full uncertainty (i.e., uncertainty in the field and in the parameters) into account. We present here a model [4] which allows for uncertainty in the covariance process and still produces an analytical posterior predictive distribution. We assume that the measurements are distributed as the $n$-variate normal distribution,

$$Y \sim \mathrm{Norm}_n \left( \mathbf{F}^T \beta, \sigma^2 \mathbf{K} \right). \tag{2}$$

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 $(i,j)$th element is $\mathbf{K}_{ij} = \mathrm{Cor}[y_i, y_j]$. Note that $\mathbf{K}$ is symmetric, positive definite, with 1's on the diagonal. We assume a finite correlation range in space, $r_\mathrm{s} \in \mathbb{R}_{>0}$, and in time, $r_\mathrm{t} \in \mathbb{R}_{>0}$, such that if $\|s_i - s_j\| \geq r_\mathrm{s}$ or $|t_i - t_j| \geq r_\mathrm{t}$, then $\mathbf{K}_{ij} = \mathbf{K}_{ji} = 0$. For gradient calculations, we also assume that the correlation map $s_i \mapsto \mathbf{K}_{ij}$ is $C^2$ (which implies that $s_j \mapsto \mathbf{K}_{ij}$ is also $C^2$). In many kriging applications, an assumption of second-order stationarity restricts the covariance between two samples to a function of the difference between their locations. Instead, we make no restrictions on the correlation function itself, beyond the finite range and continuous spatial differentiability. The matrix $\mathbf{F}$ is determined by a set of $p \in \mathbb{Z}_{>0}$ known basis functions $f_i : \mathcal{D}_e \to \mathbb{R}$ evaluated at the space-time locations $X$, i.e.,

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

We will also use $\mathbf{f}(x) = (f_1(x), \dots, f_p(x))^T \in \mathbb{R}^p$ to denote the vector of *time-dependent* basis functions evaluated at a single point in $\mathcal{D}_e$. It should be pointed out here that the standard approach in kriging is to use basis functions which do not change with time. We include the possibility of space-time basis functions for completeness, and because they are commonly used [33] in the related practice of "objective analysis" used in atmospheric sciences [34]. To ensure an analytical form for the posterior predictive distribution, we assume conjugate prior distributions for the parameters,

$$\beta|\sigma^2 \sim \mathrm{Norm}_p\left(\beta_0, \sigma^2\mathbf{K}_0\right), \tag{3a}$$

$$\sigma^2 \sim \Gamma^{-1}\left(\frac{\nu}{2}, \frac{q\nu}{2}\right). \tag{3b}$$

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. [35]). Since it is a correlation matrix, it should be noted that $\mathbf{K}_0$ must be positive definite. A common practice in statistics is to use $\mathbf{K}_0$ proportional to the identity matrix.

**Proposition 2.1 (Posterior predictive distribution)** *Under the Bayesian model* (2)*, the posterior predictive at $x_0 \in \mathcal{D}_e$ is a shifted Students t distribution (see, e.g. [35]) with $\nu + n$ degrees of freedom, with probability density function, for $z = Z(x_0)$,*

$$p(z|Y, X) \propto \mathrm{Var}[z|Y, X]^{-\frac{1}{2}} \left(1 + \frac{(z - \mathrm{E}[z|Y, X])^2}{(\nu + n - 2)\,\mathrm{Var}[z|Y, X]}\right)^{-\frac{\nu+n+1}{2}}.$$

*Here, the expectation is given by*

$$\mathrm{E}[z|Y, X] = \left(\mathbf{f}(x_0) - \mathbf{F}\mathbf{K}^{-1}\mathbf{k}\right)^T \beta^\dagger + \mathbf{k}^T\mathbf{K}^{-1}Y,$$

$$\beta^\dagger = (\mathbf{E} + \mathbf{K}_0^{-1})^{-1}\left(\mathbf{F}\mathbf{K}^{-1}Y + \mathbf{K}_0^{-1}\beta_0\right),$$

*where $\mathbf{E} = \mathbf{F}\mathbf{K}^{-1}\mathbf{F}^T$ and $\mathbf{k} = \mathrm{Cor}[Y, z] \in \mathbb{R}^n$. The variance is given by*

$$\mathrm{Var}[z|Y, X] = \varphi(Y, X)\phi(x_0; X),$$

$$\phi(x_0; X) = \mathrm{Cor}[z, z] - \mathbf{k}^T\mathbf{K}^{-1}\mathbf{k} + \xi_0^T\left(\mathbf{K}_0^{-1} + \mathbf{E}\right)^{-1}\xi_0,$$

$$\xi_0 = \mathbf{f}(x_0) - \mathbf{F}\mathbf{K}^{-1}\mathbf{k},$$

$$\varphi(Y, X) = \frac{1}{\nu + n - 2}\left(q\nu + (Y - \mathbf{F}^T\beta_0)^T\left(\mathbf{K} + \mathbf{F}^T\mathbf{K}_0\mathbf{F}\right)^{-1}(Y - \mathbf{F}^T\beta_0)\right).$$

The proof of the proposition follows from the application of Bayes Theorem to the model given by (3). Alternatively, it can also be derived from results in [4] and [25] using a technique similar to the one used later in the proof of Proposition I.3 in Appendix I. Note that since $\mathbf{K}_0$ and $\mathbf{K}$ are positive definite, the quantities $\phi(x_0; X)$ and $\varphi(Y, X)$ are well posed.

**Remark 2.2 (Terms in the posterior predictive variance)** Note the form for the posterior predictive variance in Proposition 2.1 as a product of two terms. The first term, $\varphi(Y, X)$, is the posterior mean of the parameter $\sigma^2$, given the sampled data. We refer to it as the *sigma mean*. The second term, $\phi(x_0; X)$, can be thought of as the scaled posterior predictive variance conditioned on $\sigma^2$. We refer to it as the *conditional variance*. •

The conditional variance is very close to what the predictive variance would look like if $\sigma^2$ were known, as we show next. The following result may be derived by applying Bayes Theorem to the model specified by (2) and (3a), with $\sigma^2$ treated as known.

**Proposition 2.3 (Kriging variance)** *If the variance parameter $\sigma^2$ is known, the result is the* universal kriging predictor, *and the posterior predictive variance takes the form,*

$$\mathrm{Var}_{UK}[z|Y, X] = \sigma^2 \left(\mathrm{Cor}[z, z] - \mathbf{k}^T\mathbf{K}^{-1}\mathbf{k} + \xi_0^T\mathbf{E}^{-1}\xi_0\right).$$

*If, in addition, the mean of the field is known, the result is the* simple kriging predictor, *and the posterior predictive variance is given by,*

$$\mathrm{Var}_{SK}[z|Y, X] = \sigma^2 \left(\mathrm{Cor}[z, z] - \mathbf{k}^T\mathbf{K}^{-1}\mathbf{k}\right).$$

**Remark 2.4 (Extension of subsequent results to kriging)** The simple and universal kriging results are simplified versions of our overall model, and results from the rest of this paper may be applied to those models with minimal modifications. An exception is that when approximating $\mathrm{Var}_{\mathrm{UK}}$ using subsets of measurements, care must be taken to ensure well-posedness. Specifically, an assumption that $n > p$ is required to ensure that the matrix $\mathbf{E}$ is nonsingular. •

Table III provides an overview of notational conventions introduced in this section.

### *2.3. Tools for distributed computation*

Consider a network, $\mathfrak{Q}$, of $m$ nodes with limited communication capabilities. We write $\mathfrak{Q} = (\mathrm{N}, E)$, where $\mathrm{N} = (\mathrm{N}_1, \ldots, \mathrm{N}_m)$ denotes the vector of nodes, and $E$ the set of communication edges (i.e.,

| Notation | Description |
|---|---|
| $X$, $x_i$ | Vector of spatiotemporal coordinates of samples, element of coordinate vector |
| $Y$, $y_i$ | Vector of sample values, element of sample vector |
| $\mathrm{E}[A], \mathrm{Var}[A]$ | Expectation and variance of random vector $A$ |
| $\mathrm{Cor}[a, b]$ | Correlation between random variables (or vectors) $a$ and $b$ |
| $z$ | Random space-time process of interest |
| $\mathbf{K}$ | Sample correlation matrix |
| $\mathbf{k}$ | Samples to prediction correlation vector |
| $\mathbf{F}$ | Matrix of basis functions evaluated at sample locations |
| $\mathbf{f}$ | Basis vector evaluated at predictive location |
| $\xi_0$ | Generalized least squares error estimating $\mathbf{f}$ from $\mathbf{F}$ |
| $\beta$ | Unknown mean regression parameters |
| $\beta_0, \mathbf{K}_0$ | Prior mean vector and correlation matrix of $\beta$ |
| $\sigma^2$ | Unknown variance scalar parameter |
| $q, \nu$ | Tuning parameters of prior distribution for $\sigma^2$ |
| $\phi$ | Predictive variance conditional on $\sigma^2$ ("conditional variance") |
| $\varphi$ | Posterior mean of $\sigma^2$ ("sigma mean") |

Table III. Statistical notation.

$(i, j) \in E$ if $\mathrm{N}_i$ and $\mathrm{N}_j$ can communicate). We are only concerned with connected graphs (i.e., graphs in which every vertex is connected to every other vertex via a sequence of edges). We will make use of the *degree*, $\deg_{\mathfrak{Q}}$, *diameter*, $\mathrm{diam}_{\mathfrak{Q}}$, and *number of edges*, $\mathrm{Ed}_{\mathfrak{Q}}$ of $\mathfrak{Q}$ defined as,

$$\deg_{\mathfrak{Q}} = \max_{i \in \{1,\dots,m\}} \deg_{\mathfrak{Q}}(i) \qquad \mathrm{diam}_{\mathfrak{Q}} = \max_{i,j \in \{1,\dots,m\}} |L_{\min}(i,j)| \qquad \mathrm{Ed}_{\mathfrak{Q}} = |E|, \tag{4}$$

where we have used $L_{\min}(i, j)$ to denote a minimum length path between vertices $i$ and $j$, and $\deg_{\mathfrak{Q}}(i) = |\{j \in \{1, \dots, m\} \mid (i, j) \in E\}|$ the degree of node $i$.

Here we briefly describe some tools for distributed computations. Let $a_{ij} \in \{0, 1\}$, $i, j \in \{1, \dots, m\}$ be 1 if $(i, j) \in E$, and 0 otherwise. Let $b = (b_1, \dots, b_m)^T \in \mathbb{R}^m$, $C = [c_{ij}] \in \mathbb{R}^{m \times m}$, and assume node $i$ knows $b_i$ and the $i$th row of $C$. Additionally assume that $c_{ii} \neq 0$ and, for $i \neq j$, $c_{ij} \neq 0$ if and only if $(i, j) \in E$. As an example, one can think of $C = \mathbf{K}$ and $b = \mathbf{k}$, with the notation of Proposition 2.3. The correlation matrix $\mathbf{K}$ has a sparsity structure determined by the physical distance

of samples from each other. Under these assumptions the following results hold.

**JOR:** The network can compute the vector $y = C^{-1}b$ via a *distributed Jacobi overrelaxation* algorithm [17, 36], formulated as the discrete-time dynamical system,

$$y_i(l+1) = (1-h)y_i(l) - \frac{h}{c_{ii}}\Big(\sum_{j\neq i} c_{ij}y_j(l) - b_i\Big), \tag{5}$$

for $l \in \mathbb{Z}_{\geq 0}$ and $i \in \{1,\dots,m\}$, where $y(0) \in \mathbb{R}^m$ and $h \in \Big(0, \frac{2}{\lambda_{\max}(C)}\Big)$. At the end of the algorithm, node $i$ knows the $i$th element of $C^{-1}b$.

**Discrete-time average consensus:** The network can compute the arithmetic mean of elements of $b$ via the discrete dynamical system [37],

$$x_i(l+1) = x_i(l) + \epsilon\sum_{j\neq i} a_{ij}(x_i(l) - x_j(l)),\ x(0) = b,$$

where $\epsilon \in (0, \frac{1}{\deg_{\Omega}})$. At the end of the algorithm, all nodes know $\frac{\sum_{i=1}^n b_i}{m}$.

**Maximum consensus:** The network can calculate the maximum value of elements of $b$ via a leader election algorithm [38]. Each node sends the current estimate of the maximum to all neighbors, then updates its estimate. If the process is repeated a number of times equal to the diameter of the network, then every node will know the maximum.

The first two results above are only exact asymptotically, but convergence is exponential with time.

## 3. Problem statement

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

### *3.1. Robotic sensor network model*

We introduce a hybrid network comprised of mobile sensors and static nodes. This model provides a stable communication structure, allows for integration of the predictive variance over the entire spatial domain, and reduces the computational burden on the mobile units. Consider a group $\{\mathrm{N}_1,\dots,\mathrm{N}_m\}$ of $m \in \mathbb{Z}_{>0}$ static nodes at spatial locations $Q = (q_1,\dots,q_m)^T \in \mathcal{D}^m$, and the Voronoi partition, $\mathcal{V}(Q)$ generated by them. $\mathrm{N}_i$ will be responsible for approximate prediction over the region $V_i(Q)$. In addition to the static nodes, consider a group $\{\mathrm{R}_1,\dots,\mathrm{R}_n\}$ of $n$ mobile robotic sensor agents. The

robots take point samples of the spatial field at discrete instants of time in $\mathbb{Z}_{\geq 0}$. Our results below are independent of the particular robot dynamics, so long as each agent is able to move up to a distance $u_{\max} \in \mathbb{R}_{>0}$ between consecutive sampling times. Since the focus of this work is the online planning of optimal sampling paths, any bounded delay incurred by network communication or calculations may be incorporated into this maximum radius of movement between sampling instants. Bounds on such delay may be inferred from the complexity analysis in Section 5.1. The communication radius of the robots is also $r_{\text{com}}$. If there is a chance that $\mathrm{R}_j$ will be within correlation range of $V_i(Q)$ after moving a distance of $u_{\max}$ (i.e., at the following sample time), then $\mathrm{N}_i$ must be able to communicate with $\mathrm{R}_j$. To that end, let

$$r_{\text{com}} \geq \max_{i \in \{1,\ldots,m\}} \{\mathrm{CR}(V_i(Q))\} + r_{\text{s}} + u_{\max}. \tag{6}$$

Among all the partitions of $\mathcal{D}$, the Voronoi partition $\mathcal{V}(Q)$ ensures that (6) is satisfied with the smallest communication radius. Figure 1 illustrates the communication requirements of the hybrid network. The robots can sense the positions of other robots within a distance of $2u_{\max}$. At discrete timesteps,

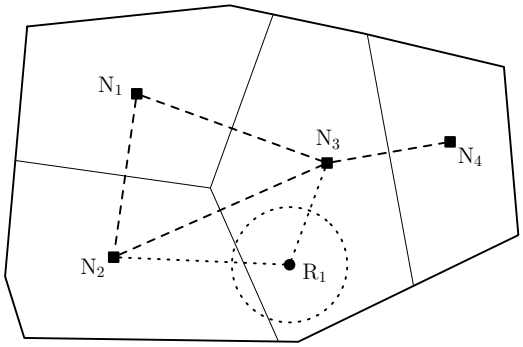



Figure 1. The static nodes are depicted as filled boxes, with the Voronoi partition boundaries as solid lines. Each node can communicate with its Voronoi neighbors, and with any mobile robot within a distance of $r_{\text{s}} + u_{\max}$ (dotted circle) of the Voronoi cell (in the plot, $N_2$ and $N_3$ can communicate with $R_1$).

each robot communicates the sample and spatial position to static nodes within communication range, along with the positions 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 hybrid network model as $\mathfrak{Q}_{\text{hybrid}}$, and the communication network of just the nodes as $\mathfrak{Q}_{\text{N}}$. Note that both $\mathfrak{Q}_{\text{hybrid}}$ and $\mathfrak{Q}_{\text{N}}$ are connected networks.

