Communication-Aware, Scalable Gaussian Processes for Decentralized Exploration

Georgios P. Kontoudis

Dissertation submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy
in
Electrical Engineering

Daniel J. Stilwell, Chair
Walid Saad
Ryan K. Williams
Craig A. Woolsey
Pratap Tokekar

December 15, 2021
Blacksburg, Virginia

Keywords: Gaussian Processes, Kriging, Distributed Optimization, Underwater Acoustic Communications, Decentralized Networks.

Communication-Aware, Scalable Gaussian Processes for Decentralized Exploration

Georgios P. Kontoudis

Academic Abstract

In this dissertation, we propose decentralized and scalable algorithms for Gaussian process (GP) training and prediction in multi-agent systems. The first challenge is to compute a spatial field that represents underwater acoustic communication performance from a set of measurements. We compare kriging to cokriging with vehicle range as a secondary variable using a simple approximate linear-log model of the communication performance. Next, we propose a model-based learning methodology for the prediction of underwater acoustic performance using a realistic propagation model. The methodology consists of two steps: i) estimation of the covariance matrix by evaluating candidate functions with estimated parameters; and ii) prediction of communication performance. Covariance estimation is addressed with a multi-stage iterative training method that produces unbiased and robust results with nested models. The efficiency of the framework is validated with simulations and experimental data from field trials. The second challenge is to perform predictions at unvisited locations with a team of agents and limited inter-agent information exchange. To decentralize the implementation of GP training, we employ the alternating direction method of multipliers (ADMM). A closed-form solution of the decentralized proximal ADMM is provided for the case of GP hyper-parameter training with maximum likelihood estimation. Multiple aggregation techniques for GP prediction are decentralized with the use of iterative and consensus methods. In addition, we propose a covariance-based nearest neighbor selection strategy that enables a subset of agents to perform predictions. Empirical evaluations illustrate the efficiency of the proposed methods.

Communication-Aware, Scalable Gaussian Processes for Decentralized Exploration

Georgios P. Kontoudis

General Audience Abstract

In this dissertation, we propose decentralized and scalable algorithms for collaborative multi-agent learning. Mobile robots, such as autonomous underwater vehicles (AUVs), can use predictions of communication performance to anticipate where they are likely to be connected to the communication network. The first challenge is to predict the acoustic communication performance of AUVs from a set of measurements. We compare two methodologies using a simple model of communication performance. Next, we propose a model-based learning methodology for the prediction of underwater acoustic performance using a realistic model. The methodology first estimates the covariance matrix and then predicts the communication performance. The efficiency of the framework is validated with simulations and experimental data from field trials. The second challenge regards the efficient execution of Gaussian processes using multiple agents and communicating as little as possible. We propose decentralized algorithms that facilitate local computations at the expense of inter-agent communications. Moreover, we propose a nearest neighbor selection strategy that enables a subset of agents to participate in the prediction. Illustrative examples with real world data are provided to validate the efficiency of the algorithms.

Dedication

This dissertation is dedicated to my wife, Vasia Abatzi.

Acknowledgments

I would like to thank my advisor Prof. Daniel Stilwell for his mentorship during my PhD. Dan treated me like family and I will never forget that. Moreover, I would like to thank my committee Prof. Saad, Prof. Williams, Prof. Woolsey, and Prof. Tokekar for their guidance.

Thanks to my labmates, Taylor, Jorge, Naina, Lakshmi, Michael, Cong, Stephen, Ben, Charles, Chris, Hans, Min Gyu, Conlan, Jonathan, Henry, Rohit, Devon, Justin, Kushal, Mohamed, and Tayo for creating an amazing and friendly environment in the Center for Marine Autonomy & Robotics.

I would like to thank my friends, Vaggelis Kokorakis, Thodoris Theotokis, Lorentzos Mikroutikos, and Giorgos Koukas for their support throughout my PhD. Moreover, thanks to my friends in academia, Minas Liarokapis and Aris Kanellopoulos for their crucial advice and support during academic speed bumps.

I would like to thank the Blacksburg Rescue Squad, the staff and doctors at LewisGale Montgomery Hospital and LewisGale Medical Center, and the members of the Cook Counseling Center for saving my life and helping me to recover after a cardiac arrest.

I would like to thank my family, Xanthi Kontoudi, Pantelis Kontoudis, Mairy Kontoudi, Stelios Katsourinis, Sotiris Katsourinis, and Xanthippe Katsourini for their love.

Finally, I would like to thank my wife, Vasia Abatzi for her endless love all these years. Vasia is my significant other and none of this research would have been conducted without her support.

Contents

List of Figures

List of Tables

List of Abbreviations

ADMM Alternating Direction Method of Multipliers

apx-GP analytical px-ADMM GP training

AUV Autonomous Underwater Vehicle

BCM Bayesian Committee Machine

BIC Bayesian Information Criterion

c-GP consensus ADMM GP training

CBNN Covariance-Based Nearest Neighbor

COK Ordinary Co-Kriging

DAC Discrete-Time Average Consensus

DALE Distributed Algorithm to solve systems of Linear Equations

DEC-apx-GP Decentralized apx-ADMM-GP training

DEC-BCM Decentralized BCM prediction

DEC-c-GP Decentralized c-ADMM-GP training

DEC-gapx-GP Decentralized generalized gapx-ADMM-GP training

DEC-gPoE Decentralized gPoE prediction

DEC-grBCM Decentralized grBCM prediction

DEC-NN-BCM Decentralized nearest neighbor BCM prediction

DEC-NN-gPoE Decentralized nearest neighbor gPoE prediction

DEC-NN-grBCM Decentralized nearest neighbor grBCM prediction

DEC-NN-NPAE Decentralized nearest neighbor NPAE prediction

DEC-NN-PoE Decentralized nearest neighbor PoE prediction

DEC-NN-rBCM Decentralized nearest neighbor rBCM prediction

DEC-NPAE Decentralized NPAE prediction

DEC-PoE Decentralized PoE prediction

DEC-rBCM Decentralized rBCM prediction

eMSE empirical Mean Square Error

eNLPD empirical Negative Log Predictive Density

FACT-GP Factorized GP training

FULL-GP Full GP training

g-FACT-GP Generalized Factorized GP training

gapx-GP generalized apx-ADMM-GP training

GLS Generalized Least Squares

GP Gaussian Process

gPoE generalized Product of GP Experts

GPS Global Positioning System

grBCM generalized robust Bayesian Committee Machine

JOR Jacobi Over-Relaxation

LNLL Local Negative Log-Likelihood

MCOK Multicollocated Ordinary Co-Kriging

MLE Maximum Likelihood Estimation

MSE Mean Square Error

MVE Mean Variance Error

NLL Negative Log-Likelihood

NLPD Negative Log Predictive Density

NPAE Nested Pointwise Aggregation of GP Experts

OK Ordinary Kriging

OLS Ordinary Least Squares

PD Positive Definite

PoE Product of GP Experts

PSD Positive Semi-Definite

px-ADMM proximal ADMM

rBCM robust Bayesian Committee Machine

RBF Radial Basis Function

REML Restricted Maximum Likelihood

RMSE Root Mean Square Error

SNR Signal-To-Noise Ratio

UK Universal Kriging

UWA Underwater Acoustic

WLS Weighted Least Squares

Chapter 1

Introduction

1.1 Motivation

Learning of Underwater Communication Performance

Coordination of multiple autonomous underwater agents requires effective communication for various cooperative missions [2]. For agents that operate underwater, inter-vehicle communication is usually accomplished using wireless underwater acoustic (UWA) signals. In the majority of the literature, wireless communication performance is treated as a deterministic, range-dependent function [15, 35, 87, 91, 92, 118, 139, 140]. In the graph theory literature this is also known as r-disk communication graph [14, 25, 60, 62, 63, 108, 133]. Indeed, communication performance is a function of vehicle range, but it is also dependent on many other environmental effects, including multi-path propagation and background noise [116]. In addition to the exchange of data, acoustic communication can also provide vehicle range information to improve navigation, as global positioning system (GPS) is unavailable in subsea environments [121].

Our aim is to predict UWA communication performance at unvisited locations using a set of communication performance measurements from nearby locations. We employ a two-step learning methodology that comprises: i) the estimation of covariance parameters and the statistical selection of a covariance function; and ii) the prediction of the communication

performance and its corresponding variance. Intuitively, the two-step process can be interpreted as first training from data, and then predicting the variable of interest at unvisited locations. The estimation of the covariance function and of its parameters merits special consideration, because it encodes the assumption on a stationary random field and generalizes the properties of the underlying latent process. Accurate predictions of anticipated communication performance can be exploited to plan better utilization of communication resources. Our general approach may be applicable to terrestrial networks, including aerial and ground communication using radio waves. The main idea is to leverage recent advances in spatial statistics and UWA communication modeling, to provide a realistic statistical prediction of inter-vehicle communication performance for teams of marine robots.

Decentralized Gaussian Processes

Teams of agents have received considerable attention in recent years, as they can address tasks that cannot be efficiently accomplished by a single entity. Multi-agent systems are attractive for their inherent property of collecting simultaneously data from multiple locations—a group of agents can collect more data than a single agent during the same time period. Central to machine learning (ML) methodologies is the collection of large datasets in order to ensure reliable training. To this end, networks of agents favor learning techniques, due to their data collection capabilities. However, they face major challenges including limited computational resources and communication restrictions. A typical approach to address these challenges relies on centralizing the collected data in a single node (e.g., cloud or data center), which requires high computational and storage resources. Yet, gathering data to a central server may lead to network traffic congestion and security or privacy issues. To ensure data privacy, a promising solution is federated learning (FL) [67]. FL aims to implement ML techniques in centralized or decentralized networks, but with no

communication of real data in order to comply to the EU/UK general data protection regulation (GDPR) [55]. For certain applications in GPS-denied environments, it is unfeasible to implement ML algorithms in a centralized network, as distant nodes may not be able to communicate directly with the central node due to communication range limitations or bandwidth. Such cases include autonomous vehicles and multi-robot systems. Finally, even if we manage to collect all the data in a central node, the time and space computational complexity for rapid updates of the ML models require resources that are not available to agents operating in the field. In this work, we propose methodologies for fully decentralizing Gaussian processes (GPs) [27, 43, 104] from training to prediction, so that they can be implemented efficiently on teams of agents. GPs are used in various multi-agent applications [3, 20, 23, 44, 52, 56, 61, 70, 71, 75, 97, 112, 119, 123, 137, 141, 143]. The major disadvantage of GPs is the poor scalability with the number of observations. Moreover, GPs are not easily decentralized for implementation across multiple agents due to high communication requirements.

Our objective is to develop fully decentralized approximate methodologies that relax the communication and computation requirements of GPs, exchanging as little information as possible and by performing only local computations. We propose three distributed optimization techniques to implement GP hyperparameter training with maximum likelihood estimation (MLE), based on the alternating direction method of multipliers (ADMM) [12]. Next, we synthesize 13 decentralized approximate methods to perform GP prediction with aggregation of GP experts [77], using iterative and consensus protocols [10, 93, 130].

1.2 Contributions

Learning of Underwater Communication Performance

The contribution is fourfold.

1. We formulate an approximate communication performance model that takes into account the environmental conditions. We use this simple model to motivate our specific approach to kriging, and to generate numerical simulations of communication performance that were used to exercise our framework.

2. We propose a bivariate approach to estimate the communication performance between two vehicles in a time-varying environment, by using cokriging.

3. After demonstrating that the communication performance is range dependent, we employ a realistic acoustic propagation model to formulate the problem as a non-stationary random field and propose model-based basis functions. Basis functions are then used to detrend the measurements and allow the implementation of stationary kriging.

4. We introduce an iterative technique to identify theoretical models that describe the unknown underwater acoustic environments. Since the covariance of the UWA propagation model is unknown, we compute the parameters of multiple theoretical covariance functions, and based on the Bayesian information criterion we select a theoretical model that fits best to the data. To this end, the iterative technique selects the most suitable theoretical covariance model for each environment.

Decentralized Gaussian Processes

The contribution is fivefold.

1. We extend a centralized GP training methodology [135] by devising augmented local datatsets to equip local entities, so that the hyper-parameter estimation accuracy of large-scale multi-agent systems is improved.

2. We introduce three decentralized GP training methods for strongly connected graph topologies and we derive a closed-form solution on the decentralized inexact ADMM [19] that reduces the computational requirements of local agents.

3. We decentralize the implementation of multiple aggregation of GP experts methods (PoE [51], gPoE [17], BCM [126], rBCM [30], and grBCM [76]) for strongly connected graph topologies, by using the discrete-time average consensus (DAC) [93].

4. We decentralize the implementation of NPAE [107] for strongly complete graph topologies, by combining Jacobi over-relaxation (JOR) [10, Chapter 2.4] and DAC. Moreover, we introduce a technique to recover the optimal relaxation factor of JOR [127] for strongly complete graph topologies by using the power method (PM) [40, Chapter 8]. The later ensures faster convergence.

5. We introduce a covariance-based nearest neighbor (CBNN) technique that selects statistically correlated agents for GP prediction on locations of interest, and provide a consistency proof. The CBNN is applicable to the decentralized versions of PoE, gPoE, BCM, rBCM, and grBCM introduced in 3). In addition, CBNN allows the use of a distributed algorithm to solve systems of linear equations (DALE) [78, 130] which replaces JOR in the decentralized NPAE of 4) and relaxes the graph topology from strongly complete to strongly connected.

1.3 List of Publications

Journal papers

[1] G. P. Kontoudis, D. J. Stilwell, "Fully Decentralized, Scalable Gaussian Processes for Collaborative Multi-Agent Learning–Part II: Prediction," 2022. *(in submission)*

[2] G. P. Kontoudis, D. J. Stilwell, "Fully Decentralized, Scalable Gaussian Processes for Collaborative Multi-Agent Learning–Part I: Training," 2022. *(in submission)*

[3] G. P. Kontoudis, S. Krauss, D. J. Stilwell, "Model-Based Learning of Underwater Acoustic Communication Performance for Marine Robots," *Robotics and Autonomous Systems **(RAS)***, 2021.

Conference papers

[1] G. P. Kontoudis, D. J. Stilwell, "Decentralized Nested Gaussian Processes for Multi-Robot Systems," *IEEE International Conference on Robotics and Automation **(ICRA)***, Xi'an, China, 2021.

[2] G. P. Kontoudis, D. J. Stilwell, "Prediction of Acoustic Communication Performance in Marine Robots Using Model-Based Kriging," *American Control Conference **(ACC)***, New Orleans, USA, 2021.

[3] G. P. Kontoudis, D. J. Stilwell, "A Comparison of Kriging and Cokriging for Estimation of Underwater Acoustic Communication Performance," *ACM International Conference on Underwater Networks and Systems **(WuWNet)***, Atlanta, USA, 2019.

1.4 Outline

The remainder of this dissertation is organized as follows. Chapter 2 discusses the related work, Chapter 3 focuses on model-based prediction of the UWA communication performance, Chapter 4 focuses on decentralized and scalable Gaussian processes for multi-agent systems, while Chapter 5 concludes the dissertation and discusses future directions.

Chapter 2

Literature Review

In this chapter, we present previous works in kriging methods for prediction of communication performance and distributed Gaussian processes (GPs).

2.1 Kriging for Communication Performance

The importance of communication in multi-robot systems was discussed in [6]. The authors investigated the importance of communication in three types of missions with simulations and experiments. Indeed, in several cases inter-vehicle communication improved the performance of the mission. Although communication is evidently of crucial importance for the success of multi-robot missions, communication cannot be always guaranteed for multiple reasons. A survey of prospects and problems in UWA communications is documented in [72]. Since acoustic waves demonstrate relatively low absorption in subsea environments, they are the major mode of wireless underwater communication. In underwater wireless sensor networks, kriging (equivalent to Gaussian processes [43, 104]) has been used to model communication performance in several applications. Horner *et al.* [54], proposed a methodology based partially on ordinary kriging for the generation of local and global acoustic communication performance maps to facilitate collaborative navigation. A distributed kriging methodology was used in [128] to estimate coverage holes in large-scale wireless sensor networks. The authors in [134] developed a cooperative robust algorithm to compose a spatial map of un-

derwater acoustic communication signals and channel parameters using an H_∞ filter and ordinary kriging. In [122], the acoustic communication performance of micro autonomous underwater vehicles (AUVs) was assessed with field trials. The results of the latter reveal that for non-stationary transmission, i.e. moving vehicle, several factors reduce communication performance, including multi-path effect of acoustic transmission and the Doppler effect. In [117], a methodology that combines ordinary kriging and compressive sensing methods, was utilized for prediction of acoustic intensity. Prediction of communication performance has been addressed for radio applications. In [81], the authors employ maximum-likelihood estimation for the parameters of the covariance matrix, logarithmic transformation for the underlying mean towards a model-based approach, and compressive sensing for prediction with sparse data. In addition, they show that the location of measurements may improve the prediction quality. In [4], the authors proposed an ordinary kriging prediction framework with detrended data to build radio environment maps and they also considered positional error of the measurements. Gaussian processes have also been used to build communication maps of known terrestrial environments with multiple agents [73]. Specifically the authors used a Gaussian process with constant mean value [104, (2.38), p.27] (equivalent to ordinary kriging) and squared exponential covariance function. Their methodology uses communication priors based on four communication path-loss models to reduce the uncertainty of the communication maps. In the same spirit, in [64] a Gaussian process with fixed mean function and a squared exponential covariance function is proposed to predict the WiFi channel quality and find the optimal relay position for mobile networks. Ordinary kriging assumes that the underlying process is stationary. In addition, in all of these works it was assumed that the covariance model follows a specific theoretical covariance function. In our work, we formulate the problem as a non-stationary random field with universal kriging, which is equivalent to GPs with model-based fixed basis functions [104, (2.41), p.28]. Moreover, we investigate multiple theoretical models for the statistical selection of the covariance function.

Communication performance estimation can be used to estimate the position of a vehicle. In [46], the authors employed Gaussian processes to determine a likelihood model of the received signal strength (RSS) for WiFi to estimate the location of robots. This approach requires to compare a training set of RSS observations to a ground truth map, yet this is a computationally demanding process for large maps. To alleviate the computational burden, the authors in [34] used a Gaussian process latent variable model (GP-LVM) to: i) generate the RSS map, ii) compute the position of the vehicle, and iii) build the seafloor map. In these works, only the RSS measurements were used for the construction of RSS maps. In our work, we also use the distance between communicating vehicles to build basis functions for detrending of non-stationary processes.

Adaptive sampling is another cooperative application of AUVs to monitor and model the environment. Unambiguously, the prediction of underwater communication performance is critical for the efficiency of subsea adaptive sampling missions. In [42], the authors survey methodologies to connect hierarchical spatio-temporal techniques [7] with distributed algorithms. A review of distributive adaptive sampling of mobile agents for spatio-temporal processes is listed in [95]. A sub-sampling method was proposed in [41] to alleviate the computational efforts. Then, the authors use kriging to map a terrain at higher resolution. This map is used to plan paths for unmanned and manned vehicles based on three cost functions. Kriging has been used in adaptive sampling for the statistical modeling of the environment, yet without a rigorous learning method for estimating the covariance that addresses proper covariance model selection, robustness, and bias correction.

2.2 Decentralized Gaussian Processes

Despite their effectiveness in function approximation and uncertainty quantification, GPs scale poorly with the number of observations. Particularly, provided N observations, the training entails $\mathcal{O}(N^3)$ computations and the prediction requires $\mathcal{O}(N^2)$ computations. Another limitation for the implementation of GPs in multi-agent systems is the communication. For centralized GPs, every agent has to communicate all observations to a central node. However, excessive communication is challenging in decentralized networks. Moreover, agents in networks can pass messages only within a communication range [14] which may vary in space and time [68].

To overcome the computational burden of hyper-parameter GP training with maximum likelihood estimation (MLE), a factorized GP training method is discussed in [30, 90]. That is a centralized method which is based on a server-client structure and distributes the computations to multiple entities. The main idea is to assume independence between sub-models, which results in the approximation of the inverse covariance matrix by the inverse of a block diagonal matrix. To this end, a significant reduction in computation of the inverse of multiple covariance matrices is achieved at the cost of excessive communication overhead. More specifically, every local entity transmits multiple inverted blocks of the covariance matrix per MLE iteration. Recently, Xu *et al.* [136] reformulated the factorized GP training method using the exact consensus alternating direction method of multipliers (ADMM) [12], which is appealing in centralized multi-agent settings [47]. Consensus ADMM reduces the communication overhead of GP training, but requires high computational resources to solve a nested optimization problem at every ADMM-iteration. Subsequently, the authors in [135] employed the inexact proximal ADMM [53] to alleviate the computation demand. However, both ADMM-based factorized GP training methods require a centralized network topology.

Two major research directions for GP prediction are based on global and local approximations [77]. Global approximation methods promote sparsity by using either a subset of N_{sub} observations or by introducing a set of N_{sub} pseudo-inputs, where $N_{\text{sub}} \ll N$ [50, 101, 113]. Sparse GPs have been used in mobile sensor networks to model spatial fields [44]. In [137], a GP with truncated observations in a mobile sensor network is proposed, and in [20] a subset of observations is used for traffic modeling and prediction. These methods require global knowledge of the observations, which increases inter-agent communications. Additionally, the methods that utilize pseudo-inputs do not retain the interpolation property.