*3.1.1. Voronoi contraction for collision avoidance* In addition to the maximum velocity and the restriction to $\mathcal{D}$, we impose a minimum distance requirement $\omega \in \mathbb{R}_{>0}$ between robots for collision avoidance. Consider the spatial locations $P = (p_1, \ldots, p_n)$ of the $n$ robotic agents at the $k$th timestep. Define $\Omega_i^{(k)} = (V_i(P))_{\text{ctn:}\omega/2} \cap \overline{B}(p_i, u_{\max})$, where $(V_i(P))_{\text{ctn:}\omega/2}$ denotes the $\frac{\omega}{2}$-contraction of $V_i(P)$. For each $j \neq i \in \{1, \ldots, n\}$, we have $\operatorname{dist}(\Omega_i^{(k)}, \Omega_j^{(k)}) \geq \omega$. Between timesteps $k$ and $k+1$, we restrict $\mathrm{R}_i$ to the region $\Omega_i^{(k)}$. Figure 2 shows an example in $\mathbb{R}^2$ of this set. The region of allowed movement of

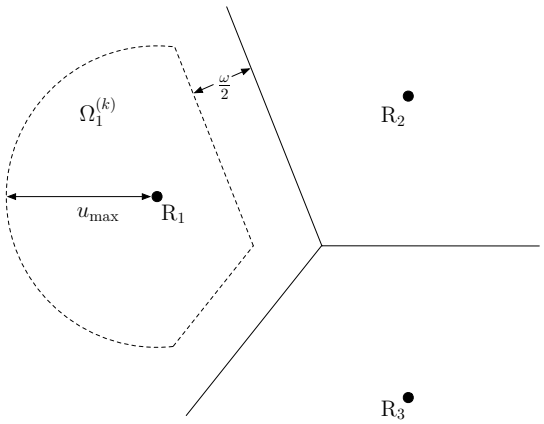



Figure 2. Example contraction region $\Omega_1^{(k)}$ (dashed) with Voronoi partition boundaries (solid) for comparison.

all robotic agents at timestep $k \in \mathbb{Z}_{\geq 0}$ is then the Cartesian product of the individual restrictions, i.e., $\Omega^{(k)} = \prod_{i=1}^n \Omega_i^{(k)} \subset (\mathbb{R}^d)^n$. Note that each $\Omega_i^{(k)}$ is the intersection of sets which are closed, bounded, and convex, and hence inherits these properties, which are in turn also inherited by $\Omega^{(k)}$. Further note that computation of the Voronoi partition generated by all robots is not required, only the partition boundaries between adjacent robots separated by less than $2u_{\max}$. We assume that the region $\Omega_i^{(k)}$ is reachable between steps $k$ and $k+1$. For vehicles with restricted dynamics, minimal modifications allow extension of these results to any nonempty, convex subset of $\Omega_i^{(k)}$.

### *3.2. The average variance as objective function*

For predictions over a spatiotemporal region, the average variance is a natural measure of uncertainty. Using Proposition 2.1, we define the average over the region of the posterior predictive variance,

$$\mathcal{A} = \varphi(Y, X) \int_{\mathcal{D}} \int_T \phi((s_0, t_0); X)\, dt_0\, ds_0. \tag{7}$$

Here, $Y \in (\mathbb{R}^n)^{k_{\max}}$ is a sequence of samples taken at discrete times $\{1, \dots, k_{\max}\}$, $k_{\max} \in \mathbb{Z}_{>0}$, at space-time locations $X \in (\mathcal{D}_e^n)^{k_{\max}}$. We take $T = [1, k_{\max}]$ to be the time interval of interest, indicating that the goal of the experiment is to develop an estimate of the space-time process over the entire duration. Other time intervals may be of interest in different experiments. Their use follows with minimal changes to the methods described here.

One would like to choose the sample locations that minimize $\mathcal{A}$. Since samples are taken sequentially, with each new set restricted to a region nearby the previous, and since the sigma mean depends on the actual values of the samples, one cannot simply optimize over $(\mathcal{D}_e^n)^{k_{\max}}$ a priori. Consider, instead, a greedy approach in which we use past samples to choose the positions for the next ones. At each timestep we choose the next locations to minimize the average posterior variance of the predictor given the data known so far. In Section 4, we develop a sequential formulation of the average posterior predictive variance and discuss its amenability to distributed implementation over $\mathfrak{Q}_{\text{hybrid}}$.

## 4. Distributed criterion for adaptive design

In this section we develop an optimality criterion to maximally reduce the average predictive variance at each timestep. We begin by introducing some notation that will help us make the discussion precise.

Let $Y^{(k)} \in \mathbb{R}^n$, $k \in \{1, \dots, k_{\max}\}$, denote the samples taken at timestep $k$, at space-time positions $X^{(k)} \in \mathcal{D}_e^n$. Let $Y^{(k_1:k_2)} = \big(Y^{(k_1)}, \dots, Y^{(k_2)}\big) \in \mathbb{R}^{n(k_2-k_1+1)}$, $k_1 < k_2$, denote the vector of samples taken over a range of timesteps, at positions $X^{(k_1:k_2)} = \big(X^{(k_1)}, \dots, X^{(k_2)}\big) \in \mathcal{D}_e^{n(k_2-k_1+1)}$. At step $k$, the samples $Y^{(1:k)}$ have already been taken. We are interested in choosing spatial locations, $P \in \Omega^{(k)}$, at which to take the next samples. To that end, let $X^{(1:k+1)} : \Omega^{(k)} \to \mathcal{D}_e^{n(k+1)}$ map a new set of spatial locations to the vector of spatiotemporal locations which will result if the $(k+1)$th samples are taken there, i.e., $X^{(1:k+1)}(P) = \big(X^{(1:k)}, (P, k+1)\big)$. The adaptive design approach is then to use the samples that minimize the average prediction variance *so far*,

$$\mathcal{A}^{(k)}(P) = \varphi\big(Y^{(1:k+1)}, X^{(1:k+1)}(P)\big) \int_{\mathcal{D}} \int_T \phi\big((s_0, t_0); X^{(1:k+1)}(P)\big)\, dt_0\, ds_0. \tag{8}$$

This sequential formulation of the problem allows us to use past measurements without worrying about the ones at steps after $k+1$. However, efficient distributed implementation still suffers from three major obstacles. First, the spatially distributed nature of the problem implies that not all sample locations are accessible to any given agent at any given time. Second, inversion of the $n(k+1) \times n(k+1)$ correlation

matrix, which grows with $k^2$, quickly becomes an unreasonable burden. Finally, the sigma mean also depends on the actual values of the samples at step $k+1$, which are not known until the measurements are taken. We handle these problems through a series of approximations, first to the conditional variance in Section 4.1, then to the sigma mean in Section 4.2, resulting in an approximation of the value of $\mathcal{A}^{(k)}$ which is both distributed in nature and computationally efficient.

### *4.1. Upper bound on conditional variance*

We seek an efficient approximation of the conditional variance term $\phi\big((s_0,t_0);X^{(1:k+1)}(P)\big)$ in (8). As noted in Remark 2.2, $\phi$ represents the direct effect of the sample locations on the predictive uncertainty (i.e., conditional on $\sigma^2$). The network of static nodes provides a convenient method for calculating the spatial average. The average over the entire region may simply be written as the sum of averages over each cell in the Voronoi partition generated by the static nodes. As those samples which are spatially near a given cell have the most influence on reducing the variance of predictions there, we consider using *local information only* in these regional calculations. Likewise, the interaction between current samples and those far in the past is minimal, and we restrict attention to recent timesteps to avoid the problem of growing complexity. The following result gives an approximation of the integrated conditional variance which may be calculated by $\mathfrak{Q}_{\mathrm{N}}$ *based on local information only*.

**Proposition 4.1 (Approximate integrated conditional variance)** *Let $X_j^{(k+1-\lfloor r_t\rfloor:k+1)}(P)$ denote an ordering of the set of past or current space-time locations correlated in space to $V_j(Q)$ and in time to $k+1$ such that*

$$i_{\mathbb{F}}\left(X_j^{(k+1-\lfloor r_t\rfloor:k+1)}(P)\right)=\left\{(s,t)\in i_{\mathbb{F}}\left(X^{(1:k+1)}(P)\right)\;|\;\mathrm{dist}(s,V_j(Q))<r_s \text{ and } k+1-t<r_t\right\}.$$

*Let $\phi_j^{(k)}:\mathcal{D}_e\times\Omega^{(k)}\to\mathbb{R}$ map a prediction location $x_0\in\mathcal{D}_e$ and a vector of potential spatial locations to sample $P\in\Omega^{(k)}$ to the conditional variance of a prediction made at $x_0$ using only the samples at space-time locations $X_j^{(k+1-\lfloor r_t\rfloor:k+1)}(P)$. Then the following holds,*

$$\int_{\mathcal{D}}\int_T\phi((s_0,t_0);X^{(1:k+1)}(P))\,dt_0\,ds_0\le\sum_{j=1}^m\int_{V_j(Q)}\int_T\phi_j^{(k)}((s_0,t_0);P)\,dt_0\,ds_0.$$

**Proof.** Note that although $X_j^{(k+1-\lfloor r_t\rfloor:k+1)}(P)$ is not unique, the invariance of the conditional variance to permutations of the sample locations ensures uniqueness of $\phi_j\big(x_0;P\big)$. The result follows from Proposition I.2 in Appendix I. ■

### *4.2. Approximate sigma mean*

In this section, we describe our approach to deal with the sigma mean term $\varphi(Y^{(1:k+1)}, X^{(1:k+1)}(P))$ in (8). Note that the effect of the sigma mean on prediction is indirect, and its value has the same influence on predictions regardless of the predictive location. As such, we do not use spatially local approximations as we did for the conditional variance. However, to avoid the problem of complexity growth, we use samples from only a subset of the timesteps. We discuss this next. Subsequently, we address the issue of unrealized sample values by using a generalized least squares estimate.

*4.2.1. Incorporating new data.* Here we consider approximating the value of $\varphi$ as calculated with samples from a subset of timesteps. There are various reasons for using different sample subsets depending on the field under study, the objectives of the experiment, and the desired accuracy of optimization. Since $\varphi$ is invariant under permutations of the sample vector, the specific ordering is irrelevant. Proposition 4.2 serves as the basis for choosing the samples to include in an approximation of the sigma mean. The proof follows from (9c) in Lemma I.3 in Appendix I.

**Proposition 4.2 (Approximate sigma mean)** *Let $Y_1 \in \mathbb{R}^{n_1}$ and $Y_2 \in \mathbb{R}^{n_2}$ denote two sample vectors of lengths $n_1, n_2 \in \mathbb{Z}_{>0}$, and let $Y = (Y_1, Y_2)$. Let $\varphi_2 = \mathrm{E}[\sigma^2|Y_2]$ denote the value of the sigma mean conditional on only the samples in $Y_2$, and $\varphi = \mathrm{E}[\sigma^2|Y]$ the value conditional on the whole sample vector $Y$. Then,*

$$\varphi = \varphi_2\Big(\frac{\nu + n_2 - 2}{\nu + n_1 + n_2 - 2} + \frac{(Y_1 - \mathrm{E}[Y_1|Y_2])^T \operatorname{Var}[Y_1|Y_2]^{-1}(Y_1 - \mathrm{E}[Y_1|Y_2])}{\nu + n_1 + n_2 - 2}\Big),$$

*where $\mathrm{E}[Y_1|Y_2]$ and $\operatorname{Var}[Y_1|Y_2]$ denote the conditional expectation and variance, respectively, of $Y_1$ given $Y_2$ (see Proposition I.1 in Appendix I).*

This result contains some important implications with respect to the optimization problem. First, if we use the full value of $\varphi(Y^{(1:k)}, X^{(1:k)}(P)) = \mathrm{E}[\sigma^2|Y^{(1:k)}]$ in our optimality metric at each timestep, and all steps are optimized with respect to this measure, the new information gained at later steps diminishes significantly, while the amount of effort required to glean that information increases. Second, the additional information added by including $Y_1$ is directly related to how well $Y_1$ may be estimated from $Y_2$.

These observations lead us to suggest the following three strategies for choosing samples to use in the approximation of the sigma mean.

**Exploration-exploitation:** The diminishing returns suggest using an exploration-exploitation approach [27]. Here a block of $t_{\text{blk}} \in \{1, \ldots, k_{\max}\}$ sample steps at the beginning of the experiment designates an exploration phase, during which the sigma mean is taken into account in the optimization. Subsequent iterations constitute an exploitation phase in which the sigma mean is treated as a fixed constant and we optimize only the conditional variance.

**Recent block:** The second strategy is to always use the most recent $t_{\text{blk}}$ sample steps. This increases the computational burden over the exploration-exploitation approach, but ensures that each step takes the unknown covariance into account in optimization.

**Block update:** A third choice is to use select blocks of $t_{\text{blk}}$ timesteps for scheduled updates of the sigma mean. Let $t_{\text{skip}} \in \mathbb{Z}_{>0}$ be a fixed number of timesteps to skip between updates. The advantage of this method in reducing computational complexity comes from an assumption that $t_{\text{skip}} > \lceil r_{\text{t}} \rceil - 1$, resulting in a block diagonal correlation matrix.