Alternatively, the second research direction uses local approximation methods to reduce the computational burden of GP prediction. These are centralized algorithms with a server-client structure. The main idea is to aggregate local sub-models produced by local subsets of the observations [17, 30, 51, 126]. In other words, every sub-model makes a local prediction, and then the central node aggregates to a single prediction. In comparison to global approximations, local methods do not require inducing inputs, they distribute the computational load to multiple agents, and they work with all observations. However, it is proved in [5, Proposition 1] that the local methods [30, 51, 126] are *inconsistent*, i.e. as the observation size grows to infinity, the aggregated predictions do not converge to the true values. Subsequently, the authors in [107] proposed the nested point-wise aggregation of experts (NPAE) that takes into account the covariance between sub-models and produces consistent predictions. The price to achieve consistency in NPAE comes with much higher computational complexity in the central node. Liu *et al.* [76] introduced a computationally efficient and consistent methodology, termed as generalized robust Bayesian committee machine (grBCM). The latter entails additional communication between agents to enrich local datasets with a global random dataset. In addition, both NPAE and grBCM are centralized techniques, that are not well-suited for multi-agent systems [14].

A decentralized method for the computation of spatio-temporal GPs is proposed in [25]. In [23], a decentralized technique for spatial GPs with localization uncertainty is presented. Both [25] and [23] employ the Jacobi over-relaxation (JOR), which requires a strongly complete graph topology, i.e. every node must communicate to every other node. That is a conservative topology and is not common in mobile sensor networks [14]. Essentially, for not strongly complete topologies, JOR entails flooding before every iteration. In flooding each agent broadcasts all input packets to its neighbors [125]. Thus, the communication requirements of JOR are high. Yuan and Zhu [141, 142], proposed a methodology that combines nearest neighbor GPs [29] and local approximation [17]. Although [17] is consistent in terms of prediction mean, it produces overconfident prediction variances [76, Proposition 1]. In addition, arbitrary selection of nearest neighbor sets may lead to poor approximations [29] and suffers from prediction discontinuities [107]. Pillonetto *et al.* [97] proposed sub-optimal methods to distributively estimate a latent function with a GP by employing orthonormal eigenfunctions, computed by the Karhunen-Loève expansion of a kernel. An extension of this work to multi-robot systems with online information gathering is discussed in [56]. This is a promising line of research for GPs in decentralized networks, but our focus is on decentralized and scalable GP training with MLE and GP prediction with aggregation methods. Nevertheless, computing orthonormal eigenfunctions in closed-form is not feasible for all kernels and may yield significant storage requirements.

2.3 Summary

Many research groups proposed prediction methods of UWA communication performance. However, they assumed that the random field is stationary and follows a specific covariance function [4, 54, 73, 81, 117]. We are particularly interested in non-stationary random fields

with an unknown covariance function. The next topic of distributed GPs is of paramount importance for multi-robot exploration and navigation. Multiple studies suggested centralized methodologies with local computations [17, 30, 51, 76, 107, 126]. A few body of research works is focused on the decentralization of GPs using local observations [23, 25, 141]. Yet, the latter methods require strongly complete network topologies and/or excessive communication. In this dissertation, we focus on decentralized GPs of realistic network topologies and with as little communication as possible.

Chapter 3

Model-Based Kriging

In this chapter, we present a comparison of kriging and cokriging that explores the effect of the vehicle range variable to the prediction of underwater acoustic communication performance. After demonstrating that vehicle range is of paramount importance for the prediction, we exploit a realistic underwater acoustic propagation model to compose a model-based kriging technique with particular emphasis on the estimation of the covariance matrix.

3.1 Comparison Kriging and Cokriging

3.1.1 Problem Formulation

In this section we present the measurement model of the vehicles and we discuss the physical process of the environment. We also assess the acoustic communication performance with a signal-to-noise ratio (SNR) model of the sonar.

3.1.2 Communication Performance

The measurement model of all agents is identical and described by,

$$Y_i(\mathbf{x};t) = Z(\mathbf{x};t) + \epsilon, \tag{3.1}$$

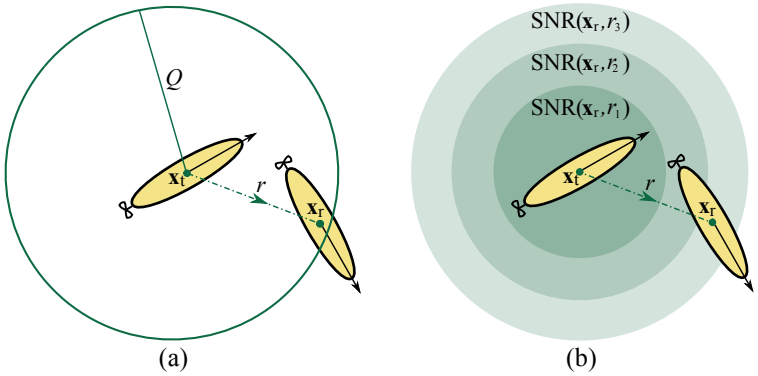


Figure 3.1: Communication scenarios of two autonomous underwater vehicles (AUVs) at range r. The transmitting vehicle is located at position $\mathbf{x}_t$ and the receiving vehicle at position $\mathbf{x}_r$. (a) The communication success relies on a deterministic maximum communication range Q. (b) The communication performance using signal-to-noise ratio (SNR) is predicted for specific vehicle ranges.

where $Y_i(\mathbf{x};t)$ is the measurement of communication performance of agent i at spatial locations $\mathbf{x} = [x \quad y]^\intercal \in \mathbb{R}^2$, $Z(\mathbf{x};t)$ represents the random field, and $\epsilon \sim (0, \sigma_Y^2)$ is a zero-mean Gaussian noise.

We seek a simple model of underwater acoustic communication performance. We employ the passive sonar equation that models direct communication between the transmitter and the receiver [33, 57]. Unlike an active sonar model, we do not consider interaction with a target system e.g., reverberation noise. Since we are interested in applications with relatively slow-moving AUVs, we ignore frequency shifting and spreading that are due to motion-induced Doppler effect.

To approximate the communication performance between two agents we use the SNR. In principle, the higher the SNR, the more likely is to detect the transmitted signal. The passive sonar equation is expressed,

$$\text{SNR} = \text{SL} - \text{TL} - \text{NL} + \text{DI}, \tag{3.2}$$

where SL is the source level, TL is the transmission loss, NL is the noise level, and DI is the directivity index. In practice, the source level is provided by the manufacturer of the transmitter and we assume that the effect of the directivity index is negligible, similarly to [111]. The transmission loss can be computed as,

$$\mathrm{TL}(r) = \mathrm{TL}_{\mathrm{sph}}(r) - \mathrm{TL}_{\mathrm{a}}(r), \tag{3.3}$$

where $\mathrm{TL}_{\mathrm{sph}}$ is the spherical spreading loss, TL_{a} is the attenuation, and $r = \|\mathbf{x}_{\mathrm{r}} - \mathbf{x}_{\mathrm{t}}\|_2$ is the range of two vehicles. In Fig. 3.1 we illustrate the case of acoustic communication between two underwater vehicles at range r, with $\mathbf{x}_{\mathrm{t}}$ the position of the *transmitting* vehicle and $\mathbf{x}_{\mathrm{r}}$ the position of the *receiving* vehicle. Spherical spreading loss is proportional to the log of range, $\mathrm{TL}_{\mathrm{sph}}(r) = 20 \log r$. Attenuation depends on the signal frequency due to the process of transferring the acoustic energy into heat. More specifically, for a signal frequency of $f = 25$ kHz the absorption coefficient is $a = 5.56$ dB/km [13]. Thus, (3.3) results in a linear-log relationship,

$$\mathrm{TL}(r) = 20 \log r - 0.00556r. \tag{3.4}$$

Environmental Conditions

In our simplified communication model, we capture various environmental effects, such as multi-path, density gradients, etc, as simply noise that reduces the SNR. The noise comprises of ambient noise, transient noise, and self-noise [33].

Sources of ambient noise include the shipping and sea state. Ambient noise is approximated by the Wenz curves [132],

$$\mathrm{NL}_{\mathrm{amb}} = \mathrm{NL}_{\mathrm{ship}} \oplus \mathrm{NL}_{\mathrm{SS}}, \tag{3.5}$$

where $\mathrm{NL}_{\mathrm{ship}}$ is the shipping noise and $\mathrm{NL}_{\mathrm{SS}}$ is the sea state noise. The power summation

operator for L_k elements, with $k = 1, \ldots, N_k$, is given by $\oplus = 10 \log \sum_{k=1}^{N_k} 10^{L_k/10}$. For a signal frequency of $f = 25$ kHz the shipping noise is almost zero, as $\mathrm{NL_{SS}} \gg \mathrm{NL_{ship}}$. To this end, (3.5) simplifies to $\mathrm{NL_{amb}} = \mathrm{NL_{SS}}$.

Subsequently, if we neglect the transient noise (e.g, biological organisms) and self-noise the communication performance yields,

$$\mathrm{SNR} = \mathrm{SL} - 20 \log r + 0.00556r - \mathrm{NL_{SS}}. \tag{3.6}$$

Remark 3.1. Since the communication signal transmits in high frequency ($f = 25$ kHz), the transient noise can be neglected. Similarly, the cavitation noise of the propeller vanishes. However, the flow noise—which is produced by the propeller—may affect the source level of the transmitted signal and/or the received signal strength. In fact, this will lead to anisotropic SNR, depending not only on the position but also on the orientation of the vehicle. In this work, we do not consider anisotropic sensing.

3.1.3 Multivariate Spatial Estimation

In this section, we introduce kriging, a spatial estimation technique that estimates values at locations of interest, based on measurements from other locations. First, we discuss the ordinary kriging (OK) and then we present the multivariate kriging, namely cokriging (COK).

Let us first introduce some basic notions of the random fields. A comprehensive discussion on the topic can be found in [27]. Let $Z(\mathbf{x})$ be a *random field* with a positive-definite covariance matrix $\mathrm{Cov}(Z(\mathbf{x}_1), Z(\mathbf{x}_2)) \succ 0$ for all $\mathbf{x} \in \mathbb{R}^2$. The random field is *intrinsically stationary* if $\mathrm{Cov}(Z(\mathbf{x}_1), Z(\mathbf{x}_2)) = C(\mathbf{x}_1 - \mathbf{x}_2)$ for all $\mathbf{x} \in \mathbb{R}^2$ and the function $C(\cdot)$ is called *covariogram*. An intrinsically stationary random field with a constant mean is called *second-*

order stationary. The semivariogram of a second-order stationary process with constant mean $\mathrm{E}\{Z(\mathbf{x})] = \mu$ is defined,

$$\gamma(\mathbf{x}_1, \mathbf{x}_2) \coloneqq \frac{1}{2}\mathrm{E}\{(Z(\mathbf{x}_1) - Z(\mathbf{x}_2))^2\} = \frac{1}{2}\mathrm{Var}[Z(\mathbf{x}_1) - Z(\mathbf{x}_2)]. \tag{3.7}$$

Moreover, if $C(\mathbf{x}_1 - \mathbf{x}_2)$ is only a function of the Euclidean norm $\|\mathbf{x}_1 - \mathbf{x}_2\|_2$, then the covariogram is *isotropic*. The *correlogram* is defined,

$$\rho(\mathbf{x}) \coloneqq \frac{C(\mathbf{x})}{C(0)}, \tag{3.8}$$

where $C(0) = \mathrm{Var}[Z(\mathbf{x})]$ is the *sill* and the data is normalized so that it has zero mean an unit variance (see (3.37)). For a second-order stationary random field with normalized measurements and $\|\mathbf{x}_1 - \mathbf{x}_2\|_2 = h$, the semivariogram is the mirror image of the covariance, resulting,

$$\gamma(h) = 1 - C(h). \tag{3.9}$$

Next, we present fundamental notions of the multivariate case [129]. In multivariate statistics the covariance comprises of direct and cross-covariance functions. The *joint second-order* hypothesis assumes a constant mean for every variable,

$$\mathrm{E}[Z_j(\mathbf{x})] = \mu_j, \tag{3.10}$$

and a cross-covariance function in the form,

$$\mathrm{E}[(Z_j(\mathbf{x}_1) - \mu_i)(Z_l(\mathbf{x}_2) - \mu_l)] = C_{jl}(h). \tag{3.11}$$

The cross-covariance function C_{jl} captures the variation of variables over distance. The *joint*

intrinsic model imposes the cross-variogram structure,

$$\gamma_{jl}(\mathbf{x}_1, \mathbf{x}_2) = \frac{1}{2}\mathrm{E}[(Z_j(\mathbf{x}_1) - Z_j(\mathbf{x}_2))(Z_l(\mathbf{x}_1) - Z_j(\mathbf{x}_2))]. \tag{3.12}$$

That is, the cross-variogram measures the difference of variances over distance. Furthermore, the cross-correlogram, by assuming the *intrinsic correlation model*, is expressed,

$$\rho_{jl}(h) = \frac{C_{jl}(h)}{C_j(0)C_l(0)}, \tag{3.13}$$

where $C_j(0) = \mathrm{Var}[Z_j(\mathbf{x})]$, $C_l(0) = \mathrm{Var}[Z_l(\mathbf{x})]$ are the sills where for normalized measurements $C_j(0) = C_l(0) = 1$.

Ordinary Kriging

Let us now describe the ordinary kriging technique. We consider multiple measurements at locations $\mathbf{x}_j \in \mathbb{R}^2$, $j = 1, \ldots, M$ with $M \in \mathbb{N}$. In ordinary kriging the Gaussian random field is modeled as,

$$Z(\mathbf{x}) = \mu + \nu(\mathbf{x}), \tag{3.14}$$

where $Z(\mathbf{x}) \in \mathbb{R}$ is a second-order stationary random field, $\mu \in \mathbb{R}$ is the unknown constant mean that represents the large scale variation, and $\nu(\mathbf{x})$ is the zero-mean Gaussian field that captures the medium scale variability. We are interested in estimating the mean value of the random field at an unmeasured location $\mathbf{x}_0$, based on the measured data $Z(\mathbf{x})$. We use

a linear unbiased estimator,

$$\begin{aligned}\hat{Z}(\mathbf{x}_0) &= \sum_{j=1}^{N_j} \beta_j Z(\mathbf{x}_j) + (1 - \sum_{j=1}^{N_j} \beta_j)\mu \\ &= \boldsymbol{\beta}^\intercal \mathbf{Z}(\mathbf{x}),\end{aligned} \tag{3.15}$$

where $\boldsymbol{\beta} = [\beta_1 \ \dots \ \beta_{N_j}]^\intercal \in \mathbb{R}^{N_j}$ are the weights we seek to obtain. The unbiasedness of the estimator $\sum_{j=1}^{N_j} \beta_j = 1$ relaxes the assumption of a known global mean μ. As a result, we can perform kriging with the measurements and not its residuals, $Z(\mathbf{x}_j) - \mu$. Next, we formulate the unconstrained minimization problem with a Lagrange multiplier λ_{OK} to include the unbiasedness constraint. The solution to the minimization problem results in,

$$\boldsymbol{\beta}_{\text{OK}} = \boldsymbol{\Gamma}_{\text{OK}}^{-1} \boldsymbol{\gamma}_{\text{OK}}, \tag{3.16}$$

where $\boldsymbol{\beta}_{\text{OK}} = [\boldsymbol{\beta}^\intercal \ \lambda_{\text{OK}}]^\intercal \in \mathbb{R}^{N_j+1}$ is a vector that contains the weights $\boldsymbol{\beta}$ and the Lagrange multiplier λ_{OK}. The non-singular matrix $\boldsymbol{\Gamma}_{\text{OK}} \in \mathbb{R}^{(N_j+1)\times(N_j+1)}$ considers the *redundancy* of measurements and is given by,

$$\boldsymbol{\Gamma}_{\text{OK}} = \begin{bmatrix} \gamma(\mathbf{x}_1, \mathbf{x}_1) & \dots & \gamma(\mathbf{x}_1, \mathbf{x}_N) & 1 \\ \vdots & \ddots & \vdots & \vdots \\ \gamma(\mathbf{x}_N, \mathbf{x}_1) & \dots & \gamma(\mathbf{x}_N, \mathbf{x}_N) & 1 \\ 1 & \dots & 1 & 0 \end{bmatrix} \coloneqq \begin{bmatrix} \boldsymbol{\Gamma} & \mathbf{1} \\ \mathbf{1}^\intercal & 0 \end{bmatrix}, \tag{3.17}$$

where $\mathbf{1} \in \mathbb{R}^{N_j}$ is a vector of ones. The vector $\boldsymbol{\gamma}_{\text{OK}} \in \mathbb{R}^{(N_j+1)}$ takes into account the *closeness* of the measurements to the location of interest $\mathbf{x}_0$ and yields,

$$\boldsymbol{\gamma}_{\text{OK}} = \begin{bmatrix} \gamma(\mathbf{x}_0, \mathbf{x}_1) \\ \vdots \\ \gamma(\mathbf{x}_0, \mathbf{x}_N) \\ 1 \end{bmatrix} := \begin{bmatrix} \boldsymbol{\gamma}_0 \\ 1 \end{bmatrix}. \tag{3.18}$$

The unique solution of (3.16) yields the vector of unknown weights,

$$\boldsymbol{\beta} = \boldsymbol{\Gamma}^{-1}\Big(\boldsymbol{\gamma}_0 - \mathbf{1}\lambda_{\text{OK}}\Big), \tag{3.19}$$

and the Lagrange multiplier,

$$\lambda_{\text{OK}} = \frac{\mathbf{1}^{\intercal}\boldsymbol{\Gamma}^{-1}\boldsymbol{\gamma}_0 - 1}{\mathbf{1}^{\intercal}\boldsymbol{\Gamma}^{-1}\mathbf{1}}, \tag{3.20}$$

Sequentially, the weights $\boldsymbol{\beta}$ and the Lagrange multiplier λ_{OK} can be used for the computation of the ordinary kriging variance as,

$$\sigma^2_{\text{OK}}(Z(\mathbf{x}_0)) = \text{Var}_{\text{OK}}[Z(\mathbf{x}_0)] = \boldsymbol{\beta}^{\intercal}\boldsymbol{\gamma}_0 + \lambda_{\text{OK}}. \tag{3.21}$$

In terms of the covariance matrix for normalized measurements, we use (3.9) and the solution follows accordingly.

Multicollocated Ordinary Cokriging

In this section we shall describe the multicollocated ordinary cokriging (MCOK). We observe in (3.6) that our simplified model of communication performance is a linear-log function of the range of the vehicles. Moreover, the range measurements are acquired simultaneously

with the SNR. Interestingly, there exists a spatial correlation of these two variables. For instance, when we seek to estimate the communication performance at a specific location, the range of the vehicles is critical. In case that the vehicles navigate in close proximity, then the communication performance is expected be high. On the contrary, in case that the vehicles have large range, then the communication signal will be degraded and corrupted by noise. Therefore, we want to estimate the communication performance at a specific location for a given range.

Cokriging is the multivariate kriging that augments the estimation process with the covariances and cross-covariances of the variables involved in the process [129]. The key idea underlying this work is to use the range of the vehicles as a secondary variable in cokriging in order to improve the SNR estimation. Thus, we incorporate two variables: i) the communication performance as the primary variable and ii) the range of the vehicles as the secondary variable. The ordinary cokriging estimator for two variables yields,

$$\begin{aligned}\hat{Z}(\mathbf{x}_0) &= \sum_{j=1}^{N_j} \beta_{j,1} Z_1(\mathbf{x}_j) + \sum_{l=1}^{N_l} \beta_{l,2} Z_2(\mathbf{x}_l) \\ &= \boldsymbol{\beta}_{\mathrm{COK},1}^{\intercal} \mathbf{Z}_1(\mathbf{x}) + \boldsymbol{\beta}_{\mathrm{COK},2}^{\intercal} \mathbf{Z}_2(\mathbf{x}),\end{aligned} \tag{3.22}$$

where $\boldsymbol{\beta}_{\mathrm{COK},1} = [\beta_{1,1}, \ldots, \beta_{N_j,1}]^{\intercal}$, $\boldsymbol{\beta}_{\mathrm{COK},2} = [\beta_{1,2}, \ldots, \beta_{N_l,2}]^{\intercal}$ are the stacked vectors of the unknown weights of two variables, $\mathbf{Z}_1 \in \mathbb{R}^{N_j}$ and $\mathbf{Z}_2 \in \mathbb{R}^{N_l}$ with $N_l > N_j$ are the stacked vectors of the measurements of the two variables at locations $\mathcal{X}_{\mathrm{pr}} = \{\mathbf{x}_j\}_{j=1}^{N_j}$ and $\mathcal{X}_{\mathrm{sec}} = \{\mathbf{x}_l\}_{l=1}^{N_l}$ respectively. The unbiasedness of the estimator for the primary variable $\mathbf{1}^{\intercal}\boldsymbol{\beta}_{\mathrm{COK},1} = 1$ and for the secondary variable $\mathbf{1}^{\intercal}\boldsymbol{\beta}_{\mathrm{COK},2} = 0$, relaxes the assumption of known global means. Therefore, we implement cokriging with the measurements and not its residuals. Then, we formulate the unconstrained minimization problem with two Lagrange multipliers to account for the unbiasedness constraints $\lambda_{\mathrm{COK},1}$, $\lambda_{\mathrm{COK},2}$. The solution to the minimization problem

results in the system of linear equations,

$$\boldsymbol{\beta}_{\mathrm{COK}} = \boldsymbol{\Gamma}_{\mathrm{COK}}^{-1}\boldsymbol{\gamma}_{\mathrm{COK}}, \tag{3.23}$$

where $\boldsymbol{\beta}_{\mathrm{COK}} = [\boldsymbol{\beta}_{\mathrm{COK},1}^{\intercal}\ \boldsymbol{\beta}_{\mathrm{COK},2}^{\intercal}\ \lambda_{\mathrm{COK},1}\ \lambda_{\mathrm{COK},2}]^{\intercal} \in \mathbb{R}^{N_j+N_l+2}$ is the unknown vector we seek to obtain. The non-singular matrix $\boldsymbol{\Gamma}_{\mathrm{COK}} \in \mathbb{R}^{(N_j+N_l+2)\times(N_j+N_l+2)}$ captures the measurement redundancy and has the form of,

$$\boldsymbol{\Gamma}_{\mathrm{COK}} = \begin{bmatrix} \boldsymbol{\Gamma_1} & \boldsymbol{\Gamma_{12}} & \mathbf{1} & \mathbf{0} \\ \boldsymbol{\Gamma_{21}} & \boldsymbol{\Gamma_2} & \mathbf{0} & \mathbf{1} \\ \mathbf{1}^{\intercal} & \mathbf{0}^{\intercal} & 0 & 0 \\ \mathbf{0}^{\intercal} & \mathbf{1}^{\intercal} & 0 & 0 \end{bmatrix}. \tag{3.24}$$

The vector $\boldsymbol{\gamma}_{\mathrm{COK}} \in \mathbb{R}^{(N_j+N_l+2)}$ considers the closeness of the measurements to the location of interest and leads to,

$$\boldsymbol{\gamma}_{\mathrm{COK}} = \begin{bmatrix} \boldsymbol{\gamma_{0,1}} \\ \boldsymbol{\gamma_{0,12}} \\ 1 \\ 0 \end{bmatrix}. \tag{3.25}$$

In general, the practical challenges with cokriging are: i) the modeling of all covariances and cross-covariances, ii) all covariances and cross covariances jointly need to be positive definite, and iii) the solution generates very large linear systems, i.e. $(N_j + N_l + 2)$-equations. For these reasons, we employ the multicollocated cokriging which accounts for: i) all primary variable measurements, ii) all secondary variable measurements at the locations of the primary variable measurements, and iii) the secondary variable measurement at the location of interest, as shown in Fig. 3.2. The orphan secondary variable measurements $\mathcal{X}_{\mathrm{orp}} = \mathcal{X}_{\mathrm{pr}} \backslash \mathcal{X}_{\mathrm{sec}}$,

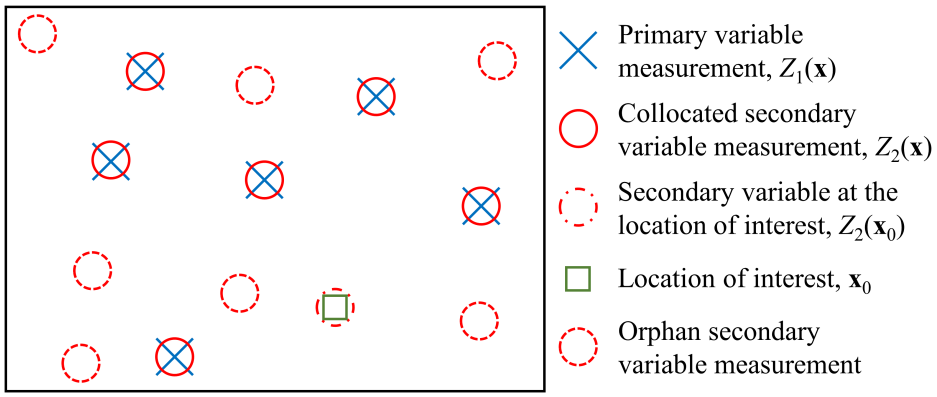


Figure 3.2: The multicollocated setup. The primary variable measurements $Z_1(\mathbf{x})$ are shown in blue x-marks, the collocated secondary variable measurements $Z_2(\mathbf{x})$ are depicted with red solid circles, the secondary variable measurement at the location of interest $Z_2(\mathbf{x}_0)$ is shown in red dash-dotted line, and the location of interest $\mathbf{x}_0$ is presented with a green rectangular. The dashed red circle represent the orphan secondary variable measurements $\mathcal{X}_{\text{orp}}$ that are not used in the multicollocated cokriging.

i.e. not collocated with primary variable measurements, are not used in this framework. The multicollocated cokriging model (or Markov Model 2) has been proven to be necessary and sufficient for cokriging in the stationary case [58, 106]. Next, we introduce the Markov screening and the Bayesian updating assumptions.

Assumption 3.2 (Markov Screening)**.** *The primary variable Z_1 at any location $\boldsymbol{x}_1$ depends conditionally only on the secondary variable Z_2 at location $\boldsymbol{x}_1$, screening out the influence of the secondary variable Z_2 at any other location $\boldsymbol{x}_2$, which yields,*

$$E[Z_1(\boldsymbol{x}_1) \mid Z_2(\boldsymbol{x}_1), Z_2(\boldsymbol{x}_2)] = E[Z_1(\boldsymbol{x}_1) \mid Z_2(\boldsymbol{x}_1)]. \tag{3.26}$$

Assumption 3.3 (Bayesian Updating)**.** *The primary and the secondary variables are linearly related through the correlation coefficient $\rho_{12}(0)$ at any location, which yields,*

$$E[Z_1(\boldsymbol{x}) \mid Z_2(\boldsymbol{x})] = \rho_{12}(0) Z_2(\boldsymbol{x}). \tag{3.27}$$

From Assumption 3.2 and Assumption 3.3 the cross-correlogram takes the form,

$$\rho_{12}(\mathbf{h}) = \rho_{12}(0)\rho_2(\mathbf{h}), \tag{3.28}$$

which in terms of covariogram yields,

$$\gamma_{12}(\mathbf{h}) = p\gamma_2(\mathbf{h}), \tag{3.29}$$

where $p = \rho_{12}(0)\sigma_1/\sigma_2$ is the slope of the linear regression with σ_1, σ_2 the standard deviations of the primary and secondary variables respectively. Note that if the measurements are normalized with respect to the variance, then $\sigma_1 = \sigma_2 = 1$ and subsequently $p = \rho_{12}$. Next, we consider a regression model of the primary variable on the secondary variable in the form,

$$Z_1(\mathbf{x}) = pZ_2(\mathbf{x}) + R(\mathbf{x}), \tag{3.30}$$

where $R(\mathbf{x})$ is the orthogonal residual which can also be considered as $R(\mathbf{x}) = Z_1(\mathbf{x}) - pZ_2(\mathbf{x})$. Note that since $Z_1(\mathbf{x})$ and $Z_2(\mathbf{x})$ are Gaussian, $R(\mathbf{x})$ is also Gaussian.

Assumption 3.4 (Residual Independence)**.** *The residual $R(\boldsymbol{x})$ is an independent random function of the secondary variable at any location, which yields,*

$$Cov(R(\boldsymbol{x}), Z_2(\boldsymbol{x})) = 0. \tag{3.31}$$

Due to Proposition 3.4, the linear regression (3.30) maintains the *homoscedasticity* properties of kriging, i.e. the variance of the primary variable can be computed at locations of interest, without actual measurement of the primary variable at this location.

The orthogonal residual can be computed with the ordinary kriging as discussed in Subsec-

tion 3.1.3 with a linear unbiased estimator in the form,

$$\hat{R}(\mathbf{x}_0) = \boldsymbol{\beta}_{\mathrm{R}}^{\intercal} R(\mathbf{x}), \tag{3.32}$$

where $\boldsymbol{\beta}_{\mathrm{R}}$ are the residual corresponding weights of the ordinary kriging. Note that the domain of measurements for the ordinary kriging of the orthogonal residual, does not include the location of interest $\mathcal{D}_{\mathrm{x}} = \mathcal{X}_{\mathrm{pr}} \cup \mathcal{X}_{\mathrm{sec}} \not\ni \mathbf{x}_0$. Then, we use the residual variogram function γ_{R} to construct the covariance of the primary variable as,

$$\gamma_1(h) = p^2 \gamma_2(h) + \gamma_{\mathrm{R}}(h). \tag{3.33}$$

The rest elements of the non-singular matrix $\boldsymbol{\Gamma}_{\mathrm{MCOK}}$ result from (3.29) and the experimental variogram of the secondary variable. The multicollocated ordinary cokriging estimator for two variables yields,

$$\begin{aligned} \hat{Z}_1(\mathbf{x}_0) &= p Z_2(\mathbf{x}_0) + \hat{R}(\mathbf{x}_0) \\ &= \sum_{j=1}^{N_j} \beta_{\mathrm{R},j} Z_{1,j} + p\Big(Z_2(\mathbf{x}_0) - \sum_{l=1}^{N_l - 1} \beta_{\mathrm{R},l} Z_{2,l} \Big). \end{aligned} \tag{3.34}$$

where $Z_2(\mathbf{x}_0)$ is the measurement of the secondary variable measurement at the location of interest and $N_j = |\mathcal{D}_{\mathrm{x}}|$. The corresponding variance yields,

$$\sigma^2_{\mathrm{MCOK}}(Z_1(\mathbf{x}_0)) = \mathrm{Var}_{\mathrm{MCOK}}[Z_1(\mathbf{x}_0)] = \mathrm{E}[\hat{R}(\mathbf{x}_0) - R(\mathbf{x}_0)]. \tag{3.35}$$

Remark 3.5. The multicollocated cokriging estimation (3.34) does not require the cross-covariance function and also results in a significantly smaller system of equations. To this end, we just need to compute the ordinary kriging of the residual R that comprises of

$(N_j + 1)$-equations and retain the same properties of solving the ordinary cokriging that consists of $(N_j + N_l + 2)$-equations, with $N_l > N_j$. This constitutes a significant reduction in the computational effort of the proposed technique.

Remark 3.6. In Proposition 3.3 we considered a linear relation of the primary with the secondary variable. However, according to (3.6) the communication performance is linear-logarithmically related with the range of the vehicles. Therefore, we expect smoother estimation results than the ground truth values.

3.1.4 Spatial Estimation Framework

In this section, we discuss the structure of the proposed communication performance estimation with multicollocated cokriging and the computational complexity of both kriging and cokriging.

Estimation Structure

The multicollocated ordinary cokriging is shown in Fig. 3.3. The structure consists of collecting the measurements; normalizing the measurements; computing the correlation factor and the orthogonal residual; kriging the residual; and estimating the communication performance at the unknown location.

We start by collecting measurements of communication performance (SNR) and the range of the vehicles. SNR is the primary variable Z_1 and range the secondary Z_2. Then, we normalize the measurements with respect to the variance,

$$\tilde{Z}_{\delta,j} = \frac{Z_{\delta,j} - \mu_\delta}{\sqrt{\text{Var}[Z_\delta]}}, \tag{3.36}$$

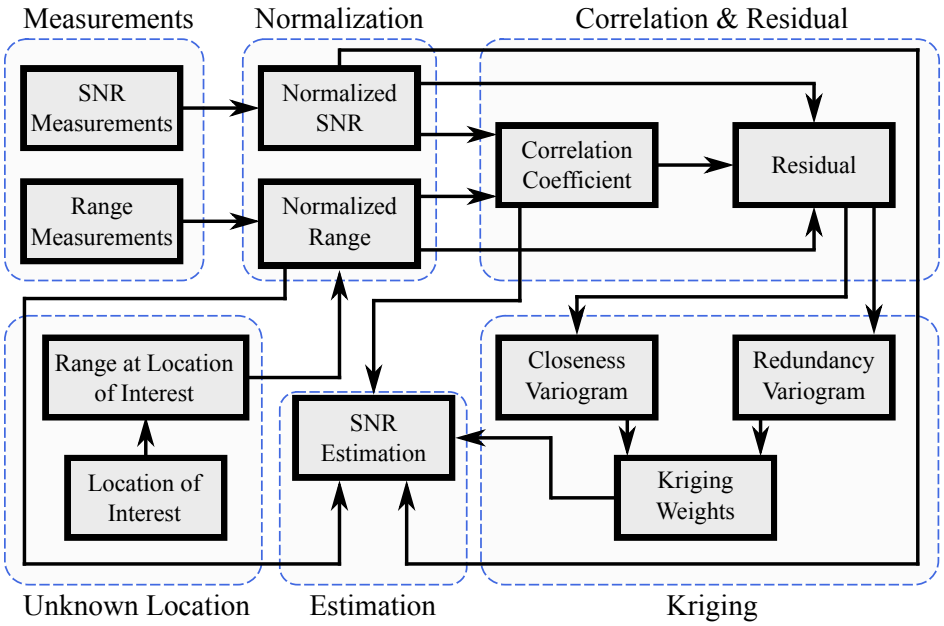


Figure 3.3: The structure of the communication performance estimator with multicollocated ordinary cokriging. The sequence operates clockwise, starting from the measurements. The structure incorporates six stages: 1) collection of measurements, 2) normalization of measurements, 3) computation of the correlation coefficient and the orthogonal residual, 4) ordinary kriging of the residual, and 5) the unknown location to 6) estimate the communication performance.

where we assume there are $j = 1, \ldots, N_j$ measurements. Primary measurements correspond to $\delta = 1$, secondary measurements correspond to $\delta = 2$, and $\mu_\delta = (1/N_j) \sum_{j=1}^{N_j} Z_{\delta,j}$ is the mean of the corresponding δ variable. This normalization results in a zero mean $\tilde{\mu}_\delta = 0$ and a variance $\mathrm{Var}[\tilde{Z}_\delta] = 1$ for both primary and secondary variable measurements. Thus, the slope of the linear regression in (3.29) matches the correlation coefficient, $p = \rho_{12}(0)$. Next, we compute the correlation coefficient $\rho_{12}(0)$ and the residual R as in (3.30). Then, we perform ordinary kriging to the residual to obtain the residual weights $\boldsymbol{\beta}_{\mathrm{R}}$ as in (3.19). An important aspect of kriging is the variogram which in our case is modeled as a spherical function,

$$\gamma(h) = \begin{cases} C_1(0)\left(\frac{3}{2}\frac{h}{\alpha} - \frac{1}{2}\left(\frac{h}{\alpha}\right)^3\right) & , \ h < \alpha \\ C_1(0) & , \ h \geq \alpha, \end{cases} \tag{3.37}$$

where α is the kriging range and h the distance of the measurements. The kriging range represents the maximum distance of correlation between measurements. Thus, beyond the kriging range the measurements are considered uncorrelated. Finally, we employ the orthogonal residual weights, the normalized SNR measurements, the normalized range measurements, and the correlation coefficient to estimate the SNR at the location of interest and its variance as in (3.34), (3.35) respectively.

Remark 3.7. The kriging range, the sill, and the nugget are user defined in our simulation environment, yet in practice should be experimentally identified. A robust methodology to fit variogram models with experimental data is discussed in [26].

Computational Complexity

We discussed that ordinary cokriging can be reduced to ordinary kriging of the orthogonal residual in a multicollocated setup. Thus, instead of $\mathcal{O}(N_j + N_l)^3$ computations for $\mathbf{\Gamma}^{-1}$ of ordinary cokriging (3.24), the proposed methodology requires $\mathcal{O}(N_j)^3$ computations of ordinary kriging (3.17), where usually $N_l > N_j$. Even though the multicollocated cokriging reduces the computational effort, it still remains intractable for online implementation with large number of measurements. To alleviate the online implementation, acceleration methods [114] may be used.

3.1.5 Simulations and Results

In this section, we provide simulations to compare the efficacy of the ordinary kriging to the proposed cokriging technique. We also present the communication performance between vehicles in a time-varying underwater environment.

Simulation Environment

The simulation environment captures the time-varying water conditions of the ambient noise with a 2D Gaussian. This is a common practice for the ambient noise, yet the mean of the Gaussian should not be zero [116]. Thus, the mean follows $\mu_{\text{amb}}(\mathbf{x}) = 0.3 + 1.2\text{e}^{-\|\mathbf{x}-[0.5 \;\; 1]^\intercal\|^2} + \text{e}^{-\|\mathbf{x}-[1.5 \;\; 1.5]^\intercal\|^2}$. We evaluated the mean over a grid of points in the space $\mathcal{S} \coloneqq \mathbb{X} \times \mathbb{Y}$, where $\mathbb{X} = \{-2, -1.95, \ldots, 4\}$ and $\mathbb{Y} = \{-2, -1.95, \ldots, 3.95, 4\}$. The spatial environmental conditions as well as the global path of the vehicles are shown in Fig. 3.4. Based on the Wenz curves [132], typical ambient noise ranges $\text{NL}_{\text{amb}} \in [25, 45]$ dB, for signal frequency $f = 25$ kHz. The resulting mean for the space of interest outputs values $\mu_{\text{amb}}(\mathbf{x}) \in [0.50, 2.12]$. Thus, we assign ambient noise values to every cell, following a linear relation. For example, a cell with mean value $\mu_{\text{amb}}(\mathbf{x}) = 1.00$ results in ambient noise level,

$$\begin{aligned}\text{NL}_{\text{amb}}(\mathbf{x}) &= \text{NL}_{\text{amb}}^{\max} - \text{NL}_{\text{amb}}^{\min}\Big(\frac{\mu_{\text{amb}}(\mathbf{x}) - \mu_{\text{amb}}^{\min}}{\mu_{\text{amb}}^{\max} - \mu_{\text{amb}}^{\min}}\Big) \\ &= 45 - 25\Big(\frac{1.00 - 0.50}{2.12 - 0.50}\Big) = 37.28\,\text{dB}.\end{aligned}$$

The Wenz curves indicate ambient noise $\text{NL}_{\text{amb}} = 25$ dB for wind speed of less then 1 knot and $\text{NL}_{\text{amb}} = 45$ dB for wind speed of 28 to 33 knots. Therefore, the environment shown in Subfig. 3.4(a) is an extreme environment with high variations in wind speed that corrupt the SNR. The source level is chosen to be $\text{SL} = 181$ dB.

For the simulated measurements we need to evaluate the communication performance in the intermediate locations of the two vehicles. Thus, we introduce the evaluation path which is the straight line that connects the transmitting vehicle and the receiving vehicle. Next, we search for grid cells which accommodate the evaluation path and compute the average mean to assign an SNR value. Let the accommodating grid cells of the evaluation path to be $\mathcal{S}_{\text{x}} = \{\mu_{\text{amb}}(\mathbf{x}_1), \ldots, \mu_{\text{amb}}(\mathbf{x}_M)\} \subset \mathcal{S}$. Then, the resulting ambient noise is computed as

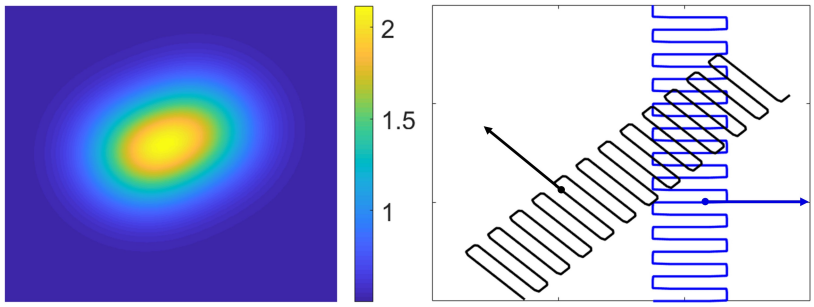


Figure 3.4: The environmental conditions and the global path of the vehicles. (a) The spatial environmental conditions are modeled with a 2D Gaussian where higher mean values represent more corrupted SNR with noise. (b) The path of the first vehicle is shown with a black solid line and of the second vehicle with a blue solid line.

$\mathrm{NL}_{\mathrm{amb}}(\mathbf{x}) = (1/M)\sum_{m=1}^{M}\mu_{\mathrm{amb}}(\mathbf{x}_m)$. To this end, we not only consider the environmental conditions at the location of the transmitting $\mathbf{x}_{\mathrm{t}}$ and the receiving vehicle $\mathbf{x}_{\mathrm{r}}$, but also we acknowledge the environmental conditions of the path that the SNR propagates.

Communication Performance Estimation

We perform two sets of simulations focusing on the estimation of the communication performance with and without partial information of the environment with high ambient noise. We assume that the vehicles can acquire range measurements during all communication events.