These three approximation methods trade off computational complexity for accuracy. The appropriateness of each method depends on the specific requirements of the scenario. In all three cases, the maximum size of the correlation matrix which must be inverted is $nt_{\text{blk}} \times nt_{\text{blk}}$. Furthermore, using Lemma I.3, it can be seen that this matrix inversion is the most computationally intensive part of calculating the sigma mean. Therefore, using any one of these three methods, we avoid computational complexities which grow out of proportion to the information gain. However, we still have the problem that the sigma mean includes sample values which have not been taken yet. We address this issue in the next section. To avoid unnecessary notation, we assume throughout the sequel that all available samples are used to calculate $\varphi^{(k)}$. To accommodate the approximations outlined in this section, it is necessary only to replace them with the subsampled ones, and modify the number of samples accordingly.

*4.2.2. Approximating unrealized sample values.* While seeking to optimize the $(k+1)$th set of measurements, we would like to incorporate the effect of their *locations* on the posterior variance, but the actual values have not yet been sampled. Our approach is to use a generalized least squares approximation of $Y^{(k+1)}$. We describe this in detail in the next result whose proof is in Appendix I.

**Proposition 4.3 (Generalized least squares estimate of sigma mean)** *Let $\hat{Y}_{LS}^{(k)} : \Omega^{(k)} \to \mathbb{R}^n$ map a vector of spatial locations $P \in \Omega^{(k)}$ to the generalized least squares estimate, based on the*

*sample vector $Y^{(1:k)}$, of a vector of samples to be taken at space-time positions $(P, k+1)$. Let $\hat{\varphi}^{(k+1)} : \Omega^{(k)} \to \mathbb{R}$ be defined by,*

$$\hat{\varphi}^{(k+1)}(P) = \frac{1}{\nu + n(k+1) - 2}\Big[q\nu + \beta_0^T \mathbf{K}_0^{-1}\beta_0 + \Upsilon^{(k)} - \big(\mathbf{K}_0^{-1}\beta_0 + \Gamma^{(k)}\big)^T \big(\mathbf{K}_0^{-1} + \mathbf{E}^{(k+1)}(P)\big)^{-1}\big(\mathbf{K}_0^{-1}\beta_0 + \Gamma^{(k)}\big)\Big], \quad \textit{where}$$

$$\Upsilon^{(k)} = (Y^{(1:k)})^T (\mathbf{K}^{(1:k)})^{-1} Y^{(1:k)}, \qquad \Gamma^{(k)} = \mathbf{F}^{(1:k)} (\mathbf{K}^{(1:k)})^{-1} Y^{(1:k)},$$

*and $\mathbf{E}^{(k+1)}(P)$ denotes the matrix $\mathbf{E}$ as calculated with space-time location vector $X^{(k+1)}(P) = \big(X^{(1:k)}, (P, k+1)\big)$. After the new samples, $Y^{(k+1)}$ have been taken at locations $(P, k+1)$, let $\overline{y}_{LS}^{(k)} : \Omega^{(k)} \to \mathbb{R}^n$ denote the estimation error, i.e., $\overline{y}_{LS}^{(k)}(P) = Y^{(k+1)} - \hat{Y}_{LS}^{(k)}(P)$. Then,*

$$\varphi^{(k+1)} = \frac{\varphi^{(k)}(\overline{y}_{LS}^{(k)}(P) - 2\overline{\mu}_{k+1|\leq k})^T \operatorname{Var}[Y^{(k+1)}|Y^{(1:k)}]^{-1}}{\nu + n(k+1) - 2}\overline{y}_{LS}^{(k)}(P) + \hat{\varphi}^{(k+1)}(P), \quad \textit{where}$$

$$\overline{\mu}_{k+1|\leq k} = (\mathbf{F}^{(k+1)} - \mathbf{F}^{(1:k)}(\mathbf{K}^{(1:k)})^{-1}\operatorname{Cor}[Y^{(1:k)}, Y^{(k+1)}])^T \times (\mathbf{E}^{(1:k)} + \mathbf{K}_0^{-1})^{-1}\left(\mathbf{F}^{(1:k)}(\mathbf{K}^{(1:k)})^{-1}Y^{(1:k)} + \mathbf{K}_0^{-1}\beta_0\right).$$

In other words, $\varphi^{(k+1)}$ may be estimated by $\hat{\varphi}^{(k+1)}(P)$. The estimation is exact if $Y^{(k+1)} = \hat{Y}_{\mathrm{LS}}^{(k)}(P)$.

**Remark 4.4 (Quality of estimation)** It should be pointed out here that the generalized least squares estimate is not the best guess of the value of the unrealized data. That would correspond to the conditional expectation given all past data. However, from Proposition 4.2, we can see that this choice would result in zeroing out the influence of the locations on the posterior mean of $\sigma^2$. Instead, we use an optimality criterion which accounts for this influence. •

### *4.3. The aggregate average prediction variance and its smoothness properties*

Building on the results from Sections 4.1 and 4.2, we define here the *aggregate average prediction variance* $\tilde{\mathcal{A}}^{(k)}$. Unlike $\mathcal{A}^{(k)}$, the function $\tilde{\mathcal{A}}^{(k)}$ may be computed efficiently in a distributed manner over the network $\mathfrak{Q}_{\text{hybrid}}$. The following result is a direct consequence of Propositions 4.1 and 4.3.

**Proposition 4.5. (Spatiotemporal approximation for distributed implementation)** *Let $\tilde{\mathcal{A}}_j^{(k)} : \Omega^{(k)} \to \mathbb{R}$ be defined by*

$$\tilde{\mathcal{A}}_j^{(k)}(P) = \hat{\varphi}^{(k+1)}(P) \int_{V_j(Q)} \int_T \phi_j^{(k)}\left((s,t), P\right) dt\, ds.$$

*Under the assumption that the error term from Proposition 4.3 satisfies,*

$$\lim_{k\to\infty} \frac{\varphi k(\overline{y}_{LS}^{(k)}(P) - 2\overline{\mu}_{k+1|\leq k})^T \operatorname{Var}[Y^{(k+1)}|Y^{(1:k)}]^{-1}}{\nu + n(k+1) - 2} \overline{y}_{LS}^{(k)}(P) = 0,$$

*then* $\tilde{\mathcal{A}}^{(k)}(P) = \sum_{j=1}^{m} \tilde{\mathcal{A}}_j^{(k)}(P)$ *satisfies* $\lim_{k\to\infty} \tilde{\mathcal{A}}^{(k)}(P) \geq \lim_{k\to\infty} \mathcal{A}^{(k)}(P)$.

Next, we characterize the smoothness properties of $\tilde{\mathcal{A}}^{(k)}$. Let $\nabla_{il}$ denote the partial derivative with respect to $p_{il}$, the $l$th spatial component of the spatial position of $\mathrm{R}_i$. We denote by $\nabla_i$ the partial derivative with respect to $p_i$, i.e., $\nabla_i = (\nabla_{i1}, \ldots, \nabla_{id})^T$. Thus the gradient of $\tilde{\mathcal{A}}^{(k)}$ at location $P$ may be represented as the $n\,d$-dimensional vector $\left(\nabla_1^T \tilde{\mathcal{A}}^{(k)}(P), \ldots, \nabla_n^T \tilde{\mathcal{A}}^{(k)}(P)\right)^T$. Given a matrix $A$, denote by $\nabla_{il} A$ the component-wise partial derivative of $A$. The proof of the next result amounts to a careful bookkeeping of the smoothness properties of the various ingredients in the expressions.

**Lemma 4.6 (Gradient of conditional variance)** *If $f_1, \ldots, f_p$ are $C^1$ with respect to the spatial position of their arguments, then the map $P \mapsto \phi_j^{(k)}(x_0, P)$ is $C^1$ on $\Omega^{(k)}$ with partial derivative,*

$$\begin{aligned}
\nabla_{il}\phi_j^{(k)}(x_0, P) &= -2\mathbf{k}^T\mathbf{K}^{-1}\nabla_{il}\mathbf{k} + \mathbf{k}^T\mathbf{K}^{-1}\nabla_{il}\mathbf{K}\mathbf{K}^{-1}\mathbf{k} \\
&\quad - \xi_0^T \left(\mathbf{K}_0^{-1} + \mathbf{E}\right)^{-1} \nabla_{il}\mathbf{E} \left(\mathbf{K}_0^{-1} + \mathbf{E}\right)^{-1} \xi_0 + 2\xi_0^T \left(\mathbf{K}_0^{-1} + \mathbf{E}\right)^{-1} \nabla_{il}\xi_0, \text{ with} \\
\nabla_{il}\xi_0 &= -\nabla_{il}\mathbf{F}\mathbf{K}^{-1}\mathbf{k} - \mathbf{F}\mathbf{K}^{-1}\nabla_{il}\mathbf{k} + \mathbf{F}\mathbf{K}^{-1}\nabla_{il}\mathbf{K}\mathbf{K}^{-1}\mathbf{k}, \\
\nabla_{il}\mathbf{E} &= \nabla_{il}\mathbf{F}\mathbf{K}^{-1}\mathbf{F}^T + \mathbf{F}\mathbf{K}^{-1}\nabla_{il}\mathbf{F}^T - \mathbf{F}\mathbf{K}^{-1}\nabla_{il}\mathbf{K}\mathbf{K}^{-1}\mathbf{F},
\end{aligned}$$

*where the matrices* $\mathbf{K}$, $\mathbf{E}$, *and* $\mathbf{F}$ *and the vectors* $\mathbf{k}$, *and* $\xi_0$ *are calculated from the space-time location subvector,* $X_j^{(k+1-\lfloor r_t \rfloor:k+1)}(P)$.

*If, in addition, the partial derivatives of $f_1, \ldots, f_p$ are $C^1$ with respect to the spatial position of their arguments, then the map $P \mapsto \nabla_i \phi_j^{(k)}(x_0, P)$ is globally Lipschitz on $\Omega^{(k)}$.*

It is worth noting that the matrix $\nabla_{il}\mathbf{F}$ is nonzero only in column $i$. The matrix $\nabla_{il}\mathbf{K}$ is nonzero only in row and column $i$. Additionally, due to the finite correlation range in space and time, only those elements corresponding to correlation with other measurement locations $x = (s, t)$ which satisfy $\|p_i - s\| \leq r_\mathrm{s}$ and $t > k + 1 - r_\mathrm{t}$ are nonzero.

Note that the value of $\hat{\varphi}^{(k+1)}(P)$ depends on $P$ only through the matrix $\mathbf{E}^{(k+1)}(P)$, whose partial derivative is analogous to that of $\mathbf{E}$ in Lemma 4.6. If subsampling is used to approximate $\varphi$ per Section 2.1, we set $\nabla\hat{\varphi}^{(k+1)}(P) = \mathbf{0}$ for those steps which are skipped. Here we assume no subsampling for notational convenience. This leads us to the following result.

**Lemma 4.7 (Gradient of sigma mean)** *If $f_1, \dots, f_p$ are $C^1$ with respect to the spatial position of their arguments, then $\hat{\varphi}^{(k+1)}$ is $C^1$ on $\Omega^{(k)}$ with partial derivative,*

$$\nabla_{il}\hat{\varphi}^{(k+1)}(P) = -\frac{\Psi(P)^T \nabla_{il}\mathbf{E}^{(k+1)}(P)\,\Psi(P)}{\nu + n(k+1) - 2}, \quad \textit{where}$$
$$\Psi(P) = \left(\mathbf{K}_0^{-1} + \mathbf{E}^{(k+1)}(P)\right)^{-1}\left(\mathbf{K}_0^{-1}\beta_0 + \Gamma^{(k)}\right).$$

*Additionally, if the partial derivatives of $f_1, \dots, f_p$ are $C^1$ with respect to the spatial position of their arguments, the gradient $\nabla\hat{\varphi}^{(k+1)}$ is globally Lipschitz on $\Omega^{(k)}$.*

We are now ready to state the smoothness properties of $\tilde{\mathcal{A}}^{(k)}$ and provide an explicit expression for its gradient. This is a direct consequence of the lemmas above.

**Proposition 4.8 (Gradient of approximate average variance)** *If $f_1, \dots, f_p$ are $C^1$ with respect to the spatial position of their arguments, then $\tilde{\mathcal{A}}^{(k)}$ is $C^1$ on $\Omega^{(k)}$ with partial derivative,*

$$\nabla_i\tilde{\mathcal{A}}^{(k)}(P) = \hat{\varphi}^{(k+1)}(P)\int_{V_j(Q)}\int_T \nabla_i\phi_j^{(k)}\left((s,t),P\right)\,dt\,ds + \nabla_i\hat{\varphi}^{(k+1)}(P)\int_{V_j(Q)}\int_T \phi_j^{(k)}\left((s,t),P\right)\,dt\,ds.$$

*Additionally, if the partial derivatives of $f_1, \dots, f_p$ are $C^1$ with respect to the spatial position of their arguments, the gradient $\nabla\tilde{\mathcal{A}}^{(k)}$ is globally Lipschitz on $\Omega^{(k)}$.*

### *4.4. Distributed computation of aggregate average prediction variance and its gradient*

Here, we substantiate our assertion that the aggregate average prediction variance and its gradient introduced in Section 4.3 are distributed over the network $\mathfrak{Q}_{\text{hybrid}}$. Since $\mathcal{V}(Q)$ is a partition of the physical space, we may partition all sample locations spatially by region. Thus for each $(s,t) \in i_{\mathbb{F}}(X)$, there is exactly one $j \in \{1,\dots,m\}$ such that $s \in V_j(Q)$. In order for the network to calculate $\tilde{\mathcal{A}}^{(k)}$ and its gradient at $P$, it is sufficient for $\mathrm{N}_j$ to compute $\tilde{\mathcal{A}}_j^{(k)}$ and $\nabla_i\tilde{\mathcal{A}}_j^{(k)}$ for each robot in $V_j(Q)$. Then $\tilde{\mathcal{A}}^{(k)}$ may be calculated via discrete-time average consensus (cf. Section 2.3), while $\nabla_i\tilde{\mathcal{A}}^{(k)}$ may be calculated from information local to $\mathrm{R}_i$. From Propositions 4.5 and 4.8, it can be seen that the calculation of $\tilde{\mathcal{A}}_j^{(k)}$ and $\nabla_i\tilde{\mathcal{A}}_j^{(k)}$ requires only local information in addition to the (global) values of $\hat{\varphi}^{(k+1)}(P)$ and $\nabla_i\hat{\varphi}^{(k+1)}(P)$. Let us explain how these two quantities can be calculated.

For $i \in \{1,\dots,nk\}$, let $x_i^{(1:k)}$ and $y_i^{(1:k)}$ denote the $i$th element of the vector $X^{(1:k)}$ and $Y^{(1:k)}$, respectively. Let $\mathrm{I}_{\text{Local}}^{(k)} : \mathbb{Z}_{>0} \to \mathbb{F}(\mathbb{Z}_{>0})$ map the index of the node to the set of indices of samples

whose spatial position lies inside its Voronoi cell, and whose time element is correlated to time $k+1$,

$$\mathrm{I}^{(k)}_{\mathrm{Local}}(j) = \left\{ i \in \{1,\dots,nk\} \mid x_i^{(1:k)} = (s,t) \text{ and } s \in V_j(Q) \right\}$$

With a slight abuse of notation, define $\mathrm{I}^{(k+1)}_{\mathrm{Local}}(j,P)$ to be the equivalent set of indices into the full vector of space-time measurement locations, $X^{(k+1)}(P)$.