In Fig. 3.5, we present the first set of simulations comprising of two scenarios with two vehicles following different paths. In the upper row of Fig. 3.5 the x–marks (black for vehicle 1 and red for vehicle 2) represent the 150 locations of measurements and the squares (gray for vehicle 1 and magenta for vehicle 2) the 283 unknown locations of interest. Note that in both cases we did not collect measurements from the area with increased ambient noise (depicted in the background with yellow). For the simulation shown in Fig. 3.5(a) we seek to assess communication performance when in the presence of ambient noise. That is significantly different from the measured communication performance, i.e. without any

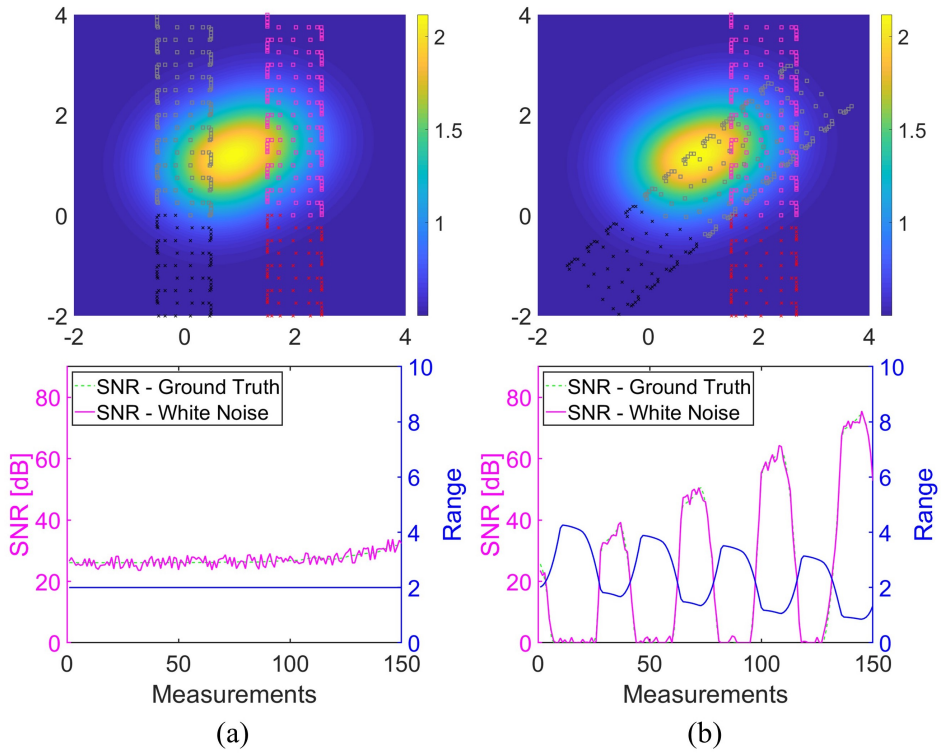


Figure 3.5: The first set of simulations with the vehicles paths and their corresponding measurements. (a) The vehicles follow similar zig-zag paths at the same direction and they collect 150 measurements right before the high-varying environment. (b) The vehicles follow opposite zig-zag paths at the same direction and they collect 150 measurements.

knowledge of the high variability of the environment. The corresponding SNR and range measurements are provided in the bottom row of Fig. 3.5. In Fig. 3.5(a), the vehicles follow similar zig-zag paths, and they are always facing in direction. As a result, the measurements are almost identical at all locations. The correlation coefficient of the normalized SNR and range measurements yields $\rho_{12}(0) = p = -0.098$. In Fig. 3.5(b), the vehicles follow opposite zig-zag paths at different directions and the correlation coefficient is computed $\rho_{12}(0) = p = -0.993$. Therefore, not only the measurements are highly varying, but also produce different amplitude. Since, in both cases the measurements were collected at a similar environment, the communication performance measurements are only affected by variations in range.

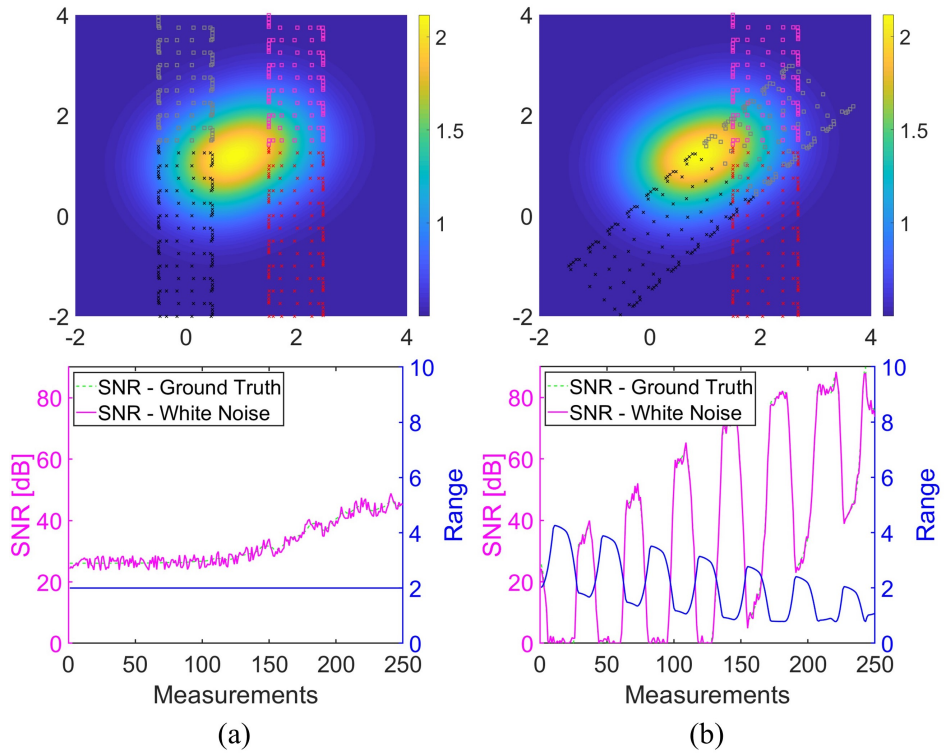


Figure 3.6: The second set of simulations with the vehicles paths and their corresponding measurements. (a) The vehicles follow similar zig-zag paths at the same direction and they collect 250 measurements including half of the the high-varying environment. (b) The vehicles follow opposite zig-zag paths at the same direction and they collect 250 measurements.

In Fig. 3.8, we show the absolute error of the SNR estimation with the ground truth of the ordinary kriging (OK) in red, and the multicollocated ordinary cokriging (MCOK) in blue. The shaded areas represent the variation of the estimation and the dashed lines the mean of the absolute error. In the first case, OK and MCOK have identical estimation outcomes, yet for large indices which corresponding to being far from locations where measurements were acquired, the MCOK provides more reliable estimates. Also, the MCOK mean is slightly lower, 4.35% from the OK mean. The higher error values of both techniques from the first estimate to approximately the 160-th estimate indicates the high ambient noise in the center of the environment. In the second case, the OK estimates are equally accurate at points of interest very close to the last measurements, yet the error increases much faster for the

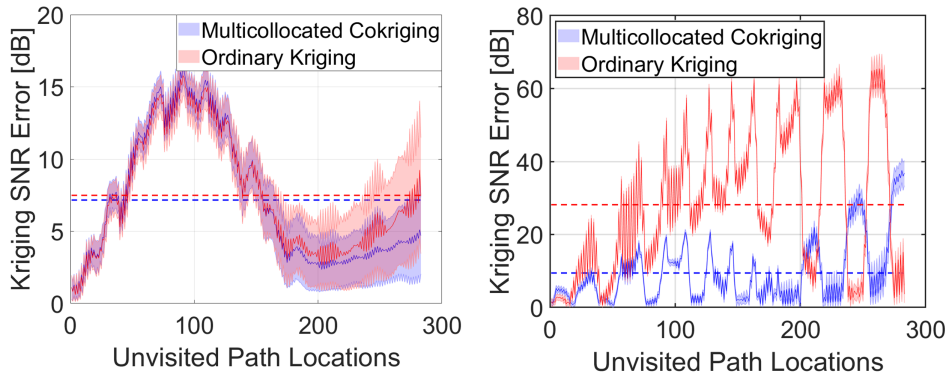


Figure 3.7: The absolute error values with their variance for the first set of simulations. The mean of the average error of the multicollocated cokriging and the ordinary kriging are illustrated in blue and red dashed lines respectively.

OK estimates at distant locations of interest to the acquired measurements. Thus, MCOK outperforms in long-term estimates and its mean is significantly lower, 66.47% from the OK mean.

The second set of simulations is shown in Fig. 3.6. We consider two cases following identical paths with the previous set of simulations, but with more measurements to cover half of the high ambient noise area, appearing in the center of the environment. Our objective is to provide more measurements to both methodologies with information on the high ambient noise area of the environment. More specifically, we gather 250 measurements as illustrated in the upper row of Fig. 3.6 with black and red x-marks corresponding to vehicle 1 and vehicle 2 respectively. The unknown locations of interests are represented by gray and magenta squares corresponding to vehicle 1 and vehicle 2. The correlation coefficients result in $\rho_{12}(0) = p = -0.064$ and $\rho_{12}(0) = p = -0.957$ for the first and the second case respectively. Surprisingly, the second set of simulations provides insufficient results for both techniques with radially unbounded errors, even with more measurements, as presented in Fig. 3.8. In Fig. 3.8(a) the OK and MCOK provide sufficient estimates for locations of interest close to the last measurements, yet for distant locations of interest the estimation error is unsatisfactory.

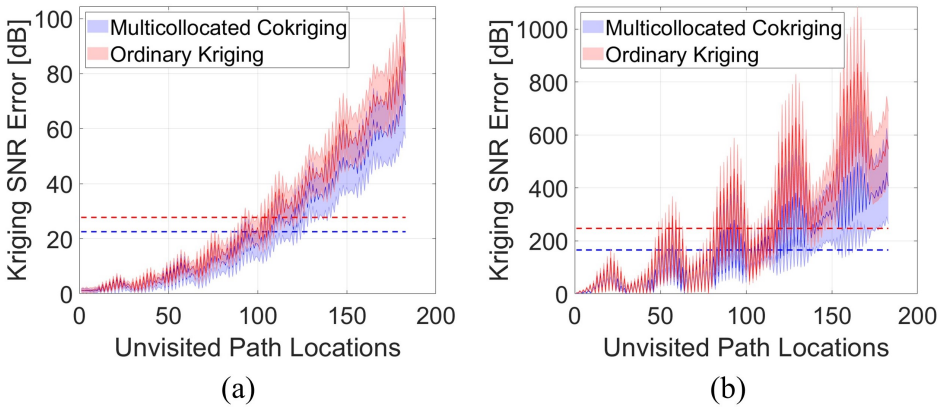


Figure 3.8: The absolute error values with their variance for the second set of simulations. Both approaches provide poor performance, yet the multicollocated cokriging outperforms the ordinary kriging estimates.

The MCOK produces lower mean error, 18.71% from the OK mean. In Fig. 3.8(b) both techniques show poor performance with high error measurements. Although, both techniques have unsatisfactory performance, the MCOK produces significantly lower mean in the order of 32.92% from the OK mean. The high absolute errors appear because ordinary kriging assumes constant means which consequently lead to locally biased kriging estimates.

In all cases the MCOK produces lower mean errors, revealing that the effect of the range is crucial to obtain better estimation results. Although in long-term estimates the MCOK provides more accurate results, in very close proximity to the measurements the OK provides similar results. In the second set of simulations both techniques demonstrate poor performance. This is occurred due to the nature of ordinary kriging that assumes a stationary constant mean, as discussed in (3.14). In practice, the spatial global mean is a conservative assumption, as usually the mean follows a *trend* over the spatial domain. An alternative kriging method with a non-stationary mean is the universal kriging, that considers basis functions to capture the underlying trend in the mean value.

3.1.6 Conclusion

Our work illustrates deficiencies in kriging for generating communication performance estimates, arising mainly from the structure of the assumptions. Moreover, our work shows that using range as a secondary variable in a cokriging formulation of the problem, yields lower absolute errors and performs better in long-term estimates. More specifically, we compare the proposed methodology with ordinary kriging and we show that the proposed framework provides better communication performance estimates with lower absolute errors in all simulation scenarios. Only in very short-term estimates and in certain cases the ordinary kriging computes similar absolute errors. However, at distant locations of interest from the acquired measurements the proposed methodology provides better results. The simulations reveal that for realistic applications the assumption of stationary global mean of both techniques is rather conservative and develops unacceptable absolute errors.

In the following section, we consider a realistic underwater acoustic propagation model to design basis functions for the mean estimation. In addition, a rigorous estimation technique for the covariance matrix is presented.

3.2 Learning of Communication Performance

3.2.1 Problem Formulation

In this section we discuss the foundations of random fields, describe the problem, and present the UWA communication performance model. In addition, we formulate the problem as a Gaussian random field.

Foundations

The notation here is standard. The set of real numbers is denoted $\mathbb{R}$, the set of all positive real numbers $\mathbb{R}_{>0}$, and the set of all non-negative real numbers $\mathbb{R}_{\geq 0}$. The transpose and inverse operators are denoted $(\cdot)^\intercal$ and $(\cdot)^{-1}$ respectively. The expectation, the variance and the covariance operators are represented by $\mathrm{E}[\cdot]$, $\mathrm{Var}[\cdot]$, and $\mathrm{Cov}(\cdot,\cdot)$ respectively. The notation $\mathbf{y} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ denotes $\mathbf{y}$ that is drawn from a Gaussian distribution with a vector of means $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$. We denote by I_n the identity matrix of $n \times n$ dimension. The vector of n zeros is represented as $\mathbf{0}_n$ and the matrix of $n \times m$ zeros as $\mathbf{0}_{n\times m}$. The hat $\hat{y}$ denotes the estimated value of y and the superscript in parenthesis $\hat{y}^{(n)}$ the n-th iteration of an estimation process. The cardinality of the set K is denoted $\mathrm{card}(K)$, the absolute values is denoted $|\cdot|$, and $\|\cdot\|$ denotes the L_2 norm.

Next, we introduce basic notions of random fields. For a more in-depth discussion the reader may refer in [1, 27, 85]. A *random field* is a stochastic process indexed in the Euclidean space. Let $Z(\mathbf{x})$ be a *random field* [1] with covariance function $\mathrm{Cov}(Z(\mathbf{x}), Z(\mathbf{x}+\mathbf{h}))$ for all $\mathbf{x}$, $\mathbf{x}+\mathbf{h} \in \mathbb{R}^m$, where $\mathbf{x}$ denotes the spatial coordinates and $\mathbf{h}$ is the separation vector between two locations, and m is the dimension of the coordinates, e.g. $m = 2$ for planar coordinates. The *variogram* is a statistical measure of spatial autocorrelation that is defined by,

$$2\gamma(\mathbf{h}) \coloneqq \mathrm{E}\left[\Big(Z(\mathbf{x}+\mathbf{h}) - Z(\mathbf{x})\Big)^2\right], \tag{3.38}$$

where $\gamma(\mathbf{h}) : \mathbb{R}^m \to \mathbb{R}_{\geq 0}$ is a conditionally negative definite function [129] termed as *semi-variogram*. The condition ensures that the variance of the random field $Z(\mathbf{x})$ is positive.

Lemma 3.8. *A semivariogram function $\gamma : \mathbb{R}^m \to \mathbb{R}$ is a conditionally negative definite*

[1] Throughout the dissertation, we use the "random field," "random process," and "random function" interchangeably.

function if and only if $\exp\{-\zeta\gamma\}$ *is positive definite for all* $\zeta > 0$.

Proof. The proof follows from [9, page 74]. □

A random field is *intrinsically stationary* if both $\mathrm{E}[Z(\mathbf{x}+\mathbf{h}) - Z(\mathbf{x})] = 0$ and $\mathrm{Var}[Z(\mathbf{x}+\mathbf{h}) - Z(\mathbf{x})] = 2\gamma(\mathbf{h})$ for all $\mathbf{x},\ \mathbf{x}+\mathbf{h} \in \mathbf{R}^m$ are satisfied. An intrinsically stationary random field with constant mean $\mathrm{E}[Z(\mathbf{x})] = \mu$ and $\mathrm{Cov}[Z(\mathbf{x}), Z(\mathbf{x}+\mathbf{h})] = C(\mathbf{h})$ is called *second-order stationary*. Note that the covariance function $C(\cdot)$ is a conditionally positive definite function and *stationary*—depending only on the separation vector $\mathbf{h}$ and not on spatial coordinates $\mathbf{x}$. Second-order stationarity implies intrinsically stationarity and the Gaussian assumption, yet the converse is not always true.

For a second-order stationary random field the correlation function is defined by $\rho(\mathbf{h}) \coloneqq C(\mathbf{h})/C(\mathbf{0})$, where $\rho(\mathbf{h}) \in [-1, 1]$ with $|C(\mathbf{h})| \leq C(\mathbf{0}) = \mathrm{Var}[Z(\mathbf{x})]$ and $C(\mathbf{0}) = \sigma^2 + \tau^2$ is the *sill* of the semivariogram with σ^2 the *partial sill* and τ^2 the *nugget effect*. The partial sill σ^2 is a semivariogram value where no correlation of data further exists and the nugget τ^2 represents the variance of the data measurement error at a given location.

Given a covariance function $C(\mathbf{h})$ the variogram (3.38) yields,

$$\begin{aligned} 2\gamma(\mathbf{h}) &= \mathrm{Var}[Z(\mathbf{x}+\mathbf{h}) - Z(\mathbf{x})] \\ &= \mathrm{Var}[Z(\mathbf{x}+\mathbf{h})] + \mathrm{Var}[Z(\mathbf{x})] - 2\mathrm{Cov}[Z(\mathbf{x}+\mathbf{h}), Z(\mathbf{x})] \\ &= C(\mathbf{0}_m) + C(\mathbf{0}_m) - 2C(\mathbf{h}) \\ &= 2(C(\mathbf{0}_m) - C(\mathbf{h})). \end{aligned} \tag{3.39}$$

We cannot always construct the covariance from the variogram, as the variogram may be unbounded. Thus, let us assume that the random field is *ergodic*. That is as $\|\mathbf{h}\| \to \infty$ then $C(\mathbf{h}) \to 0$. In other words, when the distance between two measurements is very large

$\|\mathbf{h}\| \to \infty$, there is no spatial correlation $C(\mathbf{h}) \to 0$. The limit of (3.39) as $\|\mathbf{h}\| \to \infty$ yields,

$$C(\mathbf{h}) = \gamma(\infty) - \gamma(\mathbf{h}), \tag{3.40}$$

where $\gamma(\infty) = \sup_{\mathbf{h}} \gamma(\mathbf{h}) < \infty$ is non-negative.

When the variogram depends only on the displacement vector norm, i.e. $2\gamma(\mathbf{h}) = 2\gamma(\|\mathbf{h}\|)$, then the variogram is *isotropic*, otherwise it is *anisotropic*. The ensuing discussion assumes second-order stationarity and isotropic variogram after detrending.

Problem Formulation

We consider the problem of inter-vehicle UWA communication of two vehicles. In Fig. 3.1, we illustrate two cases of UWA communication between two vehicles at range r, with $\mathbf{x}_{\mathrm{t}}$ the position of the *transmitting* vehicle and $\mathbf{x}_{\mathrm{r}}$ the position of the *receiving* vehicle. The first case is shown in Fig. 3.1-(a) where the success of the communication event depends solely on a maximum communication range Q. This means that if the vehicle range exceeds the communication range $r > Q$, then the communication cannot be accomplished. In practice, this binary approach is unrealistic, as multiple spatially-dependent factors may affect the communication of two vehicles, such as scattering, motion-induced Doppler effect, background noise and change of environmental conditions. To this end, we propose multi-dimensional communication performance maps for various ranges as illustrated in Fig. 3.1-(b). More specifically, we assess the communication performance of an UWA network of vehicles for specific ranges by modeling the problem as a spatial Gaussian random field with a spatially varying mean. Note that the Gaussian model is a reasonable assumption, as it has been validated with multiple experimental data [100]. For the evaluation of the communication performance we employ signal-to-noise-ratio (SNR) measurements.

Let the SNR measurements be modeled by,

$$Y(\mathbf{x}; v) = \boldsymbol{\mu}(\mathbf{x}; v) + Z(\mathbf{x}; v) + \epsilon(\mathbf{x}), \tag{3.41}$$

where $Y(\mathbf{x}; v) \in \mathbb{R}^n$ is the measurement vector describing a non-stationary random field at spatial coordinates $\mathbf{x} \in \mathbb{R}^2$, $\boldsymbol{\mu}(\mathbf{x}; v)$ is the deterministic mean that represents the large-scale variation, and $\epsilon \sim \mathcal{N}(0, \tau^2 I_n)$ is an independent and identical distributed (iid) zero-mean Gaussian random field capturing micro-scale variation of the sensor. The mean $\boldsymbol{\mu}$ is the spatial trend that represents large-scale variability, the second-order stationary random field Z captures medium-scale variability, and the white noise ϵ is the small-scale variation of the sensor. The surrogate variable is denoted v and is used to represent model dependence, not explicitly accounted for spatial coordinates $\mathbf{x}$. In the Section 3.2.1, we identify the surrogate variable by using an UWA propagation channel model.

Assumption 3.9. *The deterministic mean is decomposed by a linear combination of unknown parameters expressed by* $\boldsymbol{\mu}(\mathbf{x}; v) = \mathbf{X}(\mathbf{x}; v)\boldsymbol{\beta}$*, where* $\mathbf{X}(\mathbf{x}; v) \in \mathbb{R}^{n \times p}$ *represents the matrix of known basis functions and* $\boldsymbol{\beta} \in \mathbb{R}^p$ *the vector of the unknown regressor coefficients.*

Since the measurements Y are non-stationary, we detrend the measurements, i.e. remove the mean $Y - \boldsymbol{\mu}$, to obtain a stationary random field. Next, with the detrended measurements the covariance matrix $\boldsymbol{\Sigma}$ is estimated with an iterative scheme. After estimating the covariance matrix $\boldsymbol{\Sigma}$, we employ the original measurements Y to perform predictions. A critical component for detrending is the basis functions $\mathbf{X}$, thus we are inspired by the propagation model to design $\mathbf{X}$ and accurately detrend the measurements.