In the following results, we assume that a desired level of accuracy is known a priori to all nodes so that an execution of the JOR or average consensus algorithms have a finite termination criterion. Unless stated otherwise, the executions of these algorithms may take place in serial or parallel. Our first result illustrates how the network can calculate the terms in $\hat{\varphi}^{(k+1)}(P)$ which do not depend on $P$.

**Proposition 4.9 (Distributed calculations without $P$)** *for $j \in \{1,\dots,m\}$, assume that $\mathrm{N}_j$ knows $x_i^{(1:k)}, y_i^{(1:k)}$ for each $i \in I^{(k)}_{Local}(j)$. After $p+1$ executions of the JOR algorithm and two subsequent consensus algorithms, $\mathrm{N}_j$ has access to,*

#1: *element $i$ of $(\mathbf{K}^{(1:k)})^{-1}Y^{(1:k)} \in \mathbb{R},\ i \in I^{(k)}_{Local}(j)$ via JOR;*

#2: *col$_i$ $\left(\mathbf{F}^{(1:k)}(\mathbf{K}^{(1:k)})^{-1}\right) \in \mathbb{R}^p,\ i \in I^{(k)}_{Local}(j)$ via JOR;*

#3: *$\Gamma^{(k)} \in \mathbb{R}^p$ via consensus;*

#4: *$\Upsilon^{(k)} \in \mathbb{R}^p$ via consensus.*

**Proof.** Under the assumptions on $\mathfrak{Q}_{\mathrm{hybrid}}$, the matrix $\mathbf{K}^{(1:k)}$, satisfies the requirements of the distributed JOR algorithm. The results here build on this fact and the connectedness of $\mathfrak{Q}_{\mathrm{N}}$ (which allows for consensus). ■

Next, we describe calculations that the network can execute when robotic agents are at locations $P$.

**Proposition 4.10 (Distributed calculations with $P$)** *Given $P \in \Omega^{(k)}$, assume that $\mathrm{N}_j$, for $j \in \{1,\dots,m\}$, knows $x_i^{(1:k)}$ for each $i \in I^{(k+1)}_{Local}(j,P)$ and the results of Proposition 4.9. Let $\mathbf{F}^{(k+1)}(P)$ denote the matrix of basis functions evaluated at locations $X^{(1:k+1)}(P)$. After $p$ executions of JOR and $\frac{p(p+1)}{2}$ executions of consensus algorithms, $\mathrm{N}_j$ has access to,*

#5: *col$_i$ $\left(\mathbf{F}^{(k+1)}(P)(\mathbf{K}^{(k+1)}(P))^{-1}\right) \in \mathbb{R}^p,\ i \in I^{(k+1)}_{Local}(j,P)$ via JOR;*

#6: *$\mathbf{E}^{(k+1)}(P) \in \mathbb{R}^{p\times p}$ via consensus.*

*After these computations, $\mathrm{N}_j$ can calculate $\nabla_{il}\mathbf{E}^{(k+1)}$ for $l \in \{1,\dots,d\}$, and consequently $\hat{\varphi}^{(k+1)}(P)$ and $\nabla_i\hat{\varphi}^{(k+1)}(P)$ for each robot in $\{i \in \{1,\dots,n\} \mid p_i \in V_j(Q)\}$.*

**Proof.** The matrix $\mathbf{K}^{(k+1)}(P)$ satisfies the requirements of the distributed JOR algorithm by the assumptions on $\mathfrak{Q}_{\text{hybrid}}$. The itemized results follow from this, and the symmetry of the matrix $\mathbf{E}^{(k+1)}(P)$. The calculation of $\hat{\varphi}^{(k+1)}$ and partial derivatives follow from Lemmas I.3 and 4.7. ■

## 5. Distributed optimization of the aggregate average predictive variance

Here we present a distributed algorithm which is guaranteed to converge to a stationary point of $\tilde{\mathcal{A}}^{(k)}$ on $\Omega^{(k)}$. The DISTRIBUTED PROJECTED GRADIENT DESCENT ALGORITHM in Table V allows the network of static nodes and mobile agents to find local minima of $\tilde{\mathcal{A}}^{(k)}$ on $\Omega^{(k)}$. At timestep $k$, the nodes follow a gradient descent algorithm, defining a sequence of configurations, $\{P_l^\dagger\}$, $l \in \mathbb{Z}_{>0}$, with $P_1^\dagger = P^{(k)} \in \Omega^{(k)}$, the vector of current spatial locations of the robotic agents and

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

where $\alpha$ is chosen via the DISTRIBUTED LINE SEARCH ALGORITHM outlined in Table IV. The DISTRIBUTED LINE SEARCH ALGORITHM is a distributed version of the LINE SEARCH ALGORITHM from Table II. The maximum stepsize, $\alpha_{\max} \in \mathbb{R}_{>0}$, is designed to ensure that all robots with nonzero partial derivatives can move the maximum distance.

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 $\text{N}_j$ transmits $p_i(k+1)$ to robot $\text{R}_i$, which then moves to the location between timesteps. Note that although each robot may be sending position and sample information to multiple nodes, the approximate average prediction variance is calculated *within the Voronoi cell*. As the Voronoi cells do not overlap, there is no problem with information repetition.

The following result describes some nice properties of the DISTRIBUTED PROJECTED GRADIENT DESCENT ALGORITHM. Its proof is a direct result of the construction of the algorithm and the fact that it is equivalent to a centralized projected gradient descent.

**Proposition 5.1. (Properties of the DISTRIBUTED PROJECTED GRADIENT DESCENT ALGORITHM)** *The* DISTRIBUTED PROJECTED GRADIENT DESCENT ALGORITHM *is distributed over the network $\mathfrak{Q}_{hybrid}$. Moreover, if the partial derivatives of $f_1, \ldots, f_p$ are $C^1$ with respect to the spatial position of their arguments, 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 $\Omega^{(k)}$.*

**Name:** DISTRIBUTED LINE SEARCH ALGORITHM

**Goal:** Compute step size for projected gradient descent of $\tilde{\mathcal{A}}^{(k)}$

**Input:** Configuration, $P = (p_1, \ldots, p_n) \in \Omega^{(k)}$

**Assumes:** (i) Connected network of static nodes

(ii) $\mathrm{N}_j$ 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) $\|\nabla_i \tilde{\mathcal{A}}^{(k)}(P)\| \neq 0$ for at least one $i \in \{1, \ldots, n\}$

(iv) $\mathrm{N}_j$ knows items #3 and #4 from Proposition 4.9

(v) Shrinkage factor $\tau$ and tolerance $\theta \in (0, 1)$ known a priori by all static nodes

**Uses:** (i) Projection of next set of locations on $\Omega_i$,

$$P_j'(\alpha, P) = \left\{ \mathrm{proj}_{\Omega_i}(p_i + \alpha \nabla_i \tilde{\mathcal{A}}(P)), \text{for each } i \text{ such that } \mathrm{dist}\,(p_i, V_j(Q)) \leq r_{\mathrm{s}} + u_{\max} + \omega \right\}.$$

(ii) Total distance traveled by robots entering $V_j(Q)$,

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

**Output:** Step size $\alpha \in \mathbb{R}$

Initialization

1: $\mathrm{N}_1, \ldots, \mathrm{N}_m$ calculate $\alpha_{\max} = \dfrac{u_{\max}}{\min\{\|\nabla_i \tilde{\mathcal{A}}(P)\| \,|\, \|\nabla_i \tilde{\mathcal{A}}(P)\| \neq 0\}}$ via *maximum consensus*

For $j \in \{1, \ldots, m\}$, node $\mathrm{N}_j$ executes concurrently

1: $\alpha := \alpha_{\max}$

2: **repeat**

3: calculates $\mathrm{dist}_j\,(\alpha, P)^2$

4: calculates $\hat{\varphi}^{(k+1)}\left(P_j'(\alpha, P)\right)$ according to Proposition 4.10

5: calculates $\tilde{\mathcal{A}}_j^{(k)}\left(P_j'(\alpha, P)\right)$

6: execute consensus algorithm to calculate the following:

$$\tilde{\mathcal{A}}^{(k)}\left(P'(\alpha, P)\right) = \sum_{j=1}^{m} \tilde{\mathcal{A}}_j^{(k)}\left(P_j'(\alpha, P)\right)$$

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

7: $\varpi := \frac{\theta}{\alpha} \|P - P'(\alpha, P)\|^2 + \tilde{\mathcal{A}}^{(k)}(P'(\alpha, P)) - \tilde{\mathcal{A}}^{(k)}(P)$

8: **if** $\varpi > 0$ **then**

9: $\alpha := \alpha\tau$

10: **until** $\varpi \leq 0$

Table IV. DISTRIBUTED LINE SEARCH ALGORITHM.

| | |
|---|---|
| **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<br>(ii) Static nodes deployed over $\mathcal{D}$ such that $r_{\text{com}} \geq \max_{i\in\{1,\dots,m\}} \{\text{CR}(V_i(Q))\} + r_{\text{s}} + u_{\max}$, robotic agents in initial configuration $P^{(1)} \in \Omega^{(k)}$<br>(iii) Line search shrinkage factor $\tau$ and tolerance value $\theta \in (0,1)$ known a priori by all nodes<br>(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 $P_{\text{cur}}$, respectively $P_{\text{next}}$ to hold the configuration at the current, respectively next step of the gradient projection algorithm. For ease of exposition, we use global notation although $\text{N}_j$ 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 $\text{N}_j$ executes: | Meanwhile, robot $\text{R}_i$ executes: |
|---|---|
| sets $R_{\text{cov}}(j) := \{\text{R}_i \mid \text{dist}(p_i(k), V_j(Q)) \leq r_{\text{s}}\}$ | sets $S_{\text{cov}}(i) := \{\text{N}_j \mid \text{dist}(p_i(k), V_j(Q)) \leq r_{\text{s}}\}$ |
| | takes measurement $y_i^{(k)}$ at $p_i(k)$ |
| collects $y_i^{(k)}$ and $p_i(k)$ from $\text{R}_i$ for each $i \in R_{\text{cov}}(j)$ | sends $y_i^{(k)}$ and $p_i(k)$ to nodes in $S_{\text{cov}}(i)$ |
| computes $\tilde{\mathcal{A}}_j^{(k)}(P^{(k)})$, then $\tilde{\mathcal{A}}^{(k)}(P^{(k)})$ by consensus | |
| sets $P_{\text{next}} := P^{(k)}$ | |
| **repeat** | **repeat** |
| sets $P_{\text{cur}} := P_{\text{next}}(j)$ and calculates $-\nabla\tilde{\mathcal{A}}_j^{(k)}\|_{P_{\text{cur}}}$ | |
| transmits $\nabla_i\tilde{\mathcal{A}}_j^{(k)}(P_{\text{cur}})$ to robots in $R_{\text{cov}}(j)$ | receives $\nabla_i\tilde{\mathcal{A}}_j^{(k)}(P^{(k)})$ from nodes in $S_{\text{cov}}(i)$ |
| | calculates sum $\nabla_i\tilde{\mathcal{A}}^{(k)}(P^{(k)})$ |
| collects sum $\nabla_i\tilde{\mathcal{A}}^{(k)}(P_{\text{cur}})$ from robots in $R_{\text{cov}}(j)$ | sends $\nabla_i\tilde{\mathcal{A}}^{(k)}(P^{(k)})$ to all nodes in $S_{\text{cov}}(i)$ |
| 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 $\|\tilde{\mathcal{A}}^{(k)}(P_{\text{next}}) - \tilde{\mathcal{A}}^{(k)}(P_{\text{cur}})\|$ | |
| **until** $\|\tilde{\mathcal{A}}^{(k)}(P_{\text{next}}) - \tilde{\mathcal{A}}^{(k)}(P_{\text{cur}})\| = 0$ | **until** receives finish marker from any node |
| sets $P^{(k+1)} := P_{\text{next}}$, sends position to robots in $V_j(Q)$ | receives location $p_i(k+1)$ and moves to it |

Table V. DISTRIBUTED PROJECTED GRADIENT DESCENT ALGORITHM.

The proposed algorithm is robust to failures in the *mobile* agents. If an agent stops sending position updates, 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. With minor modifications, the algorithm could be made robust to a certain number of node failures as well. However, this would require larger communication radius and extra storage (essentially having each node keep track of the sample locations stored by its Voronoi neighbors).

**Remark 5.2 (Extension to relative positioning)** It is interesting to observe that, due to the fact that the actual positions of samples are only required in a local context, our algorithm can also be implemented in a robotic network with relative positioning. The only requirements are the following: that each node can calculate the mean basis function for all local samples; that each node can calculate the correlations between pairs of local samples and that neighboring nodes can agree on the ordering of those samples within the global matrix. These modifications would not impact the convergence properties of the algorithm. •

### *5.1. Complexity analysis*

Here we examine in detail the complexity of the DISTRIBUTED PROJECTED GRADIENT DESCENT ALGORITHM in terms of the number of robotic agents and static nodes. For reference, we compare it against a centralized strategy that uses all-to-all broadcast and global information, and does not take advantage of the distributed nature of the problem. In order to avoid complexities which grow unbounded with $k$, we assume here that one of the approximation methods is employed for calculating $\varphi$ as outlined in Section 4.2.1. For the purposes of complexity, it does not matter which method is used, only that the size of the matrix which must be inverted at any step is bounded by $nt_{\text{blk}}$, and thus does not depend on $k$. Let $n_{\text{c}}^{(k)} \in \{1, \ldots, nt_{\text{blk}}\}$ denote the number of samples used in the *current* block for approximating $\varphi$ at step $k$, and let $\mathbf{K}_{\text{c}}^{(k)}$ denote the correlation matrix of those samples.

Given that the DISTRIBUTED PROJECTED GRADIENT DESCENT ALGORITHM is sequential, and designed to run for a fixed number of timesteps, we are concerned here with complexities involved in performing a single step. Below, where we refer to complexity notions over multiple iterations of an algorithm, we are considering the nested algorithms such as JOR, or consensus, which run during a single step of the DISTRIBUTED PROJECTED GRADIENT DESCENT ALGORITHM.

We examine the algorithm performance against the following notions of complexity, see [38, 39],

**Communication complexity:** the maximum number of bits transmitted over all (directed) communication channels between nodes in the network over the course of the algorithm;

**Time complexity:** the maximum number of iterations to completion of the algorithm times the maximum number of bits sent over any channel during one iteration;

**Space complexity:** the total number of bits for which space may be required *by a single node at any given time*.

We consider the complexity of the algorithms in terms of the number of agents, $n$, and the number of nodes, $m$, independently. We use the well-known *Bachmann-Landau* notation for upper and lower bounds, see e.g., [40]. Where rates of convergence of iterative methods depend on a desired level of accuracy, $\epsilon$, we use the notation $O_\epsilon$, instead of the standard $O$, to emphasize the dependence of bounds on the accuracy. Throughout the section, we make the following assumption on the diameter, degree, and number of edges of the communication graph $\mathfrak{Q}_{\mathrm{N}}$ of the network of static nodes.

**Regularity Assumption-** We assume that the group of static nodes is regular in the sense that the following three bounds are satisfied as $m$ increases:

$$\mathrm{diam}_{\mathfrak{Q}_{\mathrm{N}}} \in \Theta(\sqrt[d]{m}) \qquad \mathrm{Ed}_{\mathfrak{Q}_{\mathrm{N}}} \in \Theta(m) \qquad \deg_{\mathfrak{Q}_{\mathrm{N}}} \leq \deg_{\max} \in \mathbb{R}_{>0}.$$

**Remark 5.3 (Network assumptions are reasonable)** In two and three dimensions, the maximum diameter requirement has been shown to be consistent with a hexagonal grid network [41, 42], which is also consistent (in terms of number of neighbors) with the average case for large Voronoi networks [43]. The requirement of bounded degree is also satisfied by a hexagonal grid. The total number of edges is half the sum of the number of neighbors over all nodes, so bounded degree yields $\mathrm{Ed}_{\mathfrak{Q}_{\mathrm{N}}} \propto m$. •

We are now ready to characterize the complexities of our algorithms.

**Proposition 5.4. (Average consensus complexity)** *Let $b = (b_1, \ldots, b_m)^T \in \mathbb{R}^m$ denote a vector distributed across $\mathfrak{Q}_{\mathrm{N}}$ in the sense that $\mathrm{N}_j$ knows $b_j$ for each $j \in \{1, \ldots, m\}$. The discrete time consensus algorithm to calculate $\frac{b^T b}{m}$ to an accuracy of $\epsilon$ has communication complexity in $O_\epsilon\left(m^2 \sqrt[d]{m}\right)$, time complexity in $O_\epsilon\left(m \sqrt[d]{m}\right)$, and space complexity in $O_\epsilon(1)$.*

**Proof.** Each node sends a single message to each neighbor at each step, so the time complexity is bounded by the number of iterations to completion. The error at iteration $t$ is $e_{\mathrm{ave}}(t) = \|w_{\mathrm{ave}}(t) - w\|$,

with $w$ the $m$-vector whose elements are all $b^T b$, and $w_{\text{ave}}(t)$ the vector of current approximate values. This is bounded as $e_{\text{ave}}(t) \leq \left(1 - \frac{4}{m \operatorname{diam}_{\mathfrak{Q}_\text{N}}}\right)^t e_{\text{ave}}(0)$, where we use [44, Equation (6.10)] to lower bound the algebraic connectivity of $\mathfrak{Q}_\text{N}$. Thus the number of steps required to guarantee error less than $\epsilon$ is bounded by $t^*_{\text{ave}} \in O_\epsilon\left(-\log^{-1}\left(1 - \frac{4}{m \operatorname{diam}_{\mathfrak{Q}_\text{N}}}\right)\right)$. Applying the bound on the growth of the network diameter and replacing the logarithm with the series representation for large $m$, we deduce $t^*_{\text{ave}} \in O_\epsilon\left(m \sqrt[d]{m}\right)$. At each iteration, each node stores a single value for each neighbor, and a constant number of other values. Thus the space complexity is bounded by $\deg_{\mathfrak{Q}_\text{N}}$, which is in $O_\epsilon(1)$ by assumption. Finally, the communication complexity is bounded by a single message over each channel at each iteration. The total number of such messages from each node is bounded by a constant. ■

Since $\tilde{\mathcal{A}}^{(k)}$ uses only measurements correlated in time, the size of the matrices and vectors is limited to a constant multiple of $n$. The next result summarizes the complexity of the leader election computation, see [39] for a proof.

**Proposition 5.5. (Leader election complexity)** *The leader election algorithm may be run on $\mathfrak{Q}_\text{N}$ to calculate the quantity $\max_{i \in \{1,\ldots,n_c^{(k)}\}} \sum_{j=1}^{n_c^{(k)}} [\mathbf{K}_c^{(k)}]_{ij}$, with communication complexity in $O\left(m \sqrt[d]{m}\right)$, time complexity in $O\left(\sqrt[d]{m}\right)$, and space complexity in $O(1)$.*

For the algorithms considered next, the distribution of samples defines two different regimes for complexity. We consider both the worst case and the average based on a uniform distribution.

**Proposition 5.6. (JOR complexity)** *Assume that there is some constant $\varpi_\lambda \in (0,1)$, known a priori, such that $\lambda_{min}(\mathbf{K}_c^{(k)}) > \varpi_\lambda$. Regarding the sparsity of $\mathbf{K}_c^{(k)}$, assume that any one sample is correlated to at most $N_{cor} \in \mathbb{Z}_{>0}$ others, and that, for any $j \in \{1,\ldots,m\}$, the number of samples in $\mathcal{D} \setminus V_j(Q)$ which are correlated to samples in $V_j(Q)$ is upper bounded by a constant, $N_{msg} \in \mathbb{Z}_{>0}$. Let $b = (b_1, \ldots, b_{n_c^{(k)}})^T \in \mathbb{R}^{n_c^{(k)}}$ be distributed on the network of nodes in the sense that if $\mathrm{N}_j$ knows $col_i(\mathbf{K}_c^{(k)})$, then $\mathrm{N}_j$ knows $b_i$. Using the distributed JOR algorithm, the network may calculate $(\mathbf{K}_c^{(k)})^{-1} b$ to accuracy $\epsilon$ with communication complexity in $O_\epsilon(m \sqrt[d]{m})$, time complexity in $O_\epsilon(\sqrt[d]{m})$, and space complexity in $O_\epsilon(n)$ worst case, $O_\epsilon(\frac{n}{m})$ average case.*

**Proof.** The first step of the JOR algorithm is to calculate the relaxation parameter. For correlation matrices, Appendix II describes a near optimal parameter in the sense of minimizing the completion time. Using two leader election algorithms, the network calculates $\beta = \max_{i \neq j \in \{1,\ldots,n\}} [\mathbf{K}_\text{c}^{(k)}]_{ij}$ and $\alpha = \max_{i \in \{1,\ldots,n\}} \sum_{j \neq 1}^{n} [\mathbf{K}_\text{c}^{(k)}]_{ij}$. The relaxation parameter is then $h^* = \frac{2}{2+\alpha-\beta}$. The time

complexity of the JOR algorithm can be broken down into the maximum number of messages any node sends over any channel times the number of iterations. The number of messages $\mathrm{N}_j$ will send is equal to the number of nonzero off-diagonal entries $[\mathbf{K}_{\mathrm{c}}^{(k)}]_{ii'}$, $i \neq i'$, where $s_i \in V_j(Q)$ and $s_{i'} \in V_{j'}(Q)$, with $j \neq j'$. By assumption, this number is bounded by $N_{\mathrm{cor}}$. The error at iteration $t$ may be written $e_{\mathrm{JOR}}(t) = \|w_{\mathrm{JOR}}(t) - (\mathbf{K}_{\mathrm{c}}^{(k)})^{-1}b\|$, where $w_{\mathrm{JOR}}(t)$ is the vector of current approximate values. From [36], one has $e_{\mathrm{JOR}}(t) \leq \left(\rho\big(\boldsymbol{I} - h\mathbf{K}_{\mathrm{c}}^{(k)}\big)\right)^t e_{\mathrm{JOR}}(0)$, where $\boldsymbol{I}$ is the $n_{\mathrm{c}}^{(k)} \times n_{\mathrm{c}}^{(k)}$ identity matrix. By Proposition II.4, we have $0 \leq \rho\left(\boldsymbol{I} - h\mathbf{K}\right) < 1 - h^*\varpi_\lambda$. The assumption of sparsity and the fact that $\mathbf{K}_{\mathrm{c}}^{(k)}$ is a correlation matrix give us $0 \leq \alpha < N_{\mathrm{msg}}$, and $0 \leq \beta < 1$, which results in $1-h^*\varpi_\lambda < 1-\frac{2\varpi_\lambda}{2+N_{\mathrm{msg}}}$. Since both $\varpi_\lambda$ and $N_{\mathrm{msg}}$ are positive, we have $1-\frac{2\varpi_\lambda}{2+N_{\mathrm{msg}}} < 1$. Thus the number of iterations required to reach error $\epsilon$ is upper bounded by $t^* = (e(0) - e^*)\left(-\log^{-1}\left(1 - \frac{2\varpi_\lambda}{2+N_{\mathrm{msg}}}\right)\right)$ in $O_\epsilon(1)$. The time complexity is dominated by the time complexity of the leader election algorithm outlined in Proposition 5.5. For space complexity, we note that the maximum number of samples in a given Voronoi cell is bounded by $n_{\mathrm{c}}^{(k)}$, while the average number is $\frac{n_{\mathrm{c}}^{(k)}}{m}$. The space complexity is dominated by the requirement to store vectors of length given by the number of samples in the cell, and the same number of rows of $\mathbf{K}_{\mathrm{c}}^{(k)}$. For communication complexity, the overall algorithm requires a maximum of one message to be sent per nonzero off-diagonal entry in $\mathbf{K}_{\mathrm{c}}^{(k)}$, each iteration, plus the number of messages required for the leader election. ■

**Remark 5.7 (Interpretation of sparsity assumptions)** The assumptions on the sparsity of $\mathbf{K}_{\mathrm{c}}^{(k)}$ in Proposition 5.6 have the following interpretation: samples do not cluster in space as measured with respect to the distribution of the Voronoi cells and their size relative to the correlation range. •

The above results allow us to characterize the complexities of the DISTRIBUTED PROJECTED GRADIENT DESCENT ALGORITHM.

**Proposition 5.8. (Complexity of the DISTRIBUTED PROJECTED GRADIENT DESCENT ALGORITHM)** *Under the assumptions of Table V, the* DISTRIBUTED PROJECTED GRADIENT DESCENT ALGORITHM *may be completed within tolerance $\epsilon$ with communication complexity in $O_\epsilon(m^2\sqrt[d]{m})$, time complexity in $O_\epsilon(m\sqrt[d]{m})$, and space complexity in $O_\epsilon(n^2)$ worst case, $O_\epsilon\left(\frac{n^2}{m^2}\right)$ average case.*

**Proof.** The space complexity is dominated by the need to store the inverse correlation matrix of known samples required for $\tilde{\mathcal{A}}_j^{(k)}$. Even though the correlation matrix is sparse, the inverse is in general not, requiring the whole $\frac{n_{\mathrm{c}}^{(k)} \times (n_{\mathrm{c}}^{(k)}+1)}{2}$ storage space for the upper or lower triangle of the symmetric matrix.

The worst case corresponds to all $n_{\mathrm{c}}^{(k)}$ samples correlated to one Voronoi region, and the average to samples distributed uniformly. The time and communication complexities are dominated by the requirement of the consensus algorithm. ■

*5.1.1. Broadcast method for comparison* Here, we compare our method against a simple algorithm which floods the network with new information at each sample time as a way of judging its efficiency. This algorithm would work as follows. At each timestep, all samples and locations are disseminated through the network, such that each node obtains the entire vectors $X$ and $Y$. A (centralized) projected gradient descent algorithm can then be run by $\mathrm{N}_j$, $j \in \{1, \ldots, m\}$ to find the next sample locations for those agents within $V_j(Q)$. Since all nodes have the same information, any such algorithms should converge to the same final locations, so there is no difficulty with overlapping computations. Since this method is only given for comparison, we assume this is the case. Once a node has calculated the next location for all of the agents which will be in that Voronoi cell, the control vectors may be transmitted to them. The information dissemination corresponds to an all-to-all broadcast in which each node begins with a distinct message of length $|\mathrm{I}_{\mathrm{Local}}^{(k+1)}(j, P)|$ units. There are a number of different ways this may be carried out. Here we assume the flooding method in [45], which is optimal for time complexity. In this method, every node continues to transmit any new information to all neighbors until new information is exhausted. The proof of the next result follows from combining [45] with the assumptions on $\mathfrak{Q}_{\mathrm{N}}$ and the fact that the total number of initial messages to be disseminated is $n$ times the number of bits required to convey a spatial location plus the number required to convey a sample value. The space complexity is dominated by the requirement to store the entire inverse correlation matrix at each node.

**Proposition 5.9. (Complexity of the broadcast method)** *Under the regularity assumption on $\mathfrak{Q}_{\mathrm{N}}$, local minima of $\tilde{\mathcal{A}}^{(k)}$ may be found by all-to-all broadcast of agent positions and subsequent local projected gradient descent with,*

- *communication complexity in* $\Theta(nm)$
- *time complexity in* $\Theta(n + m)$
- *space complexity in* $\Theta(n^2)$

**Remark 5.10 (Broadcast method requires global positioning)** It should be noted here that while the DISTRIBUTED PROJECTED GRADIENT DESCENT ALGORITHM might be extended to systems with relative positioning (see Remark 5.2), the broadcast method requires global coordinates. •

| Complexity Type | Broadcast | Distributed PGD | |
|---|---|---|---|
| | | Worst | Average |
| Communication | $\Theta(mn)$ | $O_\epsilon(m^2\sqrt[d]{m})$ | $O_\epsilon(m^2\sqrt[d]{m})$ |
| Time | $\Theta(n+m)$ | $O_\epsilon(m\sqrt[d]{m})$ | $O_\epsilon(m\sqrt[d]{m})$ |
| Space | $\Theta(n^2)$ | $O_\epsilon(n^2)$ | $O_\epsilon\big(\big(\frac{n}{m}\big)^2\big)$ |

Table VI. Algorithm complexities. The worst and average cases are over distributions of samples, with the average corresponding to a uniform distribution in $\mathcal{D}$. The bounds for the broadcast method are derived from results in [45].

Table VI lists the complexity bounds side by side. One can see that the distributed method scales better overall with the number of mobile agents. The results with respect to increasing the number of static nodes are less favorable, but include a tradeoff between the average storage requirement and the communication and time complexities. For our algorithm, increasing the number of nodes has the additional benefit of decreasing the average computational burden per node.