Remark 3.10. The major difference between kriging and Gaussian processes (GPs) is that the former computes the covariance function C through the semivariogram function γ (3.40). In a second-order spatial random field, this intermediate step provides better estimates for

three reasons: i) estimation bias [38, pp. 313-320]; ii) boundedness properties [131, pp. 79–84]; and iii) trend contamination [27, pp. 70–73]. Since this paper regards a second-order spatial random field Z with trend $\boldsymbol{\mu}$, we find kriging more suitable over GPs.

Communication Performance

For communication performance, we use an UWA propagation channel model and its statistical characterization, described in [100, 115, 116]. The statistical model comprises the physical model of the UWA communication channel and random vehicle perturbations which affect the local SNR. Large-scale variability of SNR occurs due to large-scale spatial variations in environmental conditions, evoking local error variations and thus a non-stationary random field.

To approximate the communication performance between two agents we use the SNR. In principle, the higher the SNR, the more likely is to detect the signal. In this work we consider fixed signal power, frequency f, and bandwidth B. Let the power of the transmitted signal be constant, then the SNR yields,

$$SNR = \frac{P_{\mathrm{T}} G}{P_{\mathrm{N}}}, \tag{3.42}$$

where P_{T} denotes the power of the transmitted signal, G is the channel gain and P_{N} is the power of noise. The gain G has been shown to follow a log-normal distribution $\log G \sim \mathcal{N}(\bar{G}, \sigma_{\mathrm{G}}^2)$, where $\bar{G}$ represents the mean of the log channel gain and σ_{G}^2 its variance [18, 100]. On the decibel scale, the source level takes the form of $S_{\mathrm{l}}(f) = 10 \log P_{\mathrm{T}}$ and the noise level yields $NL(f, \omega) = 10 \log P_{\mathrm{N}}$ [116]. If we neglect variations of water pressure with depth, then the gain on the decibel scale $g = 10 \log G$ is a Gaussian distribution, expressed as,

$$g(r) = \bar{g}(r) + \nu, \tag{3.43}$$

where $\nu \sim \mathcal{N}(0, \sigma_\nu^2)$ a zero-mean Gaussian random field. The mean follows,

$$\bar{g}(r) = g_0 - k_0 10 \log \frac{r}{r_{\text{ref}}}, \tag{3.44}$$

where g_0 is a constant gain, r_{ref} is reference range (e.g., 1 m in our case), and k_0 is the path loss exponent, provided by taking ensemble averages [99]. Ensemble averages is a method to represent the expected value of a waveform.

Note that (3.41) has identical structure with the model of the UWA propagation channel model (3.43). Thus, using (3.44) we choose v to be the range between transmitting and receiving node, i.e. $v = r$, and the SNR measurements (3.41) are expressed,

$$Y(\mathbf{x}; r) = \mathbf{X}(\mathbf{x}; r)\boldsymbol{\beta} + Z(\mathbf{x}; r) + \epsilon(\mathbf{x}). \tag{3.45}$$

The specific goal of our UWA performance prediction application is summarized in Problem 1.

Problem 1. Predict the communication performance $\hat{Y}$ and the corresponding variance $\text{Var}[\hat{Y}]$ at unvisited locations $\mathbf{x}_0$, provided a set of communication performance measurements Y at locations $\mathbf{x}$ and the vehicle range r.

3.2.2 Training of Gaussian Random Field

In this section, we formulate basis functions $\mathbf{X}$ and use least squares on the training data $Y(\mathbf{x}; r)$ to estimate the unknown regressor coefficients $\boldsymbol{\beta}$ of the spatial trend $\boldsymbol{\mu}(\mathbf{x}; r)$. Then, we remove the trend by subtracting the mean $\boldsymbol{\mu}(\mathbf{x}; r)$ from the measurements $Y(\mathbf{x}; r)$ to retrieve a stationary random field. The detrended measurements $Y - \boldsymbol{\mu}$ are used to estimate the parameters of multiple variogram functions with a maximum likelihood-based method.

Next, we select the most suitable covariance model, based on the Bayesian information criterion. With the selected variogram model we construct the covariance matrix $\boldsymbol{\Sigma}$ and use generalized least squares to improve the accuracy of the spatial trend estimator $\boldsymbol{\mu}$. The method iterates until the parameters of the variogram function converge.

Spatial Trend Modeling

The random field in (3.45) is non-stationary due to the spatial trend. Thus, the original measurements cannot be used to estimate the parameters of the variogram. To this end, we seek basis functions $\mathbf{X}$ to model the spatial trend $\boldsymbol{\mu}$, detrend the measurements $Y - \boldsymbol{\mu}$, and recover stationarity.

A precise model of the trend is important for spatial extrapolation, ideally arising from the physics of the system [31]. The obvious choice for the elements of the basis function $\mathbf{X}$ is to employ spatial coordinates as covariates. In spatial statistics, polynomial basis functions of spatial coordinates, e.g., $\mathbf{X}(\mathbf{x}) = [1, x, y, xy, x^2, y^2]$, are often employed [27]. However, polynomial basis functions do not behave well for extrapolation, because they are radially unbounded, i.e. as $\|\mathbf{x}\| \to \infty$ then $X(\mathbf{x}) \to \infty$. To this end, Gaussian radial basis functions (RBF) are widely used in various applications [120], as they provide suitable extrapolation results. In addition, surrogate variables—arising from the physical model of the system—are useful covariates to interpret the behavior of the spatial variation [31]. A Gaussian RBF is described by,

$$X_l(\mathbf{x}; c_l, \sigma^2_{\mathrm{R},l}) = \exp\left(-\frac{(\mathbf{x} - c_l)^2}{2\sigma^2_{\mathrm{R},l}}\right), \tag{3.46}$$

where c_l is the center of each measurement, e.g., $c_l = 0$ for zero mean measurement error ϵ (3.45). The corresponding variance is denoted $\sigma^2_{\mathrm{R},l}$, where in practice is a constant value $\sigma^2_{\mathrm{R},l} = \sigma^2_{\mathrm{R}}$ for all l measurements. From (3.44), it is deduced that the range of the vehicles has

a linear-log relationship to the mean. Hence, our proposed hybrid basis function combines Gaussian RBF incorporating spatial coordinates (3.46) and linear-log range,

$$\mathbf{X}(\mathbf{x};r) = [1, \exp\left(-\frac{(x-c_x)^2}{2\sigma_x^2}\right), \exp\left(-\frac{(y-c_y)^2}{2\sigma_y^2}\right), r, \log r]. \tag{3.47}$$

For data detrending, since the covariance function is unknown, the generalized least squares (GLS) cannot be used. Thus, we initially estimate the unknown parameters using ordinary least squares (OLS),

$$\widehat{\boldsymbol{\beta}}_{\text{OLS}}^{(1)} = \mathbf{X}(\mathbf{x};r)^{\dagger} Y(\mathbf{x};r), \tag{3.48}$$

where $\mathbf{X}^{\dagger} = (\mathbf{X}^{\intercal}\mathbf{X})^{-1}\mathbf{X}^{\intercal}$, $\mathbf{X}^{\dagger} \in \mathbb{R}^{p\times n}$ is the Moore-Penroe pseudoinverse of $\mathbf{X}$. The estimated unknown parameters $\widehat{\boldsymbol{\beta}}_{\text{OLS}}^{(1)}$ are not the final estimated unknown regressor values. Instead, we shall employ $\widehat{\boldsymbol{\beta}}_{\text{OLS}}^{(1)}$ to detrend the measurements and assess their behavior with an iterative technique. The Gaussian residual random field (or detrended data) is expressed,

$$\tilde{Y}(\mathbf{x};r) = Y(\mathbf{x};r) - \mathbf{X}(\mathbf{x};r)\widehat{\boldsymbol{\beta}}_{\text{OLS}}^{(1)}. \tag{3.49}$$

Assumption 3.11. *The random field of the underlying latent process is second-order stationary after detrending, i.e. $\tilde{Y}$ is second-order stationary.*

Assumption 3.12. *The variogram function is isotropic after detrending.*

Experimental Semivariogram and Theoretical Models

In this section, we present three commonly used semivariograms and an optimization method to estimate the initial parameters of the semivariogram function. The Matheron empirical semivariogram [84] is used in the majority of the literature for the estimation of the unknown

parameters,

$$\hat{\gamma}(\mathbf{h}) = \frac{1}{2\,\mathrm{card}(N(\mathbf{h}))} \sum_{N(\mathbf{h})} |\tilde{Y}(\mathbf{x}+\mathbf{h}) - \tilde{Y}(\mathbf{h})|^2,$$

where $N(\mathbf{h}) = \{(o,p) \mid \mathbf{x}_o - \mathbf{x}_p = \mathbf{h}\}$ is the set of measurements at distance $\mathbf{h}$ and $\tilde{Y}$ is the vector of the residual measurements (3.49). The main idea is to compute the experimental semivariogram from the detrended data and then compare it to theoretical semivariogram models. The Matheron empirical semivariogram is unbiased, yet it is highly affected by outliers, due to the squared term. A robust estimator of the experimental semivariogram is proposed in [28] as,

$$\hat{\gamma}_{\mathrm{CH}}(\mathbf{h}) = \frac{\left(\frac{\sum_{N(\mathbf{h})} |\tilde{Y}(\mathbf{x}+\mathbf{h}) - \tilde{Y}(\mathbf{h})|^{1/2}}{\mathrm{card}(N(\mathbf{h}))}\right)^4}{0.914 + \frac{0.988}{2\,\mathrm{card}(N(\mathbf{h}))} + \frac{0.090}{\mathrm{card}(N(\mathbf{h}))^2}}. \tag{3.50}$$

The robustness relies on a transformation which ensures that the fourth root of the transformed distribution produces relatively small skew. Note that we cannot interpolate the experimental semivariogram to obtain a semivariogram, because the conditional negative definitiveness property may be violated. Instead, we fit the experimental semivariogram to theoretical models that ensure the desired properties of a semivariogram function.

We consider three potential theoretical semivariogram models which are conditional negative definite. The spherical semivariogram is given by,

$$\gamma_{\mathrm{s}}(\mathbf{h};\boldsymbol{\theta}) = \begin{cases} \tau^2 + \sigma^2, & \|\mathbf{h}\| \geq \alpha, \\ \tau^2 + \sigma^2 \left(\frac{3\|\mathbf{h}\|}{2\alpha} - \frac{1}{2}\left(\frac{\|\mathbf{h}\|}{\alpha}\right)^3 \right), & \|\mathbf{h}\| \leq \alpha, \end{cases} \tag{3.51}$$

where the semivariogram parameter vector $\boldsymbol{\theta} = [\tau^2\ \sigma^2\ \alpha]^\intercal \in \Theta$ contains the nugget, the partial sill, and the semivariogram range with $\Theta = \{\boldsymbol{\theta} \in \mathbb{R}^3 \mid \tau^2 \geq 0, \sigma^2 \geq 0, \alpha \geq 0\}$ the parameter space. Note that the semivariogram and the covariance parameters are identical.

We also consider the exponential semivariogram function,

$$\gamma_{\mathrm{e}}(\mathbf{h};\boldsymbol{\theta}) = \tau^2 + \sigma^2\Big(1 - \exp\Big\{-\frac{\|\mathbf{h}\|}{\alpha}\Big\}\Big). \tag{3.52}$$

Finally, the Matérn semivariogram function [83],

$$\gamma_{\mathrm{m}}(\mathbf{h};\boldsymbol{\theta}) = \tau^2 + \sigma^2\left(1 - \frac{(\|\mathbf{h}\|/\alpha)^{\kappa}}{2^{\kappa-1}\Gamma(\kappa)} K_{\kappa}\Big(\frac{\|\mathbf{h}\|}{\alpha}\Big)\right),$$

where $\Gamma(\cdot)$ is the gamma function, K_κ is the Bessel function of order κ, and κ is the smoothing parameter. Note that the Matérn semivariogram function is a general model, thus we fix the smoothing parameter at $\kappa = 3/2$ to obtain a mixed polynomial-exponential form,

$$\gamma_{\mathrm{pe}}(\mathbf{h};\boldsymbol{\theta}) = \tau^2 + \sigma^2\Big(1 - \Big(1 + \frac{\sqrt{3}\|\mathbf{h}\|}{\alpha}\Big)\exp\Big\{-\frac{\sqrt{3}\|\mathbf{h}\|}{\alpha}\Big\}\Big). \tag{3.53}$$

We will employ all semivariogram functions $\mathcal{C} = \{\gamma_{\mathrm{s}}, \gamma_{\mathrm{e}}, \gamma_{\mathrm{pe}}\}$ and evaluate their performance.

The next step is to formulate an optimization problem to fit the models $\mathcal{C}$ and derive the corresponding parameter vector $\boldsymbol{\theta}$. We utilize a weighted least squares (WLS) approach [26] which yields,

$$\widehat{\boldsymbol{\theta}}^{(0)}_{\mathrm{CWLS}} = \arg\min_{\boldsymbol{\theta}\in\Theta} \sum_{g=1}^{N_g} \mathrm{card}(N(\mathbf{h}(g))) \left(\frac{\hat{\gamma}_{\mathrm{CH}}(\mathbf{h}(g))}{\gamma(\mathbf{h}(g);\boldsymbol{\theta})} - 1\right)^2, \tag{3.54}$$

where N_g is the total number of the separation vectors $\mathbf{h}_g$.

The parameter estimation (3.54) relies on the residual measurements $\tilde{Y}$ (3.49) which incorporates measurement bias. Thus, the estimation is sensitive to the bias of the mean value.

Unbiased Semivariogram Model Fitting

In this section, we seek an unbiased estimator for the parameter vector $\boldsymbol{\theta}$ and a strategy to narrow down the parameter space Θ. Maximum likelihood (ML) estimation is used widely in statistics. In spatial statistics, due to high correlation of the observations, ML is known to generate unfavorable outcomes [65]. In addition, when the observations are limited, then the bias of the ML estimation is significant. An alternative bias-free approach is the restricted maximum likelihood (REML) estimation [48, 144], which makes use of error contrasts to remove the mean dependence from the estimation of variance.

An alternative bias-free approach is the restricted maximum likelihood (REML) estimation [48, 144], which makes use of error contrasts to remove the mean dependence from the variance estimates. The main idea is to transform the residual measurements $\tilde{Y}$ from (3.49) with a matrix $\mathbf{A} \in \mathbb{R}^{n\times(n-p)}$ such that, $\mathbf{A}^\intercal\mathbf{X} = \mathbf{0}$ and $\mathrm{E}[\mathbf{A}^\intercal\tilde{Y}] = 0$, where $\mathbf{X}$ is the basis function (3.47). In other words, each column vector of matrix $\mathbf{A} = [\mathbf{a}_1\ \mathbf{a}_2\ \ldots\ \mathbf{a}_{(n-p)}]$ is orthogonal to all columns of $\mathbf{X}$. Let us define the error contrast, $W \coloneqq \mathbf{A}^\intercal\tilde{Y}$ to obtain $W \sim \mathcal{N}(0, \mathbf{A}^\intercal\boldsymbol{\Sigma}(\boldsymbol{\theta})\mathbf{A})$, which obviously does not depend on the estimated mean parameters $\widehat{\boldsymbol{\beta}}_{\mathrm{OLS}}$. Although $\mathbf{A}$ is not unique, a matrix that satisfies the properties is the orthogonal projection onto the kernel of $\mathbf{X}$, that is, $\mathbf{A} = I_n - \mathbf{X}(\mathbf{X}^\intercal\mathbf{X})^{-1}\mathbf{X}^\intercal$. We note that $\mathbf{A}$ does not depend on the estimated mean parameters $\widehat{\boldsymbol{\beta}}_{\mathrm{OLS}}$. Therefore, the log-restricted likelihood function is defined,

$$L(\boldsymbol{\theta}|W) = -\frac{1}{2}\Big((n-p)\log(2\pi) + \log|\mathbf{X}^\intercal\mathbf{X}| - \log|\boldsymbol{\Sigma}(\boldsymbol{\theta})| - \log|\mathbf{X}^\intercal\boldsymbol{\Sigma}(\boldsymbol{\theta})\mathbf{X}| - \tilde{Y}^\intercal\boldsymbol{\Pi}(\boldsymbol{\theta})\tilde{Y}\Big), \tag{3.55}$$

where $\boldsymbol{\Pi}(\boldsymbol{\theta}) = \boldsymbol{\Sigma}(\boldsymbol{\theta})^{-1} - \boldsymbol{\Sigma}(\boldsymbol{\theta})^{-1}\mathbf{X}(\mathbf{X}^\intercal\boldsymbol{\Sigma}(\boldsymbol{\theta})^{-1}\mathbf{X})^{-1}\mathbf{X}^\intercal\boldsymbol{\Sigma}(\boldsymbol{\theta})^{-1}$, n is the measurement vector size, and p is the rank of $\mathbf{X}$. Next, the log-restricted likelihood (3.55) is maximized with

respect to $\boldsymbol{\theta} \in \Theta$ to obtain the estimated parameter vector $\widehat{\boldsymbol{\theta}}$. To reduce the search of the parameter space Θ, we use the parameter estimate $\widehat{\boldsymbol{\theta}}^{(0)}_{\text{CWLS}}$ (3.54) as a center value of the initial set of parameters in the optimization scheme. So far we computed three covariance parameter vectors $\widehat{\boldsymbol{\theta}}$ corresponding to three candidate models (3.51), (3.52), (3.53). A benefit of likelihood-based approaches is that they can be combined with statistical model selection tools [88].

Statistical Model Selection

The Bayesian information criterion (BIC) is a statistical model selection methodology, introduced by Schwarz in [109]. The BIC is given by,

$$\text{BIC}(M_k) = -2 \ln \mathcal{L}(\widehat{\boldsymbol{\theta}}_k \mid \tilde{Y}, M_k) + p_k \ln n, \tag{3.56}$$

where $\mathcal{M} = \{M_k = \boldsymbol{\Sigma}(\widehat{\boldsymbol{\theta}}_k) \mid k = 1, \ldots, K\}$ is the set of candidate models, $\widehat{\boldsymbol{\theta}}_k$ denotes the REML estimates of $\boldsymbol{\theta}_k$, $p_k = 3$ is the dimension of the parameter space Θ, $\mathcal{L}(\widehat{\boldsymbol{\theta}}_k \mid \tilde{Y}, M_k)$ represents the marginal likelihood corresponding to the density function $p(\tilde{Y}, M_k \mid \widehat{\boldsymbol{\theta}}_k)$, and n is the measurement size of the vector $\tilde{Y}$. In our case $K = 3$ corresponds to three different candidate semivariogram functions (3.51), (3.52), and (3.53). In principle, the semivariogram function with the smallest BIC represents the true model, assuming that the real model is listed among the candidate covariance models. One of the major advantages of the BIC is that it satisfies the property of *consistency*. That is even if the true model is not listed among the candidate models, the BIC selects the most parsimonious model closest to the true model, by computing the marginal likelihood with Laplace approximation.

Since the BIC (3.56) is computed in the log-scale, its evaluation may be ambiguous. Thus,

we employ the posterior probability of the BIC [89] which is approximated by,

$$P(M_k \mid \tilde{Y}) \approx \frac{\exp\left(-\frac{1}{2}\Delta_k\right)}{\sum_{k=1}^{K}\exp\left(-\frac{1}{2}\Delta_k\right)}, \tag{3.57}$$

where $\Delta_k = \text{BIC}(M_k) - \text{BIC}^\star$ denotes the BIC difference of a candidate model with the minimum BIC candidate model $\text{BIC}^\star = \min_{M_k \in \mathcal{M}} \text{BIC}(M_k)$. Essentially, $P(M_k \mid \tilde{Y})$ is a probability mass function, that provides a probability of suitability for each model to the real model.

Nested Semivariogram Model

So far we assumed that the variation of the underlying process is purely represented by either a spherical (3.51), or an exponential (3.52), or a polynomial-exponential (3.53) variogram model. However, in many cases, the spatial variability is more complex, and thus a combination of semivariogram models interprets the latent process more precisely. The nested [131] (or compositional [32]) semivariogram function is defined by,

$$\gamma_{\text{nest}}(\mathbf{h}; \widehat{\boldsymbol{\theta}}_{\text{s},k}, \widehat{\boldsymbol{\theta}}_{\text{e},k}, \widehat{\boldsymbol{\theta}}_{\text{pe},k}) := \xi_1\gamma_{\text{s}}(\mathbf{h}; \widehat{\boldsymbol{\theta}}_{\text{s},k}) + \xi_2\gamma_{\text{e}}(\mathbf{h}; \widehat{\boldsymbol{\theta}}_{\text{e},k}) + \xi_3\gamma_{\text{pe}}(\mathbf{h}; \widehat{\boldsymbol{\theta}}_{\text{pe},k}), \tag{3.58}$$

where $\xi_k \in (0,1)$, and $\sum_{k=1}^{K}\xi_k = 1$.

Lemma 3.13. *Any convex combination of semivariograms is a semivariogram.*

Proof. Let γ_k be a semivariogram and $\boldsymbol{\gamma}_{-k} = \{\gamma_l\}_{l\neq k}$ a vector of semivariograms other than γ_k. Since $\gamma : \mathbb{R}^2 \to \mathbb{R}_{\geq 0}$ and $\xi_k \in (0,1)$, then $\gamma_{\text{nest}} = \sum_{k=1}^{K}\xi_k\gamma_k > 0$ for $\|\mathbf{h}\| \neq 0$. Moreover, $\exp\{-\zeta\gamma_{\text{nest}}\}$ is positive definite for all $\zeta > 0$. Hence, from Proposition 3.8 any convex combination of variograms γ_{nest} is a variogram. □

The nested semivariogram is similar in spirit to [32], yet the authors used directly the BIC. Since the BIC is in the log-scale (3.56), it does not scale well with the nested semivariogram. Alternatively, we employ the posterior probabilities of BIC $\xi_k = P(M_k \mid \tilde{Y})$ that satisfy $\xi_k \in (0,1)$ and $\sum_{k=1}^{K} \xi_k = 1$.