### *5.2. Simulations*

We show here an implementation of the DISTRIBUTED PROJECTED GRADIENT DESCENT ALGORITHM with $n$=8 robotic agents and the square $\mathcal{D}$ with vertices $\{(1,1),(1,2),(2,2),(2,1)\}$. The mean regression functions $f_i$ are give by $\mathbf{f}((x,y),t) = (1,x,y)^T$ and the separable correlation function is defined by $\mathrm{Cov}[Z(s_1,t_1),Z(s_2,t_2)] = C_{\text{trunc}}(\|s_1-s_2\|,0.25)C_{\text{trunc}}(|t_1-t_2|,6.9)$, where

$$C_{\text{trunc}}(\delta, r_{\text{s}}) = \begin{cases} e^{-\frac{\delta}{10*r_{\text{s}}}}\big(1-\frac{3\delta}{2r_{\text{s}}}+\frac{\delta^3}{2r_{\text{s}}^3}\big) & \text{if } \delta \le r_{\text{s}}, \\ 0 & \text{otherwise.} \end{cases}$$

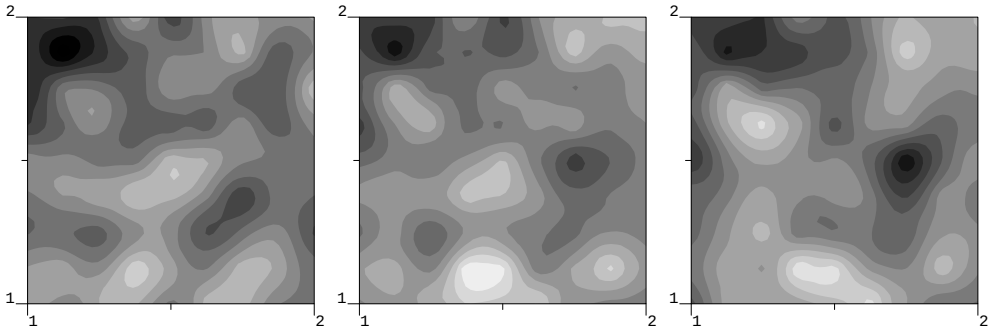
This is a tapered exponential covariance function as suggested in [46]. We use $\omega = 0.02$, $u_{\max} = 0.2$, and ran the simulated experiment for $k_{\max} = 40$ timesteps. The values of our tuning parameters were $\nu = 20$, $q = 0.1$, $\beta_0 = \mathbf{0}$, and $\mathbf{K}_0 = 10\,\boldsymbol{I}$. To illustrate the robustness to failure, $\mathrm{R}_7$ ceased communications after timestep $10$, and $\mathrm{R}_6$ after timestep $15$. As we are ultimately interested in measuring performance with respect to the actual posterior predictive distribution (i.e., after samples have been taken), the comparisons between different approaches in this section are made using that metric as opposed to the various approximations employed in the distributed implementation.

We simulated the sampled data as follows. A $9\times 9$ grid of fixed spatial locations was set up, defining a vector, $X_{\text{grid}} \in \mathcal{D}_e^{3240}$ of fixed grid points over the space-time region (one for each grid point at each

sampling timestep). A random vector was drawn from the joint prior distribution of all grid locations, under the assumption that $\sigma^2$ and $\beta$ take their prior mean values, i.e., from the multivariate distribution,

$$\mathrm{Norm}_{3240}\left(\mathbf{F}(X_{\text{grid}})^T\beta_0, \sigma_0^2\mathbf{K}(X_{\text{grid}})\right), \text{ where } \sigma_0^2 = \frac{q\nu}{\nu-2}$$

denotes the prior mean of $\sigma^2$. Each time a sample was taken by an agent, the value was determined based on least squares (simple kriging) interpolation from this grid of fixed points. Figure 3 depicts snapshots of the sample field used in the examples below. Note that while the three consecutive




(a) (b) (c)

Figure 3. Snapshot of sample field over the square domain, $\mathcal{D}$, for example problem at timesteps (a) $k = 13$, (b) $k = 14$, and (c) $k = 15$.

timesteps depicted are not identical, there is clearly some temporal correlation.

The DISTRIBUTED PROJECTED GRADIENT DESCENT ALGORITHM consists of a gradient descent using an objective function defined by three basic approximations: the least squares approximation of unrealized data; the approximate conditional variance based on regions defined by the static nodes; and the approximate sigma mean based on blocks of samples to reduce the computational burden over time. We begin by examining a centralized, computationally intensive implementation of the gradient descent technique which uses only the least squares approximation. This allows us to compare the (adaptive) gradient descent approach against two a priori approaches: a static configuration and a lawnmower trajectory. The centralized implementation is executed by running the DISTRIBUTED PROJECTED GRADIENT DESCENT ALGORITHM with one single static node ($m = 1$), responsible for the whole predictive domain and using all samples up to step $k-1$ to estimate $\varphi$ at step $k$. In the language of Section 4.2, we let $t_{\text{blk}}$=$k_{\text{max}}$. Each of the three approaches begins from the same initial configuration, and the sample values are drawn from the same predetermined field as described above.

Figure 4 illustrates the first three steps of trajectories taken in each case. We use only three steps for

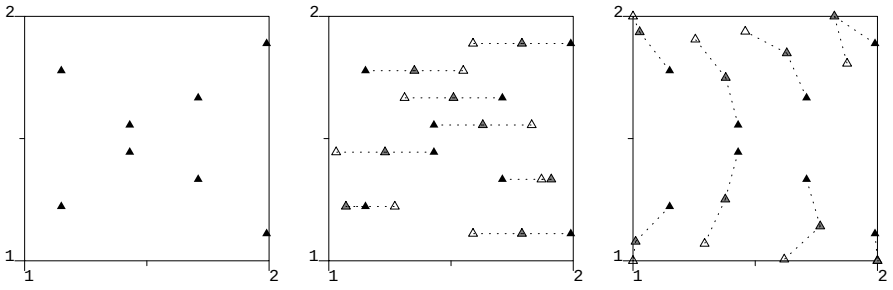



Figure 4. Illustration of the difference between the three simulated approaches. Shown are the first three samples taken by each agent using (a) the static configuration, (b) the a priori lawnmower approach, and (c) the centralized gradient descent. The initial position of each agent is depicted by a black triangle, the second and third as successively lighter triangles.

illustrative purposes, as the trajectories quickly overlap and become difficult to read.

In Figure 5, we compare snapshots of the posterior predictive mean and posterior predictive variance of each of the three approaches, based on all samples taken over the course of the experiment (although only those samples correlated to the given timestep are depicted in the figure). These snapshots, using all samples correlated to the predictive timestep instead of only those preceding it, are motivated by the stated goal of reducing the average variance over the entire predictive region. Note that both the gradient method and lawnmower seem to catch most of the major features of the sample field, while the static configuration results in small neighborhoods of very low posterior variance due to the nontrivial temporal correlation. In Figure 6, we compare the overall average posterior predictive variance resulting from the three approaches at each step of the simulation (still after the samples at the given step have been taken). It should be noted here that in our extensive simulations there were several instances in which actual sample values differed enough from the least squares estimation as to cause erratic results. Those illustrated here are among the more stable ones. Future work will be devoted to identify situations in which the least squares estimation compromises the algorithm performance.

We next ran the DISTRIBUTED PROJECTED GRADIENT DESCENT ALGORITHM with $m = 9$ static nodes using the three approximation methods discussed in Section 4.2. For the recent sample method and the block update method, we used a block size of $t_{\text{blk}} = 7$, the minimum size to cover all correlated time lags. In the block update method, we skipped $t_{\text{skip}} = 6$ timesteps between update blocks. For the

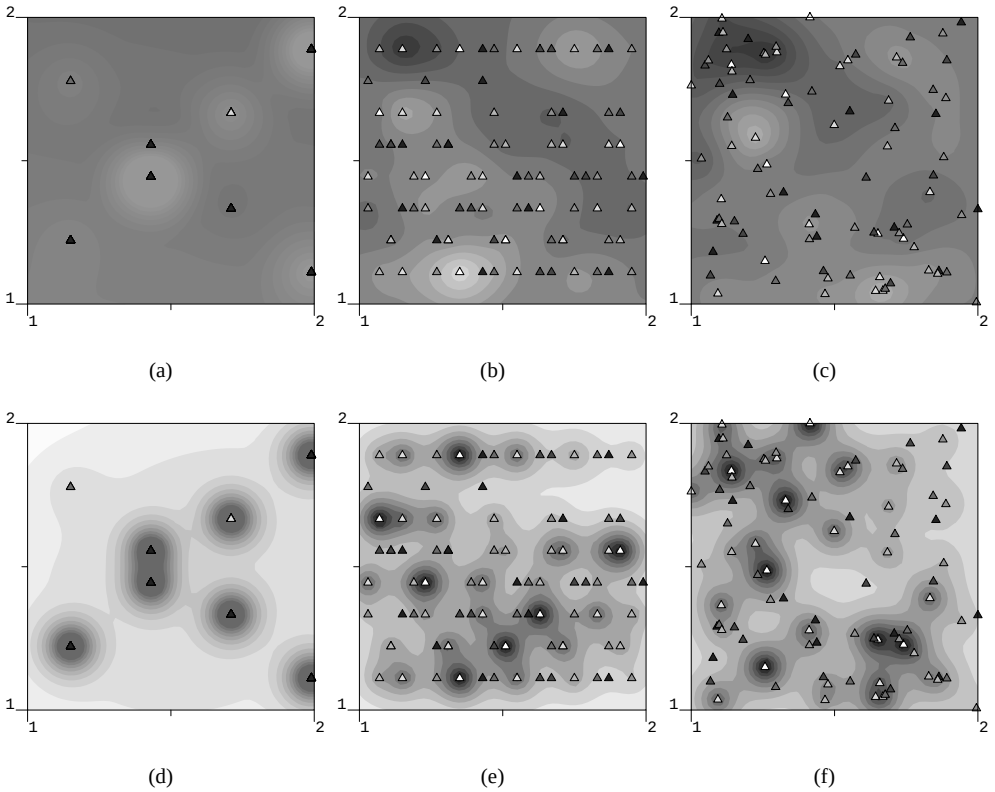



Figure 5. Contours of the first two moments of the posterior predictive distribution at time $t = 14$ based on all samples taken by each of the three centralized trajectories. To avoid unnecessary complication, we have plotted only the positions of samples, not the individual agent trajectories. In each case, all samples correlated in time to step $14$ are depicted as triangles, with the shade of the triangle representing the level of temporal correlation (white triangles are samples taken at step $14$, and darker triangles are farther away in time and thus have smaller correlation to the predictive snapshot). Plots (a) and (d) show the posterior predictive mean and variance, respectively, resulting from samples taken by the static approach. Similarly (b) and (e) correspond to the lawnmower approach, while (c) and (f) correspond to the centralized gradient descent. Note that the different shades of the triangles in the static configuration are a result of the dropped agents (the last sample of $\mathrm{R}_6$ is closely correlated to timestep $14$, thus light in color).

exploration-exploitation method, we used a larger initial block of $t_{\text{blk}} = 14$ samples. Figure 7 shows a snapshot of the posterior mean contours resulting from samples taken along each of these network trajectories. Figure 8 compares the overall average posterior variance as a function of timestep, based on samples taken using the centralized gradient approach against the three distributed approaches.

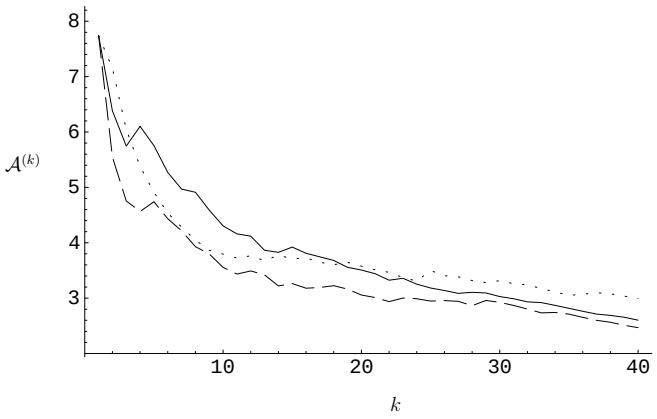



Figure 6. Value of $\mathcal{A}^{(k)}$ as a function of $k$ for the static (dotted), lawnmower (solid), and centralized gradient descent (dashed) approaches. The gradient method does the best persistently, throughout the experiment. The static approach shows the most consistent curve, but results in the highest average error overall.

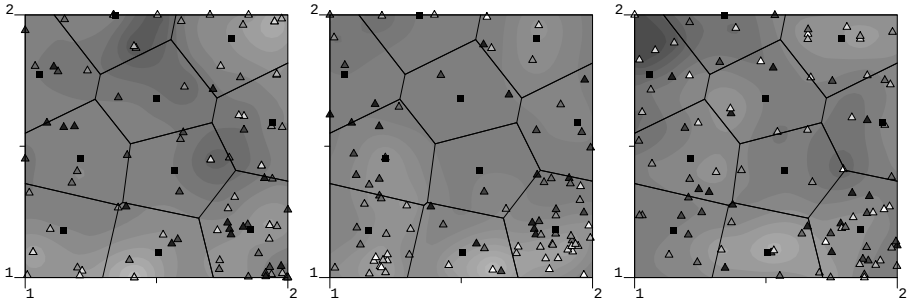



Figure 7. Contours of the posterior predictive mean at time $t = 14$ based on all samples using the (a) block update method; (b) recent sample method; and (c) the exploration-exploitation method. To avoid unnecessary complication, we have plotted only the positions of samples, not the individual agent trajectories. In each case, all samples correlated in time to step $14$ are depicted as triangles, with the shade of the triangle representing the level of temporal correlation (white triangles are samples taken at step $14$, and darker triangles are farther away in time and thus have smaller correlation to the predictive snapshot).