Iterative Parameter Training

For the iterative parameter training we utilize the estimated covariance matrix $\boldsymbol{\Sigma}(\widehat{\boldsymbol{\theta}}^{(1)})$. The covariance matrix allows the implementation of the generalized least squares (GLS) to improve the estimation of the mean. The GLS mean estimate is described by,

$$\widehat{\boldsymbol{\beta}}^{(2)}_{\text{GLS}} = \left(\mathbf{X}^{\intercal}\boldsymbol{\Sigma}\left(\widehat{\boldsymbol{\theta}}^{(1)}\right)^{-1}\mathbf{X}\right)^{-1}\mathbf{X}^{\intercal}\boldsymbol{\Sigma}\left(\widehat{\boldsymbol{\theta}}^{(1)}\right)^{-1}Y. \tag{3.59}$$

Sequentially, the residual random measurements (3.49) yield,

$$\tilde{Y}(\mathbf{x};r) = Y(\mathbf{x};r) - \mathbf{X}(\mathbf{x};r)\widehat{\boldsymbol{\beta}}^{(2)}_{\text{GLS}}. \tag{3.60}$$

In addition, the GLS mean estimation facilitates a more accurate determination of the covariance function. To this end, we employ the detrended measurements (3.60) and iterate the covariance training. The training is terminated when,

$$\|\widehat{\boldsymbol{\theta}}^{(s)} - \widehat{\boldsymbol{\theta}}^{(s-1)}\| \leq \eta \tag{3.61}$$

where $\eta \in \mathbb{R}_{>0}$ is a small error threshold. At every iteration we expect lower BIC values (3.56).Essentially, after the second iteration, the change on the mean and covariance estimate is insignificant [131, pp. 196–200], [49, 66], and usually the training is terminated.

3.2.3 Spatial Prediction

In this section, we describe universal kriging [27, 74, 129], a spatial prediction technique that predicts values at locations of interest, based on measurements from other locations Y and the estimated covariance matrix $\boldsymbol{\Sigma}$. The main difference from the ordinary kriging lies in the mean value of the random field, which is not assumed to be constant. More specifically, provided measurements Y at locations $\mathbf{x} \in \mathbb{R}^2$ the random field is described by (3.45). We use a linear unbiased estimator,

$$\hat{Y}(\mathbf{x}_0; r) = \sum_{i=1}^{n} \omega_i Y(\mathbf{x}_i; r) = \boldsymbol{\omega}^\intercal Y(\mathbf{x}; r), \tag{3.62}$$

where $\mathbf{x}_0 \in \mathbb{R}^2$ is the location of interest, $\boldsymbol{\omega} = [\omega_1 \ \dots \ \omega_n]^\intercal \in \mathbb{R}^n$ are the weights we seek to obtain, and $Y(\mathbf{x}; r)$ are the raw measurements, i.e. not the residuals. The unbiasedness of the predictor is ensured by $\mathrm{E}[\hat{Y}(\mathbf{x}_0; r) - Y(\mathbf{x}_0; r)] = 0$, that yields a system of equations known as universality conditions, $\boldsymbol{\omega}^\intercal \mathbf{X} = X_0^\intercal$, where $X_0 \in \mathbb{R}^p$ is the vector of known basis functions at the location of interest. Next, we formulate the unconstrained minimization problem of the prediction variance with multiple Lagrange multipliers $\boldsymbol{\lambda} \in \mathbb{R}^p$ to include the universality conditions. The solution is, $\boldsymbol{\omega}_{\mathrm{UK}} = \boldsymbol{\Gamma}_{\mathrm{UK}}^{-1} \boldsymbol{\gamma}_{\mathrm{UK}}$, where $\boldsymbol{\omega}_{\mathrm{UK}} = [\boldsymbol{\omega}^\intercal \ \boldsymbol{\lambda}_{\mathrm{UK}}^\intercal]^\intercal \in \mathbb{R}^{n+p}$ is a stacked vector that contains the weights $\boldsymbol{\omega}$ and the Lagrange multipliers $\boldsymbol{\lambda}_{\mathrm{UK}}$ to minimize the mean square prediction error. The non-singular matrix $\boldsymbol{\Gamma}_{\mathrm{UK}} \in \mathbb{R}^{(n+p)\times(n+p)}$ captures the

redundancy of measurements and is given by,

$$\begin{aligned}\boldsymbol{\Gamma}_{\text{UK}} &= \begin{bmatrix} \gamma(\mathbf{x}_1,\mathbf{x}_1) & \dots & \gamma(\mathbf{x}_1,\mathbf{x}_n) & 1 & X_2(\mathbf{x}_1) & \dots & X_p(\mathbf{x}_1) \\ \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ \gamma(\mathbf{x}_n,\mathbf{x}_1) & \dots & \gamma(\mathbf{x}_n,\mathbf{x}_n) & 1 & X_2(\mathbf{x}_n) & \dots & X_p(\mathbf{x}_n) \\ 1 & \dots & 1 & 0 & 0 & \dots & 0 \\ X_2(\mathbf{x}_1) & \dots & X_2(\mathbf{x}_n) & 0 & 0 & \dots & 0 \\ \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ X_p(\mathbf{x}_1) & \dots & X_p(\mathbf{x}_n) & 0 & 0 & \dots & 0 \end{bmatrix} \\ &\coloneqq \begin{bmatrix} \boldsymbol{\Gamma} & \mathbf{X} \\ \mathbf{X}^\intercal & \mathbf{0}_{p\times p} \end{bmatrix},\end{aligned}$$

The semivariogram vector $\boldsymbol{\gamma}_{\text{UK}} \in \mathbb{R}^{(n+p)}$ considers the *closeness* of the measurements to the location of interest $\mathbf{x}_0$,

$$\begin{aligned}\boldsymbol{\gamma}_{\text{UK}} &= \begin{bmatrix} \gamma(\mathbf{x}_0,\mathbf{x}_1) & \dots & \gamma(\mathbf{x}_0,\mathbf{x}_n) & 1 & X_2(\mathbf{x}_0) & \dots & X_p(\mathbf{x}_0) \end{bmatrix}^\intercal \\ &\coloneqq \begin{bmatrix} \boldsymbol{\gamma}_0 \\ X_0 \end{bmatrix}. \end{aligned} \tag{3.63}$$

The decoupled coefficients in terms of the covariance matrix yield,

$$\boldsymbol{\omega}^\intercal = \Big(\mathbf{c}_0 + \mathbf{X}(\mathbf{X}^\intercal\boldsymbol{\Sigma}^{-1}\mathbf{X})^{-1}(X_0 - \mathbf{X}^\intercal\boldsymbol{\Sigma}^{-1}\mathbf{c}_0)\Big)^\intercal\boldsymbol{\Sigma}^{-1}, \tag{3.64}$$

with Lagrange multipliers,

$$\boldsymbol{\lambda}_{\text{UK}}^\intercal = -(X_0 - \mathbf{X}^\intercal\boldsymbol{\Sigma}^{-1}\mathbf{c}_0)^\intercal(\mathbf{X}^\intercal\boldsymbol{\Sigma}^{-1}\mathbf{X})^{-1}. \tag{3.65}$$

Hence, the predictive distribution of UK with a covariance matrix is,

$$\hat{Y} \mid Y, \mathbf{x}, r \sim \mathcal{N}\Big([\mathbf{c}_0\Sigma^{-1} + (X_0 - \mathbf{c}_0\boldsymbol{\Sigma}^{-1}\mathbf{X})(\mathbf{X}^\intercal\boldsymbol{\Sigma}^{-1}\mathbf{X})^{-1}\mathbf{X}^\intercal\boldsymbol{\Sigma}^{-1}]Y, \\ C(\mathbf{0}_m) - \mathbf{c}_0\boldsymbol{\Sigma}^{-1}\mathbf{c}_0^\intercal + (X_0 - \mathbf{c}_0\boldsymbol{\Sigma}^{-1}\mathbf{X})(\mathbf{X}^\intercal\boldsymbol{\Sigma}^{-1}\mathbf{X})^{-1}(X_0 - \mathbf{c}_0\boldsymbol{\Sigma}^{-1}\mathbf{X})^\intercal\Big) \tag{3.66}$$

3.2.4 Model-Based Learning Framework

In this section, we discuss the structure and the algorithm of the prediction technique.

Learning Structure

The two-step process is depicted in Fig. 3.9. We start by collecting measurements of communication performance (SNR) along with the vehicle range. Given those measurements we seek to predict the communication performance at unvisited locations. The first step is the training of the Gaussian random field to obtain a covariance matrix, while the second is spatial prediction at unvisited locations with universal kriging. The objective of the first step is to determine the most suitable covariance function and its parameters characterizing the underlying latent process. The block of the covariance matrix is depicted in light red. The goal of the second step is to predict the SNR at unvisited locations and its corresponding variance, where their blocks are depicted in light red accordingly.

The training step comprises three modules: i) the data detrending; ii) the parameter estimation; and iii) the iterative training. The data detrending includes the hybrid basis function formulation (3.47) and the OLS computation (3.48). Next, the detrended measurements are used to compute the candidate semivariogram functions (3.51), (3.52), (3.53). The semivariograms are provided to the estimation module which is also a multistage process. The estimation module first computes the covariance parameters to be used as initial con-

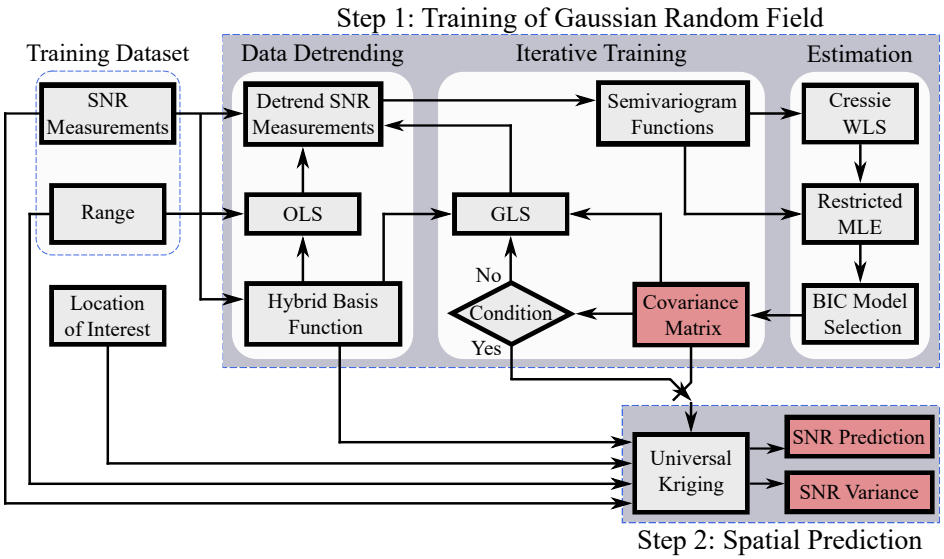


Figure 3.9: The two-step learning process. The first step is the training of the Gaussian random field that yields a covariance matrix and the second step the spatial prediction of the communication performance.

ditions, by employing the Cressie and Hawkins robust experimental semivariogram (3.50) and a weighted least squares estimation with Cressie weights (3.54). The next stage is the REML estimation that optimizes the objective likelihood function (3.55) and results in three bias-free covariance parameter vectors. The last stage of the estimation module considers the selection of the most suitable covariance model among the three candidates with the posterior BIC (3.57). Whenever the posterior probability of BIC indicates suitability of less than a probability threshold, we compute a nested semivariogram. The last module describes an iterative training for the selection of the covariance matrix. Since we have obtained a covariance matrix, the mean estimates can be improved by computing the GLS (3.59). Subsequently, we recompute the residual random function and run the estimation module to obtain a new covariance matrix. The training iterates until the parameters of the covariance matrix converge (3.61). For the numerical experiments reported herein, convergence requires no more than two iterations.

The second step is the spatial prediction. Given the measurements, the model-based basis

functions, and the covariance matrix from the previous step we use the location of interest to solve the universal kriging and obtain the kriging weights (3.64),(3.65). Finally, we predict the SNR at the location of interest and corresponding SNR variance (3.66).

Algorithm

The main routine is presented in Algorithm 1. The `initialConditions` module assigns initial values to the semivariogram parameter vector $\widehat{\boldsymbol{\theta}}^{(0)}$. More specifically, the partial sill σ^2 is assumed to be the variance of the residual measurements (3.49), the nugget effect τ^2 and the semivariogram range α are selected according to the sensor sensitivity and characteristics respectively. The initial covariance matrix estimate $\boldsymbol{\Sigma}(\widehat{\boldsymbol{\theta}}^{(0)})$ is set equal to the identity matrix. Next, the algorithm proceeds to the iterative parameter estimation process. We consider three semivariogram functions (3.51), (3.52), (3.53) at each iteration. The `basis` function computes $\mathbf{X}$ according to (3.47). The `GLS` function implements the GLS (3.59) to estimate the mean regressor parameters $\widehat{\boldsymbol{\beta}}^{(s)}$. Note that in the first iteration the initial covariance matrix is the identity matrix, and hence the algorithm implements an OLS regression (3.48). The function `detrend` is employed to compute the residual measurements (or detrended data) $\tilde{Y}^{(s)}$ by subtracting the estimated spatial trend from the measurements (3.49). With the detrended data, the function `CWLS` computes initial values for the estimation of the semivariogram parameter vector $\widehat{\boldsymbol{\theta}}_k^{(s-1)}$ by solving a WLS minimization problem (3.54). Next, the `REML` module implements the REML (3.55) to estimate the semivariogram parameter vector $\widehat{\boldsymbol{\theta}}_k^{(s)}$. The `BIC` function calculates the BIC (3.56) and the `diffBIC` computes the difference of each candidate with the lowest BIC*. Then, the `postBIC` calculates the posterior BIC (3.57) that assign probabilities of suitability for each candidate model with the underlying latent process. When the highest probability of the posterior BIC falls below a threshold φ, the `nested` function computes the covariance matrix with a nested semivariogram (3.58). The

Algorithm 1 Learning of UWA Communication Performance

Input: Y, $\mathbf{x}$, r, $\mathbf{x}_0$, n, p, p_k, γ, φ η
Output: $\widehat{Y}$, $\mathrm{Var}\{\widehat{Y}\}$

$\widehat{\boldsymbol{\theta}}^{(0)} \leftarrow \texttt{initialConditions}(Y)$
$\boldsymbol{\Sigma}(\widehat{\boldsymbol{\theta}}^{(0)}) \leftarrow I_n$; $k \leftarrow 0$;
$\mathbf{X} \leftarrow \texttt{basis}(\mathbf{x}; r)$;
for $s = 1$ to S **do** ▷ Start training
  $\widehat{\boldsymbol{\beta}}^{(s)} \leftarrow \texttt{GLS}(Y, \mathbf{X}, \boldsymbol{\Sigma}(\widehat{\boldsymbol{\theta}}^{(s-1)}))$;
  $\tilde{Y}^{(s)} \leftarrow \texttt{detrend}(Y, \mathbf{X}, \widehat{\boldsymbol{\beta}}^{(s)})$; ▷ Non-stationarity
  for each $\gamma \in \mathcal{C}$ **do**
    $\widehat{\boldsymbol{\theta}}_k^{(s-1)} \leftarrow \texttt{CWLS}(\tilde{Y}, \gamma, \widehat{\boldsymbol{\theta}}^{(s-1)})$; ▷ Robustness
    $\widehat{\boldsymbol{\theta}}_k^{(s)} \leftarrow \texttt{REML}(\tilde{Y}, \mathbf{X}, n, p, \gamma, \widehat{\boldsymbol{\theta}}_k^{(s-1)})$; ▷ Unbiasedness
    $M_k \leftarrow \boldsymbol{\Sigma}(\widehat{\boldsymbol{\theta}}_k^{(s)})$;
    $\mathrm{BIC}_k \leftarrow \texttt{BIC}(\tilde{Y}, n, p_k, \widehat{\boldsymbol{\theta}}_k^{(s)}, M_k)$;
    $k \leftarrow k + 1$;
  end for
  $\mathrm{BIC}^\star \leftarrow \min_{M_k \in \mathcal{M}}\{\mathrm{BIC}(M_k)\}$;
  for $k = 1$ to K **do**
    $\Delta_k \leftarrow \texttt{diffBIC}(\mathrm{BIC}_k, \mathrm{BIC}^\star)$;
  end for
  for $k = 1$ to K **do** ▷ Model selection
    $P(M_k \mid \tilde{Y}) \leftarrow \texttt{postBIC}(\Delta_k)$;
  end for
  if $\max_{M_k \in \mathcal{M}}\{P(M_k \mid \tilde{Y})\} < \varphi$ **then** ▷ Covariance
    $\boldsymbol{\Sigma}(\widehat{\boldsymbol{\theta}}^{(s)}) \leftarrow \texttt{nested}(P(M_k \mid \tilde{Y}), \widehat{\boldsymbol{\theta}}_k^{(s)})$;
  else
    $\boldsymbol{\Sigma}(\widehat{\boldsymbol{\theta}}^{(s)}) \leftarrow \max_{M_k \in \mathcal{M}}\{P(M_k \mid \tilde{Y})\}$;
  end if
  if $\|\widehat{\boldsymbol{\theta}}^{(s)} - \widehat{\boldsymbol{\theta}}^{(s-1)}\| \leq \eta$ **then** ▷ Iteration criterion
    break;
  end if
end for ▷ End training
$\widehat{Y}, \mathrm{Var}[\widehat{Y}] \leftarrow \texttt{UK}(Y, \mathbf{x}, r, \mathbf{X}, \mathbf{x}_0, \boldsymbol{\Sigma}(\widehat{\boldsymbol{\theta}}^{(s)}))$; ▷ Prediction

iterative training procedure is terminated when the semivariogram parameter estimation converges to an η-neighborhood (3.61). Finally, we utilize the estimated covariance matrix $\boldsymbol{\Sigma}(\widehat{\boldsymbol{\theta}}^{(s)})$ and the measurements to solve the universal kriging and obtain SNR prediction $\hat{Y}$ at the unvisited locations of interest $\mathbf{x}_0$ and its corresponding variance $\mathrm{Var}[\hat{Y}]$.

3.2.5 Computational Complexity

The time complexity of the training is $\mathcal{O}(n^3)$ for computing the inverse and determinant of the covariance matrix $\boldsymbol{\Sigma}$. These computations are performed repeatedly in (3.55) to find the hyperparameters $\boldsymbol{\theta}$ that maximize the log-restricted likelihood. Next, we store the inverse covariance $\boldsymbol{\Sigma}^{-1}$ and n measurements, which result in $\mathcal{O}(n^2+mn)$ space complexity. For small robots with limited RAM memory capacity, the space complexity may be more restrictive than the time complexity. The prediction mean and variance (3.66) require $\mathcal{O}(n)$ and $\mathcal{O}(n^2)$ computations respectively.

3.2.6 Simulations and Experiments

In this section, we provide simulations and experiments to demonstrate the efficacy of the proposed methodology.

Simulation Environment

The simulation environment is developed with a well-established, statistical UWA channel model that incorporates 34 parameters and interprets multipath formation, motion-induced Doppler, surface scattering, and large-scale variability of the channel geometry [100]. This channel model has been exhaustively compared to experimental data from multiple underwater missions, which varied in location, season, time duration, weather conditions, static nodes, and moving AUVs.

The SNR measurements consist of three components as described in (3.42). The channel gain (3.43) is computed for signal frequency $f = 25$ kHz, bandwidth $B = 5$ kHz, surface height 100 m, and vehicle depths $z_1 = 80$ m and $z_2 = 50$ m. The navigation depth corresponds

Table 3.1: Training with Exponential Semivariogram

Cases			Exponential Semivariogram Parameters	
Training Set	Validation Set	Bias	OK σ^2, α, τ^2	UK σ^2, α, τ^2
150 Long-distant	519	−10	93.27, 17822, 0	60.59, 12381, 0
		0	92.83, 17738, 0	61.00, 12378, 0
		+10	92.74, 17722, 0	60.79, 12382, 0
500 Short-distant	169	−10	80.91, 16888, 0	70.66, 14792, 0
		0	80.86, 16877, 0	71.51, 14969, 0
		+10	80.31, 16763, 0	71.15, 14895, 0

OK–Ordinary kriging; UK–Universal kriging.

Table 3.2: Training with Matérn Semivariogram

Cases			Matérn Semivariogram Parameters		
Training Set	Validation Set	Bias	OK σ^2, α, τ^2	UK σ^2, α, τ^2	Model-based UK σ^2, α, τ^2
150 Long-distant	519	−10	368.73, 2994, 0.20	364.42, 2994, 0.20	14.42, 711, 0.19
		0			
		+10			
500 Short-distant	169	−10	14.13, 548, 0.23	13.78, 548, 0.23	82.04, 1495, 0.26
		0			
		+10	111.17, 1611, 0.26		

OK–Ordinary kriging; UK–Universal kriging.

to shallow water, where the speed of sound can be considered constant [116]. We set the source level $S_l = 180$ dB which is a realistic value for UWA acoustic modems operating in such signal frequencies. The large-scale parameters, i.e. path gain and propagation delay, are computed using the Bellhop model [98]. The Doppler parameters were generated using first-order dynamics. Since the vehicles maintain constant velocity, the drifting parameters were neglected. Each vehicle depth remained constant during the simulation, yet the depth of each vehicle is different.

In addition, we impose local ambient noise to the synthetic data (denominator of (3.43)).

The local ambient noise is captured with: i) uniform noise; ii) linear noise; iii) single non-zero Gaussian distribution; and iv) two non-zero Gaussian distributions. We evaluate the ambient noise over a grid of points in the space $\mathcal{S} \coloneqq \mathbb{X} \times \mathbb{Y}$, where $\mathbb{X} = [-2000, -1990, \ldots, 3000]$ and $\mathbb{Y} = [0, 10 \ldots, 5000]$ in meters. The ambient noise for the space of interest outputs values $NL(\mathbf{x}) \in [7.75, 50]$ in dB, resulting in both mild and extreme environments.

The evaluation of the predictions is accomplished with two metrics. The first metric is the mean square error (MSE), $\text{MSE} = 1/n_u \sum_{u=1}^{n_u} (\hat{Y}(\mathbf{x}_{0,u}; r_u) - Y(\mathbf{x}_{0,u}; r_u))^2$, where n_u is the number of unknown responses at locations of interest. Next, the negative log predictive density (NLPD) [102] follows, $\text{NLPD} = -1/n_u \sum_{u=1}^{n_u} \log p(y_u \mid \mathbf{x}_{0,u}, r_u)$, where the distribution is provided by $p(y_u \mid \mathbf{x}_{0,u}, r_u) \sim \mathcal{N}(\hat{Y}(\mathbf{x}_{0,u}; r_u), \sigma^2_{\text{UK}}(\mathbf{x}_{0,u}; r_u))$. The NLPD loss characterizes not only the error of the mean value, but more importantly the uncertainty bound. More specifically, both under- and over-confident predictions are penalized.

Simulation Results

We compare five prediction techniques: i) the ordinary kriging (OK) with exponential semivariogram (3.52); ii) the OK with Matérn semivariogram (3.53); iii) the universal kriging (UK) with linear trend and exponential semivariogram (3.52); iv) the UK with linear trend and Matérn semivariogram (3.53); and v) the proposed model-based learning method with hybrid basis function and semivariogram model selected by the posterior BIC or formed as a nested structure. The OK formulation is discussed in [68]. In the first four prediction techniques, we select the exponential and the Matérn semivariogram functions, as they are widely used in the literature. Each agent collects measurements of communication performance (SNR) and vehicle range r from visited locations. Since the global paths are known, the agents are aware of their range at the unvisited locations. The GEOR package [105] is used to implement the geostatistical methodologies.

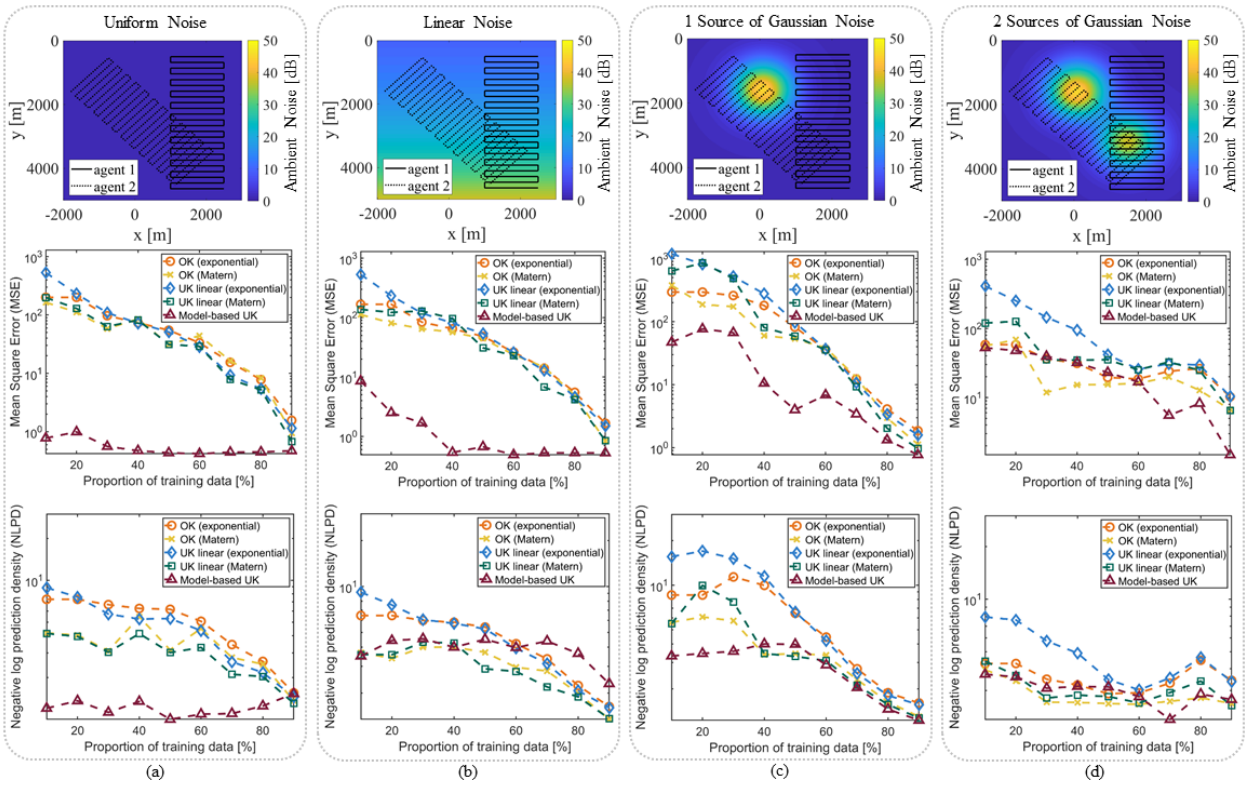


Figure 3.10: The color map on the top row depicts the ambient noise distribution that deteriorates the UWA communication performance. The solid black and dotted black lines correspond to the lawnmower paths of agent 1 and agent 2 respectively. In all cases we use 9 proportions of the training data to make predictions. (a) Uniform noise distribution case with MSE and NLPD computed for 9 proportions of the training data. (b) Linear noise distribution case with MSE and NLPD computed for 9 proportions of the training data. (c) One source of non-zero Gaussian noise distribution case with MSE and NLPD computed for 9 proportions of the training data. (d) Two non-zero Gaussian noise distribution case with MSE and NLPD computed for 9 proportions of the training data.

Training

For the evaluation of the robustness in training, we perform 30 simulations with added bias on the measurements. We consider one noise profile scenario of two non-zero Gaussian distributions. The trajectories of the mobile robots as well as the ambient noise distribution are illustrated in the top right image of Figure 3.10. The black solid and dotted line represent the lawnmower paths of agent 1 and 2 respectively. We consider two cases: i) the long-

distant prediction; and ii) the short-distant prediction. In the long-distant prediction case, each agent collects 75 measurements while in the short-distant case 250 measurements of SNR and range. We seek to predict the communication performance in the long-distant case of 260 and 259 and in the short-distant-case of 85 and 84 unvisited locations for agent 1 and 2 respectively. The effect of the bias to the semivariogram estimation, i.e. robustness, is investigated by adding a systematic error to the measurements. The added biases are: i) $+10$; ii) -10; and iii) no bias. We observe in Tables 3.1 and 3.2 that the added bias does not affect the training of the proposed technique, resulting in the same semivariogram function and semivariogram parameters. In both OK and UK methods with exponential semivariogram, the estimated parameters are clearly affected by the added bias. In the OK prediction method with Matérn semivariogram, the added bias affects only the long-distant case of $+10$ added bias, yet the difference is significant. The UK prediction method with Matérn semivariogram is not affected by the added bias. Note that the the posterior BIC selected the Matérn semivariogram as the true model. Evidently, when a statistical model selection methodology is not employed, yet the true semivariogram model is spontaneously selected, then the parameter estimation appears less variation with added bias. However, we cannot always rely on heuristic assumptions, ignoring statistical model selection methods. In addition, in many cases a single semivariogram function may not be adequate to fully describe the underlying latent process. After using the posterior BIC to select the true semivariogram function, the REML successfully removes the bias from the parameter estimation, regardless of the systematic error direction, i.e. sign of the bias. Therefore, the proposed methodology constitutes a robust and bias-free alternative of the maximum likelihood estimator.

Prediction

For the evaluation of the prediction we perform 180 simulations, comprising 9 training datasets at 4 ambient noise profile scenarios and 5 prediction techniques. The size of the training dataset varied proportionally from 10% up to 90% of the data. The remainder data act as the validation dataset of the learning process. The distant horizon of the extrapolation is associated to the proportion of the training data, e.g., 10% of training data correspond to the longest distant prediction and 90% to the shortest distant prediction. The spatial environmental conditions and the global path of the vehicles are shown in the top row of Figure 3.10. The MSE and NLPD are presented in the middle and bottom row of Figure 3.10 respectively.

In the first noise distribution scenario, i.e. uniform noise, randomness arises mostly from the statistical characterization of the UWA channel model (see Figure 3.10-(a)). That is mild ambient noise conditions, which often appear in deep ocean environments. In shallow water environments, uniform ambient noise occur when vehicles navigate in areas with no nearby shipping and mild weather conditions. Clearly, the proposed method outperforms the rest techniques both in terms of prediction accuracy and uncertainty quantification. Especially, for long-distant prediction the difference is significant, making our model-based approach three orders of magnitude more accurate in terms of MSE and the uncertainty bounds almost one order of magnitude more realistic acording to NLPD. As more data are incorporated in the training dataset, the rest methods improve their accuracy and uncertainty quantification metrics. However, only in the shortest distant prediction case, i.e. 90% training dataset, the rest methods are comparable with our technique. The results advocate that for mild ambient noise conditions the proposed model-based learning technique vastly outperforms the compared methods and can be safely used for long-distant extrapolation. Next, we impose linear ambient noise distribution to the UWA channel model, as presented

Table 3.3: Posterior BIC-based Selection of Semivariogram Function

	Semivariogram-posterior BIC [%]			
% of Data	Uniform Noise	Linear Noise	1 Gaussian Source Noise	2 Gaussian Source Noise
10	S-33; E-33; M-34	S	M	M
20	S-31; E-31; M-38	S-51; E-49	M	M
30	S-32; E-32; M-36	S	M	M
40	S-33; E-32; M-35	S	M	M
50	S-32; E-32; M-36	S	M	M
60	S-26; E-32; M-42	S	M	M
70	S-9; E-87; M-4	S	M	M
80	S-33; E-33; M-34	M	M	M
90	S-16; E-67; M-17	M	M	M

S–Spherical; E–Exponential; M–Matérn.

in Figure 3.10-(b). Linear ambient noise corresponds to a spatially large source of noise that almost equally and progressively deteriorates the communication performance of the vehicles. Similarly to the uniform noise case, the results show better predictions from all other methods, where after the 40% training dataset the predicted values become accurate with almost zero error values. Yet, the uncertainty of the proposed technique is overconfident, reporting similar NLPD values with the rest methods. The results reveal that for the linear ambient noise distribution scenario, our methodology outperforms the rest techniques and produces accurate predictions for 40% and larger training datasets. However, the uncertainty quantification is overconfident in all cases.

A single spatially small and intense source of noise is presented in Figure 3.10-(c). Such noise sources often appear in Nature and they consider to be the main reason of conservativeness in long-distant extrapolated predictions. Apparently, the spatially small source of noise obscure the UK methods and slightly favors the OK techniques. However, the proposed model-based UK methodology outperforms the rest techniques by one order of magnitude on the mean predictions and quantifies the uncertainty better according to NLPD. Thus, our learning

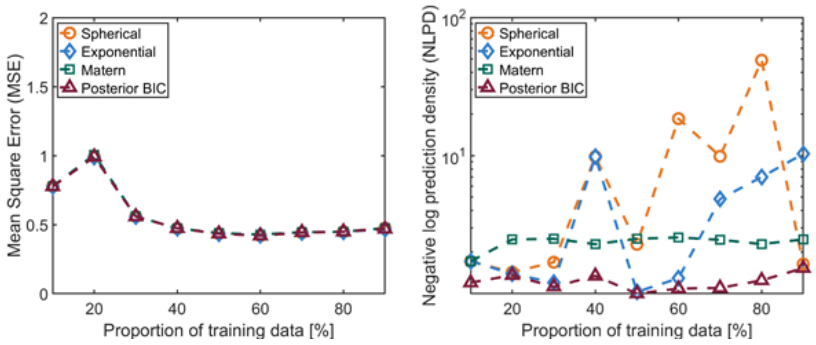


Figure 3.11: Comparison of nested semivariogram with the three candidate semivariogram functions for the uniformly distributed noise scenario.

method advocates to higher level of robustness for unexpected spatially small and intense source of noise. We extend the previous case using two spatially small and intense sources of noise with different magnitude, as illustrated in Figure 3.10-(d). The proposed method outperforms the rest techniques for the majority of the training dataset cases in terms of MSE. The biggest competitor is the most parsimonious form of prediction the OK, yet in only three out of nine training datasets the OK produces lower communication performance error values. The uncertainty quantification is reasonable in all techniques except for long distant predictions of UK with linear trend and exponential semivariogram. Although in unexpected noisy environments the model-based techniques are expected to be inefficient, our method outperforms the other techniques in the vast majority of the cases in terms of prediction and quantifies reasonably well the uncertainty.

In addition to the evaluation of prediction metrics, the effectiveness of nested semivariogram is illustrated. In Table 3.3, we list the semivariograms as selected by the posterior BIC for all 9 training datasets and 4 ambient noise profile scenarios. Interestingly, in the linear noise distribution scenario the posterior BIC changes the semivariogram function from spherical to Matérn at the 80% and 90% training datasets. This means that even if we select one semivariogram model for a specific case, there are no guarantees that the same semivariogram will describe the latent process with updated training datasets. Moreover, we observe in

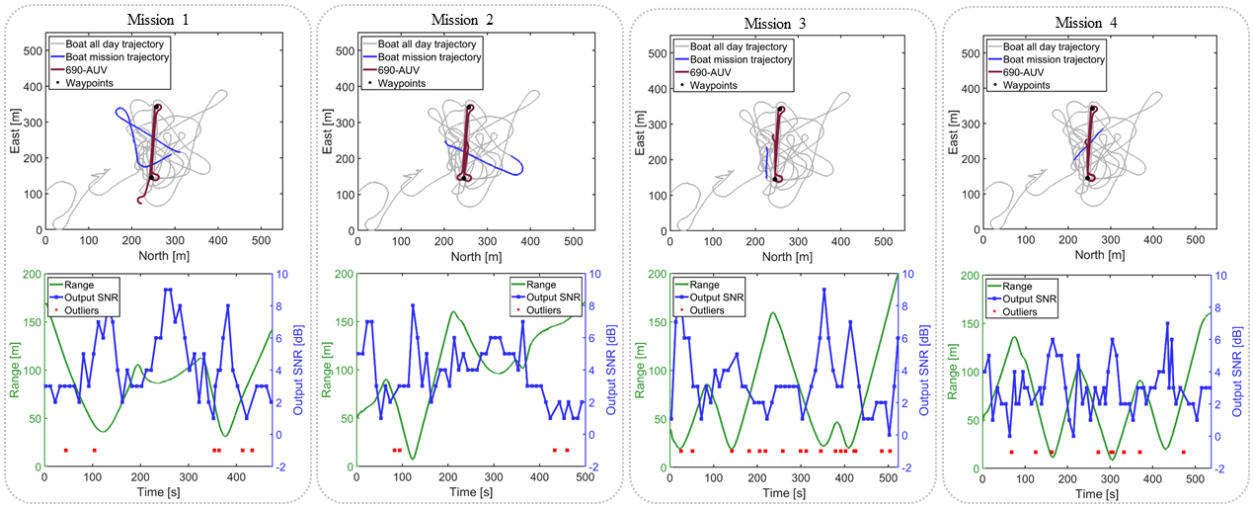


Figure 3.12: The top row depicts the trajectories of the SV and the 690-AUV. The light gray line corresponds to the SV trajectory during the day, the blue line depicts the trajectory of the SV for the current mission, and the maroon colored line represents the path of the 690-AUV. The bottom row shows the vehicle range and output SNR of the corresponding mission.

Figure 3.13: The VIRGINIA TECH 690-AUV used in the field trials.

Table 3.3 that all semivariograms are nested for the uniform noise distribution, thus we focus our attention on this scenario. In Figure 3.11, we compare the MSE and NLPD of the nested semivariogram with the three candidate semivariograms. Notably, the mean predictions are identical for single and nested semivariograms. Yet, the uncertainty quantification for nested functions is consistently better with all training datasets. This advocates that the proposed technique with nested semivariogram quantifies more realistically the uncertainty, without compromising the accuracy.

Field Trials

The experimental data were collected from field trials conducted at Claytor lake near Dublin, VA in December 2019. A manned surface vehicle (SV) and the Virginia Tech 690-AUV [86] were used in the field trials. The SV is equipped with an omnidirectional acoustic transducer and a Woods Hole Oceanographic Institute (WHOI) Micromodem-2 [39]. The AUV (pictured in Figure 3.13), can operate at a depth of 500 meters for up to 24 hours. It is equipped with a suite of navigational sensors, sidescan sonar, and the WHOI Micromodem-2.

The SV transmitted acoustic packets to the AUV every 10 seconds. The acoustic transmission each lasted 3.5 seconds and had a carrier frequency of $f = 25$ kHz and bandwidth of $B = 4$ kHz. The transponder mounted to the SV was submerged at a depth $z_1 = 1$ m while the AUV traveled at a depth of $z_2 = 3.35$ m. That is clearly shallow water navigation which makes the acoustic communication even more challenging. The maximum depth of Claytor lake is 35 m. We conducted four missions whose trajectories are illustrated in the top row of Figure 3.12. The SV trajectory throughout the day is shown in light gray, the SV trajectory during each mission is highlighted in blue, and the AUV trajectory is demonstrated in maroon. In all missions, the AUV traversed identical waypoint paths (waypoints shown in black circles). The speed of the AUV was constant at 1.6 m/s. The SV was manned-driven with different path for each mission. The missions were conducted in mild weather conditions with no nearby shipping. Note that the field tests were conducted in December, when no extramural activities take place at the lake. Thus, the only obvious source of ambient noise was from the SV. The noise arising from the SV was time-varying, as it was driven at different low speeds. In missions 1 and 2 the SV used the propulsion system to navigate and traversed longer paths. That is to intentionally create ambient noise. In missions 3 and 4 the propulsion system of the SV was not used, i.e. the SV was floating, which resulted in lower ambient noise. We used GPS for the SV position, while the AUV position was estimated by

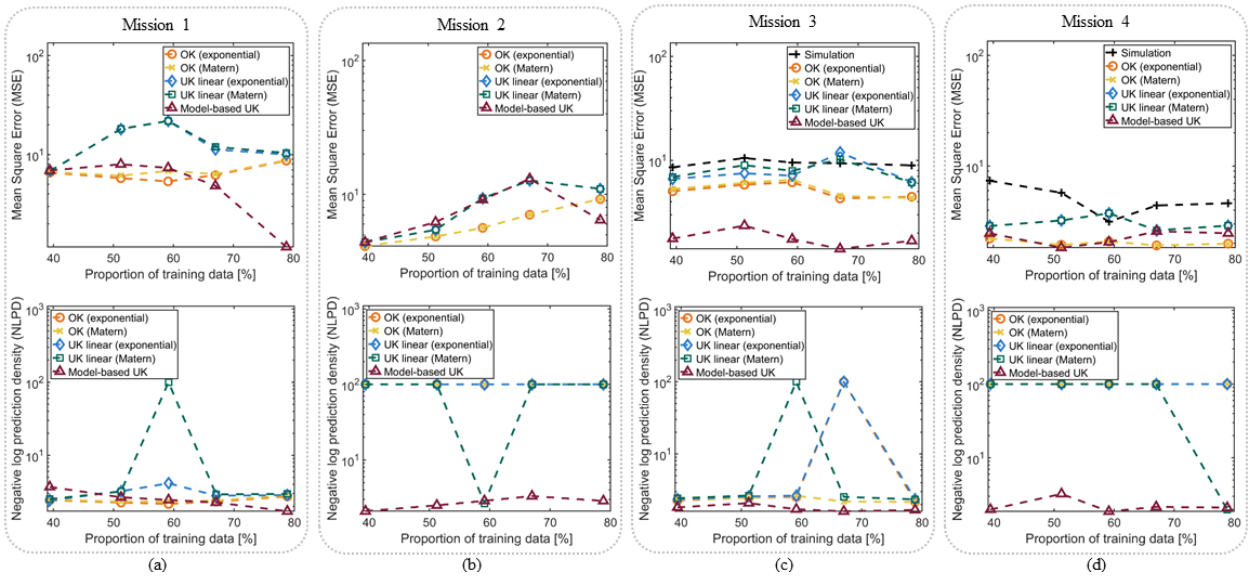


Figure 3.14: The eMSE and eNLPD metrics for all 5 prediction methods and all missions. In some cases, the uncertainty reported almost zero uncertainty, which significantly increased the NLPD values. To emphasize on the low values, we set the upper bound of the NLPD to be 100.

the AUV's unscented Kalman filter (UKF).