## 6. 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,

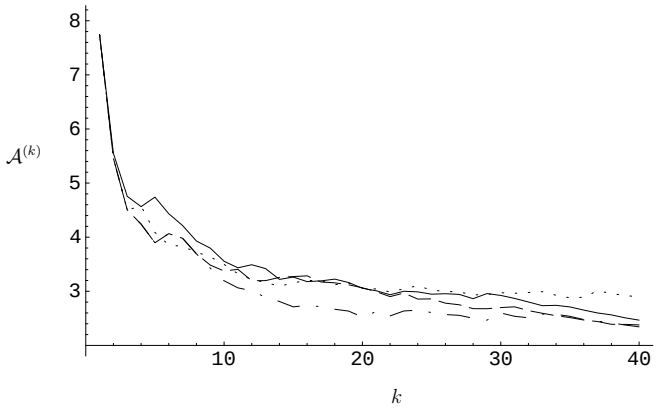



Figure 8. Value of $\mathcal{A}^{(k)}$ as a function of $k$ using the centralized gradient method (solid), block update method (dashed), recent sample method (dot-dash), and exploration-exploitation method (dotted). The three approximation methods perform similarly to the centralized gradient method. The exploration-exploitation approach lags behind at the end, while the two other approximation methods actually do better than the centralized gradient approach.

and developed a novel approximation of the variance of the posterior predictive distribution which may be calculated in a sequential and distributed fashion. Using this formulation, we have developed a projected gradient descent algorithm which is distributed over the network of nodes and robots. We have examined the complexity of this approach, and compared it against the lower bound complexity of a centralized "broadcast" method, showing that the distributed approach scales better with the number of mobile agents. Future work will focus on theoretical guarantees on the accuracy of the approximation $\tilde{\mathcal{A}}^{(k)}$ and on the robustness to failure of the proposed algorithm. Guarantees on the least squares approximation have proved extremely difficult, especially in the case of sparse samples. Future work will also explore alternative methods, such as the expected value of the gradient of the random field. The extension of these methods to networks in which the static agents are replaced by slow moving ones would also be of interest. As mentioned in Remark 2.4, special care must be taken to avoid singularities when generating local approximations for the universal kriging model. Rigorous methods for handling this situation are also worth exploring.

## 7. Acknowledgments

The authors thank B. Sansó and H. Lee for several conversations on Bayesian spatial statistics. This research was partially supported by NSF CAREER Award ECS-0546871.

REFERENCES


1. Willcox JS, Bellingham JG, Zhang Y, Baggeroer AB. Performance metrics for oceanographic surveys with autonomous underwater vehicles. *IEEE Journal of Oceanic Engineering* 2001; **26**(4):711–725.
2. Handcock MS, Wallis JR. An approach to statistical spatial-temporal modeling of meteorological fields. *Journal of the American Statistical Association* 1994; **89**(426):368–378.
3. Higdon D. A process-convolution approach to modelling temperatures in the north atlantic ocean. *Environmental and Ecological Statistics* 1998; **5**:173–190.
4. Gaudard M, Karvson M, Linder E, Sinha D. Bayesian spatial prediction. *Environmental and Ecological Statistics* 1999; **6**:147–171.
5. Lee ND, Zidek JV. *Statistical Analysis of Environmental Space-Time Processes*. Springer Series in Statistics, Springer: New York, 2006.
6. Chaloner K, Verdinelli I. Bayesian experimental design, a review. *Statistical Science* 1995; **10**(3):273–304.
7. Pukelsheim F. *Optimal Design of Experiments*, *Classics in Applied Mathematics*, vol. 50. SIAM: Philadelphia, PA, 2006.
8. Liski EP, Mandal NK, Shah KR, Sinha BK. *Topics in Optimal Design*, *Lecture Notes in Statistics*, vol. 163. Springer: New York, 2002.
9. Popa DO, Sreenath K, Lewis FL. Robotic deployment for environmental sampling applications. *International Conference on Control and Automation*, Budapest, Hungary, 2005; 197–202.
10. MacKay DJC. Information-based objective functions for active data selection. *Neural Computation* 1992; **4**(4):590–604.
11. Gramacy RB. Bayesian treed Gaussian process models. PhD Thesis, University of California, Santa Cruz, CA 95064 December 2005. Department of Applied Math & Statistics.
12. Caselton WF, Zidek JV. Optimal monitoring network designs. *Statistics & Probability Letters* 1984; **2**(4):223–227.
13. Krause A, Singh A, Guestrin C. Near-optimal sensor placements in Gaussian processes: Theory, efficient algorithms and empirical studies. *Journal of Machine Learning Research* 2008; **9**:235–284.
14. Singh A, Krause A, Guestrin C, Kaiser WJ. Efficient informative sensing using multiple robots. *Journal of Artificial Intelligence Research* 2009; **34**:707–755.
15. Leonard NE, Paley D, Lekien F, Sepulchre R, Fratantoni DM, Davis R. Collective motion, sensor networks and ocean sampling. *Proceedings of the IEEE* 2007; **95**(1):48–74.
16. Zhang F, Fiorelli E, Leonard NE. Exploring scalar fields using multiple sensor platforms: tracking level curves. *IEEE Conf. on Decision and Control*, New Orleans, LA, 2007; 3579–3584.
17. Cortés J. Distributed Kriged Kalman filter for spatial estimation. *IEEE Transactions on Automatic Control* 2009; **54**(12):2816–2827.
18. Lynch KM, Schwartz IB, Yang P, Freeman RA. Decentralized environmental modeling by mobile sensor networks. *IEEE Transactions on Robotics* 2008; **24**(3):710–724.
19. Martínez S. Distributed interpolation schemes for field estimation by mobile sensor networks. *IEEE Transactions on Control Systems Technology* 2010; **18**(2):491–500.
20. Demetriou MA, Hussein II. Estimation of spatially distributed processes using mobile spatially distributed sensor network. *SIAM Journal on Control and Optimization* 2009; **48**(1):266–291.
21. Hoffmann GM, Tomlin CJ. Mobile sensor network control using mutual information methods and particle filters. *IEEE Transactions on Automatic Control* 2010; **55**(1):32–47.
22. Ucinski D. *Optimal Measurement Methods for Distributed Parameter System Identification*. CRC Press, 2005, doi:

10.1201/9780203026786.fmatt.

23. Berke O. On spatiotemporal prediction for on-line monitoring data. *Communications in Statistics - Theory and Methods* 1998; **27**(9):2343–2369.
24. Mardia KV, Goodall C, Redfern EJ, Alonso FJ. The Kriged Kalman filter. *Test* 1998; **7**(2):217–285. With discussion.
25. Kitanidis PK. Parameter uncertainty in estimation of spatial functions: Bayesian analysis. *Water Resources Research* 1986; **22**:449–507.
26. Wiens DP. Robustness in spatial studies I: minimax prediction and II: minimax design. *Environmetrics* 2005; **16**:191–203 and 205–217.
27. Sutton RS, Barto AG. *Reinforcement Learning: An Introduction*. MIT Press, 1998.
28. Krause A, Guestrin C. Nonmyopic active learning of Gaussian processes: an exploration-exploitation approach. *International Conference on Machine Learning*, Corvallis, OR, 2007.
29. Graham R, Cortés J. A cooperative deployment strategy for optimal sampling in spatiotemporal estimation. *IEEE Conf. on Decision and Control*, Cancun, Mexico, 2008; 2432–2437.
30. Graham R, Cortés J. Distributed sampling of random fields with unknown covariance. *American Control Conference*, St. Louis, MO, 2009; 4543–4548.
31. Bertsekas DP. On the Goldstein-Levitin-Polyak gradient projection method. *IEEE Transactions on Automatic Control* 1976; **21**(2):174–184.
32. Leonard T, Hsu JSJ. *Bayesian Methods, Cambridge series in statistical and probabilistic mathematics*, vol. 1. Cambridge University Press: Cambridge, UK, 1999.
33. Gottschalk L. Interpolation of runoff applying objective methods. *Stochastic Hydrology and Hydraulics* 1993; **7**:269–281.
34. Wackernagel H. *Multivariate Geostatistics*. 3rd edn., Springer: New York, 2006.
35. Robert CP, Casella G. *Monte Carlo statistical methods*. Springer texts in statistics, Springer: New York, 2004.
36. Bertsekas DP, Tsitsiklis JN. *Parallel and Distributed Computation: Numerical Methods*. Athena Scientific, 1997.
37. Olfati-Saber R, Murray RM. Consensus problems in networks of agents with switching topology and time-delays. *IEEE Transactions on Automatic Control* 2004; **49**(9):1520–1533.
38. Bullo F, Cortés J, Martínez S. *Distributed Control of Robotic Networks*. Applied Mathematics Series, Princeton University Press, 2009. Electronically available at http://coordinationbook.info.
39. Lynch NA. *Distributed Algorithms*. Morgan Kaufmann, 1997.
40. Cormen TH, Leiserson CE, Rivest RL, Stein C. *Introduction to Algorithms*. 2 edn., MIT Press, 2001.
41. Carle J, Myoupo JF. Topological properties and optimal routing algorithms for three dimensional hexagonal grid networks. *High performance computing in the Asia-Pacific region*, 2000; 116–121.
42. Chen MS, Shin KG, Kandlur DD. Addressing, routing, and broadcasting in hexagonal mesh multiprocessors. *IEEE Transactions on Computers* jan 1990; **39**(1):10–18.
43. Okabe A, Boots B, Sugihara K, Chiu SN. *Spatial Tessellations: Concepts and Applications of Voronoi Diagrams*. 2 edn., Wiley Series in Probability and Statistics, Wiley, 2000.
44. Mohar B. The Laplacian spectrum of graphs. *Graph Theory, Combinatorics, and Applications*, vol. 2, Alavi Y, Chartrand G, Oellermann OR, Schwenk AJ (eds.). Wiley, 1991; 871–898.
45. Topkis DM. Concurrent broadcast for information dissemination. *IEEE Transactions on Software Engineering* oct 1985; **SE-11**(10):1107–1112.
46. Furrer R, Genton MG, Nychka D. Covariance tapering for interpolation of large spatial datasets. *Journal of Computational and Graphical Statistics* 2006; **15**(3):502–523.

47. Bernstein DS. *Matrix Mathematics*. Princeton University Press: Princeton, NJ, 2005.
48. Henderson HV, Searle SR. On deriving the inverse of a sum of matrices. *SIAM Review* 1981; **23**(1):53–60.
49. Udwadia FE. Some convergence results related to the JOR iterative method for symmetric, positive-definite matrices. *Applied Mathematics and Computation* 1992; **47**(1):37–45.

## APPENDIX

### I. Predictions with a subset of measurements

We present here a series of results concerning the relationship between subsets of spatiotemporal sample locations and hypothetical predictions made from partial information. Let $Y \in \mathbb{R}^n$ denote a full set of measurements at locations $X \in \mathcal{D}_e^n$. Let $n_1, n_2 \in \mathbb{Z}_{>0}$ such that $n_1 + n_2 = n$. Consider a partition of the measurements $Y = (Y_1, Y_2)$ such that $Y_1 \in \mathbb{R}^{n_1}$ and $Y_2 \in \mathbb{R}^{n_2}$, and a similar partition of $X$. Note that due to the invariance of $\varphi$ and $\phi$ under permutations of the samples, our discussion is valid for any partition of the measurements, not necessarily restricted to samples that are sequential in time. We will use $\mathbf{K}_1$, respectively $\mathbf{K}_2$, to denote the correlation matrix of locations $X_1$, respectively $X_2$, and analogous notation for the matrices $\mathbf{F}_1, \mathbf{F}_2, \mathbf{E}_1, \mathbf{E}_2$. Let $\mathbf{K}_{12} = \mathbf{K}_{21}^T \in \mathbb{R}^{n_1 \times n_2}$ denote the matrix of cross-correlation between the two location vectors.

We begin with a multivariate version of the posterior predictive variance from Proposition 2.1, which can be considered the hypothetical distribution of the measurements at space-time locations $X_2$ given the samples $Y_1$. This result can be obtained by applying Bayes Theorem to the prior model.

**Lemma I.1 (Multivariate posterior predictive distribution)** *Under the Bayesian model* (2)*, the multivariate posterior predictive distribution of hypothetical samples $Y_2$ conditional on data $Y_1$ is the $n_2$-variate shifted Students t distribution with $\nu + n_1$ degrees of freedom, which takes the form,*

$$p(Y_2|Y_1) \propto \det \mathrm{Var}[Y_2|Y_1]^{-\frac{1}{2}} \left(1 + \frac{(Y_2 - \mathrm{E}[Y_2|Y_1])^T \mathrm{Var}[Y_2|Y_1]^{-1} (Y_2 - \mathrm{E}[Y_2|Y_1])}{\nu + n_1 - 2}\right)^{-\frac{\nu+n_1+n_2}{2}}.$$

*Here, the expectation is given by*

$$\mathrm{E}[Y_2|Y_1] = \xi_{2|1}^T (\mathbf{E}_1 + \mathbf{K}_0^{-1})^{-1} \left(\mathbf{F}_1 \mathbf{K}_1^{-1} Y_1 + \mathbf{K}_0^{-1} \beta_0\right) + \mathbf{K}_{21} \mathbf{K}_1^{-1} Y_1,$$

*where $\xi_{2|1} = \mathbf{F}_2 - \mathbf{F}_1 \mathbf{K}_1^{-1} \mathbf{K}_{12}$. The covariance matrix is given by*

$$\mathrm{Var}[Y_2|Y_1] = \varphi(Y_1, X_1) \phi(X_2; X_1),$$

*where, with a slight abuse of notation, we have used $\phi(X_2; X_1)$ to denote the following multivariate*

*extensions of $\phi$ and $\varphi$,*

$$\phi(X_2;X_1) = \mathbf{K}_2 - \mathbf{K}_{21}\mathbf{K}_1^{-1}\mathbf{K}_{12} + \xi_{2|1}^T\left(\mathbf{K}_0^{-1}+\mathbf{E}_1\right)^{-1}\xi_{2|1},$$
$$\varphi(Y_1,X_1) = \frac{1}{\nu+n_1-2}\left(q\nu + \left(Y_1-\mathbf{F}_1^T\beta_0\right)^T\left(\mathbf{K}_1+\mathbf{F}^T\mathbf{K}_0\mathbf{F}\right)^{-1}\left(Y_1-\mathbf{F}_1^T\beta_0\right)\right).$$

In the sequel, we will find useful the matrices $\mathcal{M}\in\mathbb{R}^{(n+p)\times(n+p)}$ and $M_2\in\mathbb{R}^{(n_2+p)\times(n_2+p)}$ and vectors $\mathcal{U}\in\mathbb{R}^{n+p}$ and $U_2\in\mathbb{R}^{n_2+p}$ defined as,

$$\mathcal{M}=\begin{bmatrix}\mathbf{K} & \mathbf{F}^T\\ \mathbf{F} & -\mathbf{K}_0^{-1}\end{bmatrix},\quad \mathcal{U}=\begin{bmatrix}Y\\ -\mathbf{K}_0^{-1}\beta_0\end{bmatrix},\quad M_2=\begin{bmatrix}\mathbf{K}_2 & \mathbf{F}_2^T\\ \mathbf{F}_2 & -\mathbf{K}_0^{-1}\end{bmatrix},\quad U_2=\begin{bmatrix}Y_2\\ -\mathbf{K}_0^{-1}\beta_0\end{bmatrix}.$$

Note that $M_2$ is the lower right submatrix of $\mathcal{M}$ under a different partition, and $U_2$ is the corresponding subvector of $\mathcal{U}$. It can be shown that the matrices $\mathcal{M}$ and $M_2$ are invertible.

**Proposition I.2 (Approximate conditional variance)** *The term $\phi(x_0;X)$ may be written in terms of spatiotemporal locations $X_2$ as,*

$$\phi(x_0;X) = \phi(x_0;X_2) - (\mathbf{k}_1-\mu_1)^T\phi(X_1;X_2)^{-1}(\mathbf{k}_1-\mu_1),\ \textit{where}$$
$$\mu_1 = \begin{bmatrix}\mathbf{K}_{21}\\ \mathbf{F}_1\end{bmatrix}^T M_2^{-1}\begin{bmatrix}\mathbf{k}_2\\ \mathbf{f}(x_0)\end{bmatrix},\quad \mathbf{k}_1=\mathrm{Cor}[Z(x_0),Y_1],\quad \mathbf{k}_2=\mathrm{Cor}[Z(x_0),Y_2].$$

*Therefore $\phi(x_0;X)\le\phi(x_0;X_2)$ with equality if and only if $\mathbf{k}_1=\mu_1$.*

**Proof.** First, we note that the conditional variance can be written using $\mathcal{M}$ as,

$$\phi(x_0;X) = \mathrm{Cor}[Z,Z] - \begin{bmatrix}\mathbf{k}\\ \mathbf{f}(x_0)\end{bmatrix}^T\mathcal{M}^{-1}\begin{bmatrix}\mathbf{k}\\ \mathbf{f}(x_0)\end{bmatrix}.$$

Next, we point out that with the proper partitioning of $\mathcal{M}$, the matrix $\phi(X_1;X_2)$ is the Schur Complement, $(M_2\,|\mathcal{M}\,)$. Using this, and a similar partition of the vector $\mathbf{k}$, one arrives at the result. ■

The following result illustrates a number of ways in which the sigma mean may be restated.

**Lemma I.3 (Restated sigma mean)** *The quadratic form $\mathcal{U}^T\mathcal{M}^{-1}\mathcal{U}$ can be expressed as,*

$$\mathcal{U}^T\mathcal{M}^{-1}\mathcal{U} = (Y-\mathbf{F}^T\beta_0)^T(\mathbf{K}+\mathbf{F}^T\mathbf{K}_0\mathbf{F})^{-1}(Y-\mathbf{F}^T\beta_0) - \beta_0^T\mathbf{K}_0^{-1}\beta_0 \tag{9a}$$
$$= Y^T\mathbf{K}^{-1}Y - \left(\mathbf{K}_0^{-1}\beta_0+\mathbf{F}\mathbf{K}^{-1}Y\right)^T\left(\mathbf{K}_0^{-1}+\mathbf{E}\right)^{-1}\left(\mathbf{K}_0^{-1}\beta_0+\mathbf{F}\mathbf{K}^{-1}Y\right) \tag{9b}$$
$$= U_2^TM_2^{-1}U_2 + (Y_1-\mathrm{E}[Y_1|Y_2])^T\phi(X_1;X_2)^{-1}(Y_1-\mathrm{E}[Y_1|Y_2]). \tag{9c}$$

*Furthermore, the term $\varphi(Y,X)$ may be written as,*

$$\varphi(Y,X) = \frac{q\nu+\beta_0^T\mathbf{K}_0^{-1}\beta_0+\mathcal{U}^T\mathcal{M}^{-1}\mathcal{U}}{\nu+n-2}. \tag{10}$$

**Proof.** Each of the three representations of the quadratic form may be derived directly by using [47, Proposition 2.8.7] to expand the inverse matrix onto Schur complements. Plugging representation (9a) into (10) yields the form given in Proposition 2.1. ■

The generalized least squares (GLS) approximation arises naturally from partitioning the elements of the term $\mathbf{K}^{-1}Y$. The following lemma gives explicit form in terms of the GLS error.

**Lemma I.4 (Generalized least squares approximations)** *Let $\hat{Y}_{LS} = \mathbf{K}_{21}\mathbf{K}_1^{-1}Y_1$ be the generalized least squares estimate of $Y_2$ based on samples $Y_1$ (conditional on all parameters), and let $\overline{y}_{LS} = Y_2 - \hat{Y}_{LS}$. Then we can write,*

$$\mathbf{K}^{-1}Y = \begin{bmatrix} \mathbf{K}_1^{-1}Y_1 \\ \mathbf{0} \end{bmatrix} + \begin{bmatrix} -\mathbf{K}_1^{-1}\mathbf{K}_{12}\left(\mathbf{K}_1 \,|\mathbf{K}\right)^{-1}\overline{y}_{LS} \\ \left(\mathbf{K}_1 \,|\mathbf{K}\right)^{-1}\overline{y}_{LS} \end{bmatrix}. \tag{11}$$

**Proof of Proposition 4.3.** Using (11) in Lemma I.4 we write,

$$\begin{aligned} Y^T\mathbf{K}^{-1}Y &= Y_1^T\mathbf{K}_1^{-1}Y_1 + (Y_2 - \hat{Y}_{\mathrm{LS}})^T\left(\mathbf{K}_1 \,|\mathbf{K}\right)(Y_2 - \hat{Y}_{\mathrm{LS}}) \\ \mathbf{F}\mathbf{K}^{-1}Y &= \mathbf{F}_1^T\mathbf{K}_1^{-1}Y_1 + \xi_{2|1}\left(\mathbf{K}_1 \,|\mathbf{K}\right)(Y_2 - \hat{Y}_{\mathrm{LS}}). \end{aligned}$$

Here we have used the simpler indexed notation, $Y_1 = Y^{(1:k)}$ and $Y_2 = Y^{(k+1)}$ to simplify the algebra. Applying these results to (9b) in Lemma I.3 yields,

$$\begin{aligned} \mathcal{U}^T\mathcal{M}^{-1}\mathcal{U} = Y_1^T\mathbf{K}_1^{-1}Y_1 &- (\mathbf{K}_0^{-1}\beta_0 + \mathbf{F}_1\mathbf{K}_1^{-1}Y_1)^T(\mathbf{K}_0^{-1} + \mathbf{E})^{-1}(\mathbf{K}_0^{-1}\beta_0 + \mathbf{F}_1\mathbf{K}_1^{-1}Y_1) \\ &+ (Y_2 - \hat{Y}_{\mathrm{LS}})^T\left(\mathbf{K}_1 \,|\mathbf{K}\right)^{-1}(Y_2 - \hat{Y}_{\mathrm{LS}}) \\ &- 2(\mathbf{K}_0^{-1}\beta_0 + \mathbf{F}_1\mathbf{K}_1^{-1}Y_1)^T(\mathbf{K}_0^{-1} + \mathbf{E})^{-1}(\xi_{2|1}\left(\mathbf{K}_1 \,|\mathbf{K}\right)^{-1}(Y_2 - \hat{Y}_{\mathrm{LS}})) \\ &- (\xi_{2|1}\left(\mathbf{K}_1 \,|\mathbf{K}\right)^{-1}(Y_2 - \hat{Y}_{\mathrm{LS}}))^T(\mathbf{K}_0^{-1} + \mathbf{E})^{-1}(\xi_{2|1}\left(\mathbf{K}_1 \,|\mathbf{K}\right)^{-1}(Y_2 - \hat{Y}_{\mathrm{LS}})). \end{aligned}$$

Using, e.g. [48, Equation (12,17)], we can write,

$$\phi(X_2; X_1) = \left(\mathbf{K}_1 \,|\mathbf{K}\right)^{-1} + \left(\mathbf{K}_1 \,|\mathbf{K}\right)^{-1}\xi_{2|1}^T(\mathbf{K}_0^{-1} + \mathbf{E})^{-1}\xi_{2|1}\left(\mathbf{K}_1 \,|\mathbf{K}\right)^{-1}.$$

With some algebraic manipulation we arrive at the result,

$$\begin{aligned} \mathcal{U}^T\mathcal{M}^{-1}\mathcal{U} = Y_1^T\mathbf{K}_1^{-1}Y_1 &- (\mathbf{K}_0^{-1}\beta_0 + \mathbf{F}_1\mathbf{K}_1^{-1}Y_1)^T(\mathbf{K}_0^{-1} + \mathbf{E})^{-1}(\mathbf{K}_0^{-1}\beta_0 + \mathbf{F}_1\mathbf{K}_1^{-1}Y_1) \\ &+ (\overline{y}_{\mathrm{LS}} - 2\overline{\mu}_{2|1})\phi(X_2; X_1)^{-1}\overline{y}_{\mathrm{LS}}. \end{aligned}$$

The result follows from Lemma I.3. ■

## II. Near optimal relaxation parameter for JOR

Here we present some results regarding a relaxation parameter for the JOR algorithm which is nearly optimal with respect to the rate of convergence of the algorithm for a certain class of matrices. Specifically we are interested in the class of symmetric, positive definite matrices $C$ with ones on the diagonal. Let $y(t) = (y_1(t), \ldots, y_n(t))^T \in \mathbb{R}^n$ be the vector updated during the JOR iteration in (5). Let $e(t) = \|C^{-1}b - y(t)\|$ denote the error at iteration $t$. We may write,

$$e(t) \leq \left(\rho(\boldsymbol{I} - hC)\right)^t e(0), \tag{12}$$

giving a bound on the error at step $t$ based on the initial error. The value of $\rho(\boldsymbol{I} - hC)$ therefore controls the rate of convergence, and choosing the relaxation parameter, $h$, is of vital importance. Throughout this section we will use the shorthand $\lambda_{\max} = \lambda_{\max}(C)$ and $\lambda_{\min} = \lambda_{\min}(C)$. The work [49] provides results concerning the convergence of the JOR algorithm, including an optimal relaxation parameter, which in our case is equivalent to $h_{\text{opt}} = \frac{2}{\lambda_{\max}+\lambda_{\min}}$. In this section we will introduce an approximation to this optimal value which may be calculated in a distributed manner.

**Proposition II.1.** *Assume that $C \in \mathbb{R}^{n\times n}$ is a symmetric positive definite matrix with all diagonal entries equal to 1. Let $\beta$ and $\alpha$ denote the maximum off-diagonal entry of $C$ and the maximum off-diagonal row sum of $C$, respectively,*

$$\beta = \max_{i\neq j\in\{1,\ldots,n\}} \{c_{ij}\} \geq 0, \qquad \alpha = \max_{i\in\{1,\ldots,n\}} \Big\{ \sum_{\substack{j=1 \\ j\neq i}}^{n} c_{ij} \Big\} \geq 0.$$

*Let $h^* = \frac{2}{2+\alpha-\beta}$. Then using $h^*$ as the relaxation parameter in the JOR algorithm to solve $y = C^{-1}b$ results in guaranteed convergence.*

**Proof.** Recall from Section 2.3 that convergence of the JOR algorithm is guaranteed as long as $h^* \in \left(0, \frac{2}{\lambda_{\max}}\right)$. This can also be seen from (12), since $h$ outside of this range would yield $\rho(\boldsymbol{I} - hC) > 1$. Since $C$ is symmetric positive definite with 1's on the diagonal, all off-diagonal entries must have magnitude strictly less than 1. Thus $1 - \beta > 0$. The Gershgorin circle theorem (e.g. [47, Fact 4.10.13]) implies that $\lambda_{\max} \leq 1 + \alpha$. Together, these two results yield $2 + \alpha - \beta > \lambda_{\max}$, which implies that $\frac{2}{2+\alpha-\beta} < \frac{2}{\lambda_{\max}}$, and the result follows. ■

**Lemma II.2.** *Under the assumptions of Proposition II.1, $h^* \leq \frac{1}{\lambda_{min}}$, with equality if and only if $C$ is the $n \times n$ identity matrix.*

**Proof.** First, note the following implication chain,

$$\lambda_{\min} \leq 1 \implies 2\lambda_{\min} \leq 2 + \alpha - \beta \implies h^* \leq \frac{1}{\lambda_{\min}}.$$

Now, assume that $h^* = \frac{1}{\lambda_{\min}}$. This implies that $\lambda_{\min} = 1 + \alpha - \beta$, but $\lambda_{\min} \leq 1$, and $\alpha \geq \beta$. So we must have $\lambda_{\min} = 1$. Since the diagonal entries of $C$ are all 1, the smallest eigenvalue can only be 1 if all off-diagonal entries are zero, i.e., if $C = \boldsymbol{I}_n$. ■

**Lemma II.3.** *Under the assumptions of Proposition II.1,* $|1 - h^*\lambda_{min}| \geq |1 - h^*\lambda_{max}|$.

**Proof.** Using Lemma II.2, we have $|1 - h^*\lambda_{\min}| = 1 - h^*\lambda_{\min}$. The result may then be shown by two separate cases. First, note that if $h^* \leq \frac{1}{\lambda_{\max}}$ then we have,

$$|1 - h^*\lambda_{\max}| = 1 - h^*\lambda_{\max} \leq |1 - h^*\lambda_{\min}|,$$

so the result holds in this case. For the second case, assume that $h^* > \frac{1}{\lambda_{\max}}$. Then $|1 - h^*\lambda_{\max}| = h^*\lambda_{\max} - 1$. The inclusion principle and the fact that $C$ is positive definite give us the bounds $0 < \lambda_{\min} \leq 1 - \alpha$. Combined with the previously mentioned Gershgorin bound, $\lambda_{\max} \leq 1 + \alpha$, this allows us to write,

$$\frac{\lambda_{\max} + \lambda_{\min}}{2 + \alpha - \beta} \leq 1, \qquad 2\frac{\lambda_{\max} + \lambda_{\min}}{2 + \alpha - \beta} \leq 2,$$
$$h^*\left(\lambda_{\max} + \lambda_{\min}\right) \leq 2, \qquad h^*\lambda_{\max} - 1 \leq 1 - h^*\lambda_{\min}.$$

Thus in all cases, $|1 - h^*\lambda_{\max}| \leq |1 - h^*\lambda_{\min}|$. ■

**Proposition II.4.** *Under the assumptions of Proposition II.1, further assume that* $\lambda_{min} \geq \varpi_\lambda$ *for some* $\varpi_\lambda \in (0, 1)$. *Then* $0 \leq \rho(\boldsymbol{I} - h^*C) < 1 - \frac{2\varpi_\lambda}{2+\alpha-\beta}$

**Proof.** First note that the spectral radius is given by $\max\{|1 - h^*\lambda_{\min}|, |1 - h^*\lambda_{\max}|\}$, and is clearly nonnegative. From Lemma II.3, we have $\rho(\boldsymbol{I} - h^*C) = |1 - h^*\lambda_{\min}|$. From Lemma II.2, we can infer $\rho(\boldsymbol{I} - h^*C) = 1 - h^*\lambda_{\min}$. The upper bound follows by comparing $1 - h^*\lambda_{\min}$ against $1 - h^*\varpi_\lambda$. ■