The SNR measurements were collected by the WHOI Micromodem-2 at the output of the equalizer. This SNR metric is used in the literature for the evaluation of communication performance [36, 37, 124]. The disadvantages of the output are: i) averages the SNR for a communication event; and ii) provides positive rounded numbers—compromising the SNR measurement accuracy. In the bottom row of Figure 3.12, we present the SNR in blue solid line, the corresponding vehicle range in green solid line, and the outliers in red squares. Clearly, there exists a coupling of the range and the SNR. More specifically, as the range increases the SNR reduces. Note that the default value of the WHOI modem to report output SNR outliers is -9.99 dB [39], yet we plot them at -1.00 dB for scaling purposes. In Table 3.4, we list the statistics of communication events.

Table 3.4: Output SNR Values for Four-Waypoint Experiments

Mission	Duration [s]	Transmitted Signals	SNR Occurrence	SNR Success	Outliers
1	475.44	48	54	46	6
2	498.76	50	52	48	4
3	523.64	53	65	48	17
4	538.36	54	68	52	9

Experimental Results

Similarly to Section 3.2.6, we compare five prediction techniques. However, the MSE and NLPD cannot be used, as the true value of the communication performance at the location of the measurements is unknown during field trials; measurements are corrupted by multiple sources of error. Hence, to proceed with our analysis we refer to the metrics as empirical MSE (eMSE) and empirical NLPD (eNLPD) accordingly. The eMSE and eNLPD values for various proportions of data are presented in Figure 3.14. The 40% proportion of data includes at least 20 measurements for the longest distant prediction, while the 80% proportion of data corresponds to the shortest distant prediction case. In some cases the predicted values report almost zero uncertainty, making the eNLPD values very high. At these cases, we consider that the corresponding method has failed, as uncertainty quantification is a key element in communication performance prediction. Since we are interested in evaluating low scaled eNLPD values, we set its upper bound to be 100. In Figure 3.15 the prediction mean and standard deviation of three techniques: i) OK with exponential semivariogram; ii) UK with linear trend and exponential semivariogram; and iii) our method are presented. We select the exponential semivariogram for both OK and UK, because they provide better predictions in terms of eMSE and eNLPD (see Figure 3.14). The top row of Figure 3.15 corresponds to 40% proportion of data, the middle row to 59%, and the bottom row to 80%.

In mission 1 the high ambient noise affects the performance of the UK techniques. Our

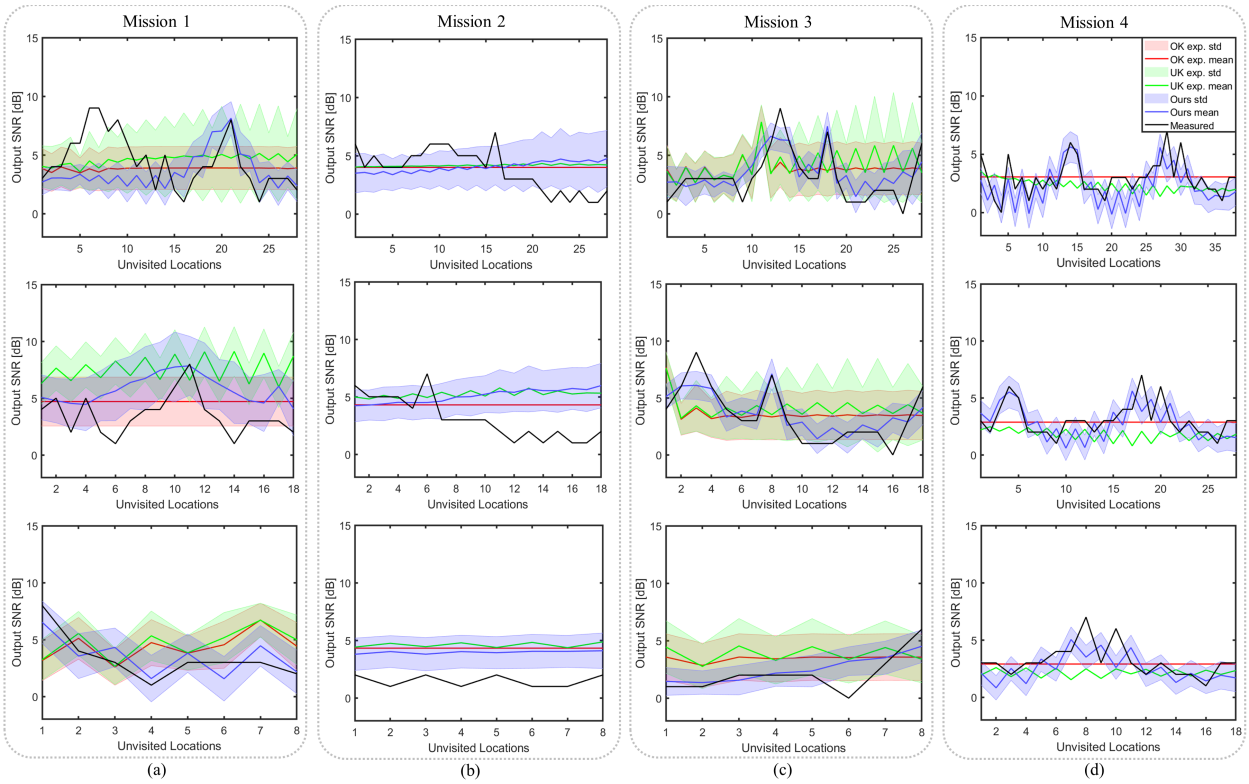


Figure 3.15: The prediction mean and standard deviation for three methods and three proportions of data in four missions.

method performs similarly to the OK methods for long-distant predictions and profoundly better for short-distant predictions. The eNLPD values are acceptable in all predictions techniques and cases, except one case of UK prediction with Matérn semivariogram. In mission 2 the high ambient noise affects the performance of all prediction techniques. Only the proposed method quantifies the uncertainty, while all other methods fail as shown in Figure 3.15-(b). Both parsimonious methods of OK report lower error values, yet with zero variance. Although the error metrics of the proposed technique are not satisfactory for this mission, the uncertainty of the proposed method is quantified, as indicated by the eNLPD in Figure 3.14-(b) and the prediction plots in Figure 3.15-(b). Paradoxically, all methods produce higher error values as more measurements are collected. In mission 3 and 4, the low ambient noise results in better predictions for our method than the rest techniques.

Table 3.5: Posterior BIC-based Selection of Semivariogram Function

	Semivariogram-posterior BIC [%]			
% of Data	Mission 1	Mission 2	Mission 3	Mission 4
40	S	S-33; E-33; M-34	E	S-34; E-32; M-34
51	S	S-33;E-33;M-34	E	S
59	S	S-32; E-33; M-35	E	S-33; E-33; M-34
67	S-29; E-37; M-34	S-33; E-32; M-35	E	S-33; E-33 ;M-34
80	S-38; E-29; M-33	S-36; E-31; M-33	E	S-33; E-33; M-34

S–Spherical; E–Exponential; M–Matérn.

Particularly, in mission 3 the error and uncertainty metrics of our technique are significantly better from the rest methods as illustrated in Figure 3.15-(c). In mission 4, all techniques provide acceptable results in terms of eMSE, yet the proposed method is the most accurate and the only one that quantifies the uncertainty. All other methods fail. In addition, even though the other techniques provide low eMSE values, their uncertainty is overconfident as indicated by the eNLPD values in Figure 3.14-(d). Realistic uncertainty bounds are reported only from the proposed methodology, while all other techniques provide predictions with zero variance as presented in Figure 3.15-(d). Note that for higher vehicle depth the ambient noise deteriorates, favoring the proposed methodology, as noise scenarios are similar to missions 3 and 4.

In Table 3.5, we list the selected semivariograms based on the posterior BIC. It is evident that there is no dominant semivariogram function and that the nested semivariogram was employed in many cases. This emphasizes the importance of nested models and the necessity of statistical model selection techniques for field trials in complex environments.

3.2.7 Conclusion

This chapter proposes a model-based, data-driven learning technique for prediction of underwater acoustic communication performance in AUVs beyond the observation area. In both ordinary kriging (OK) and universal kriging (UK) methods the estimated parameters are affected by the artificially added bias, leading to different parameter values. We show that the proposed model-based learning yields accurate predictions, outperforming up to three orders of magnitude other kriging methods in simulations. More specifically, for all ambient noise profile scenarios both OK and UK prediction methods produce high error values and quantify the uncertainty poorly. Moreover, the nested semivariogram function improves drastically the uncertainty quantification. In addition, experimental results reveal significantly better predictions with our method for low and high ambient noise environments. The proposed technique reports realistic uncertainty bounds in all missions, which other mission methods often fail to generate. In unpredictable and high ambient noise environments, our method outperforms in prediction accuracy the techniques assessed herein.

A disadvantage of the proposed technique arises from the computational requirements of the iterative training. More specifically, the training step entails the computation of three candidate covariance functions with corresponding parameters at every iteration. We found in practice that the recursive method usually terminates after two iterations, for which the execution of the training step requires six times more computations than the traditional OK and UK methods with fixed semivariogram functions. Another drawback that is subject to all techniques stems from the communication. In particular, all agents must communicate their measurements to every other agent. To this end, our focus in ongoing work is on decentralized approximate methods to implement kriging in multi-robot systems with reduced computational complexity and limited inter-vehicle communication [69]. This will allow even large networks with big data to use the proposed technique in real-time.

Chapter 4

Decentralized Gaussian Processes

In this chapter, we propose methodologies for fully decentralizing Gaussian processes (GPs) [27, 43, 104] from training to prediction, so that they can be implemented efficiently on teams of agents. More specifically, we propose three distributed optimization techniques to implement GP hyperparameter training with maximum likelihood estimation (MLE), based on the alternating direction method of multipliers (ADMM) [12]. Next, we synthesize 13 decentralized approximate methods to perform GP prediction with aggregation of GP experts [77], using iterative and consensus protocols [10, 93, 130]. In addition, we introduce a covariance-based nearest neighbor selection strategy that enables a subset of agents to perform predictions.

4.1 Preliminaries and Problem Statement

In this section, we discuss the foundations of algebraic graph theory, overview GPs, describe existing distributed approximate methods for scalable GPs, and define the problem of decentralized, scalable GP training and prediction.

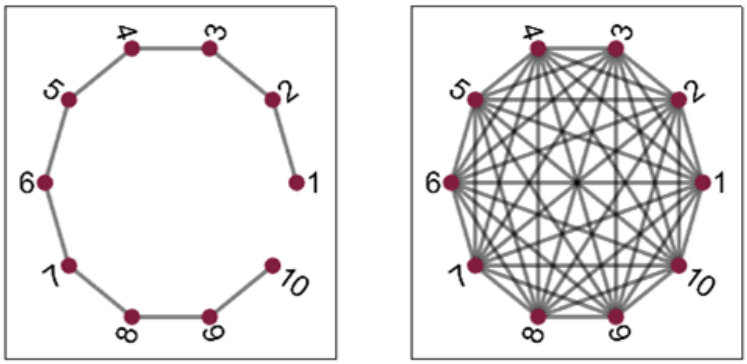


Figure 4.1: Graph topologies of multi-robot systems.

4.1.1 Foundations

The set of all positive real numbers $\mathbb{R}_{>0}$ and the set of all non-negative real numbers $\mathbb{R}_{\geq 0}$. We denote by I_n the identity matrix of $n \times n$ dimension. The vector of n zeros is represented as $\mathbf{0}_n$ and the matrix of $n \times m$ zeros as $\mathbf{0}_{n \times m}$. The superscript in parenthesis $y^{(s)}$ denotes the s-th iteration of an estimation process. The cardinality of the set K is denoted $\text{card}(K)$, the absolute values is denoted $|\cdot|$, the L_2 norm is denoted $\|\cdot\|_2$, and $\|\cdot\|_\infty$ denotes the infinity norm. The notation $\overline{\lambda}(\boldsymbol{F})$ and $\underline{\lambda}(\boldsymbol{F})$ denote the maximum and minimum eigenvalue of matrix $\boldsymbol{F}$ respectively. The i-th row of matrix $\boldsymbol{F}$ is denoted $\text{row}_i\{\boldsymbol{F}\}$, the j-th entry of the i-th row is denoted $[\text{row}_i\{\boldsymbol{F}\}]_j$, and the i-th element of a vector $\boldsymbol{x}$ is denoted $[\boldsymbol{x}]_i$ or x_i. A collection of elements that comprise a vector $\boldsymbol{x} \in \mathbb{R}^N$ is denoted $\{x_i\}_{i=1}^N$.

Suppose a network consists of M agents that can perform local computations. The network is described by an undirected time-varying graph $\mathcal{G}(t) = (\mathcal{V}, \mathcal{E}(t))$, where $\mathcal{V} = 1, \ldots, M$ is the set of nodes and $\mathcal{E}(t) \subseteq \mathcal{V} \times \mathcal{V}$ the set of edges at time t. An undirected graph implies that for all t the communication is bidirectional. Nodes represent agents and edges their communication. The neighbors of the i-th robot are denoted $\mathcal{N}_i(t) = \{j \in \mathcal{V} : (i,j) \in \mathcal{E}(t)\}$. The adjacency matrix of $\mathcal{G}(t)$ is denoted $\boldsymbol{A}(t) = [a_{ij}] \in \mathbb{R}^{M \times M}$, where $a_{ij} = 1$ if $(i,j) \in \mathcal{E}(t)$

and $a_{ij} = 0$ otherwise. Similarly, the degree matrix of $\mathcal{G}(t)$ is denoted $\boldsymbol{D}(t) = [d_{ij}] \in \mathbb{R}^{M\times M}$ and is diagonal with $d_i = \sum_{j=1}^{M} a_{ij}$. The graph Laplacian is defined as $\mathcal{L}(t) \coloneqq \boldsymbol{D}(t) - \boldsymbol{A}(t)$. The maximum degree is denoted $\Delta = \max_i\{\sum_{j\neq i} a_{ij}\}$ and represents the maximum number of neighbors in the graph. The Perron matrix is defined as $\mathcal{P}(t) \coloneqq I_M - \epsilon\mathcal{L}(t)$, where ϵ is a parameter with range $\epsilon \in (0, 1]$. The maximum shortest distance between any pair of nodes in $\mathcal{G}$ is denoted $\text{diam}(\mathcal{G})$. If the adjacency matrix A is irreducible, then the graph $\mathcal{G}$ is strongly connected [93]. In addition, a graph $\mathcal{G}$ is strongly complete if every robot can communicate to every other robot in the graph. We consider three decentralized network topologies as presented in Fig.4.1.

Assumption 4.1. *[78] There exists a positive integer $\gamma \in \mathbb{Z}_{\geq 0}$ such that for all time t the graph $\mathcal{H} = (\mathcal{V}, \mathcal{E}(t) \cup \mathcal{E}(t\gamma + 1) \cup \ldots \cup \mathcal{E}((t + 1)\gamma - 1)$ is strongly connected.*

4.1.2 Gaussian Processes

Let the observations be modeled by,

$$y(\boldsymbol{x}) = f(\boldsymbol{x}) + \epsilon, \tag{4.1}$$

where $\boldsymbol{x} \in \mathbb{R}^D$ is the input location with D the input space dimension, $f(\boldsymbol{x}) \sim \mathcal{GP}(0, k(\boldsymbol{x}, \boldsymbol{x}'))$ is a zero-mean GP with covariance function $k : \mathbb{R}^D \times \mathbb{R}^D \to \mathbb{R}$, and $\epsilon \sim \mathcal{N}(0, \sigma_\epsilon^2)$ is the i.i.d. measurement noise with variance σ_ϵ^2. We employ the separable squared exponential covariance function,

$$k(\boldsymbol{x}, \boldsymbol{x}') = \sigma_f^2 \exp\left\{-\sum_{d=1}^{D} \frac{\|\boldsymbol{x} - \boldsymbol{x}'\|^2}{l_d^2}\right\}, \tag{4.2}$$

where σ_f^2 is the signal variance and l_d the length-scale hyperparameter at the d-th direction of the input space. The goal of GPs is to infer the underlying latent function f given the

data $\mathcal{D} = \{\boldsymbol{X}, \boldsymbol{y}\}$, where $\boldsymbol{X} = \{\boldsymbol{x}_n\}_{n=1}^N$ the inputs, $\boldsymbol{y} = \{y_n\}_{n=1}^N$ the outputs, and N the number of observations.

Training

A GP is trained to find the hyperparameters $\boldsymbol{\theta} = \{l_1, \dots, l_D, \sigma_f^2, \sigma_\epsilon^2\} \in \Theta \subset \mathbb{R}^{D+2}$ that maximize the marginal log-likelihood,

$$\mathcal{L} = \ln p(\boldsymbol{y} \mid \boldsymbol{X}) = -\frac{1}{2}\Big(\boldsymbol{y}^\intercal \boldsymbol{C}_\theta^{-1}\boldsymbol{y} + \ln|\boldsymbol{C}_\theta| + N \ln 2\pi\Big), \tag{4.3}$$

where $\boldsymbol{C}_\theta = \boldsymbol{K} + \sigma_\epsilon^2 I_N$ is the positive definite (PD) covariance matrix with $\boldsymbol{K} = k(\boldsymbol{X}, \boldsymbol{X}) \succeq 0 \in \mathbb{R}^{N \times N}$ the positive semi-definite (PSD) correlation matrix. The minimization problem employs the negative marginal log-likelihood (NLL) function,

$$\begin{aligned} \text{(P1)} \quad \hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \quad & \frac{1}{2}\Big(\boldsymbol{y}^\intercal \boldsymbol{C}_\theta^{-1}\boldsymbol{y} + \ln|\boldsymbol{C}_\theta| + \frac{N}{\ln} 2\pi\Big) \\ \text{s.to} \quad & \theta_j > 0, \quad \forall j = 1, \dots, D+2. \end{aligned} \tag{4.4}$$

The bound constraints (4.4) on the length-scales l_d ensure that the correlation matrix is PSD. Additionally, in practice even small noise variance σ_ϵ^2 is useful to avoid an ill-conditioned covariance matrix. First-order iterative methods (e.g., conjugate gradient descent) or second-order iterative methods with approximated Hessian (e.g., L-BFGS-B [16]) are widely used to tackle (P1). Note that the NLL in (P1) is non-convex with respect to the hyperparameters $\boldsymbol{\theta}$ and usually multiple starting locations are randomly selected to ensure global optimality [21]. Both optimization approaches require the computation of the gradient of (P1) which

is given by,

$$\frac{\partial \mathcal{L}(\boldsymbol{\theta})}{\partial \theta_j} = \frac{1}{2}\mathrm{tr}\left\{(\boldsymbol{C}_\theta^{-1} - \boldsymbol{C}_\theta^{-1}\boldsymbol{y}\boldsymbol{y}^\intercal\boldsymbol{C}_\theta^{-1})\frac{\partial \boldsymbol{C}_\theta}{\partial \theta_j}\right\}, \tag{4.5}$$

The partial derivative of the covariance matrix $\partial \boldsymbol{C}_\theta / \partial \theta_j$ depends on the selected covariance function $k(\cdot,\cdot)$. For our selection (4.2), the partial derivative is provided in Appendix A.1.

Prediction

After obtaining the hyperparameters $\hat{\boldsymbol{\theta}}$, the predictive distribution of the location of interest $\boldsymbol{x}_* \in \mathbb{R}^D$ conditioned on the data $\mathcal{D}$ yields $p(\boldsymbol{y}_* \mid \mathcal{D}, \boldsymbol{x}_*) \sim \mathcal{N}(\mu(\boldsymbol{x}_*), \sigma^2(\boldsymbol{x}_*))$ with prediction mean and variance,

$$\mu_{\text{full}}(\boldsymbol{x}_*) = \boldsymbol{k}_*^\intercal \boldsymbol{C}_\theta^{-1}\boldsymbol{y}, \tag{4.6}$$

$$\sigma^2_{\text{full}}(\boldsymbol{x}_*) = \sigma_f^2(k_{**} - \boldsymbol{k}_*^\intercal \boldsymbol{C}_\theta^{-1}\boldsymbol{k}_*), \tag{4.7}$$

where $\boldsymbol{k}_* = k(\boldsymbol{X}, \boldsymbol{x}_*) \in \mathbb{R}^N$ and $k_{**} = k(\boldsymbol{x}_*, \boldsymbol{x}_*) \in \mathbb{R}$.

Complexity

The time complexity of the training is $\mathcal{O}(N^3)$ for computing the inverse of the covariance matrix in (P1) and (4.5). Note that only the inverse of the covariance matrix $\boldsymbol{C}_\theta^{-1}$ is required to be computed for the training (P1) and not the logarithm of its determinant $\ln|\boldsymbol{C}_\theta|$. That is because the covariance matrix is symmetric and PD, and thus the Cholesky decomposition can be employed to compute the logarithm of the covariance matrix determinant [104, Appendix A.4]. The inverse computation of the covariance matrix is performed repeatedly in the optimization (P1) to find the hyperparameters $\hat{\boldsymbol{\theta}}$. After solving (P1) and obtaining the

Table 4.1: Time, Space, and Communication Complexity of GP Training

			FULL-GP	FACT-GP [90]	G-FACT-GP [76]
Local		Time	-	$\mathcal{O}(N^3/M^3)$	$\mathcal{O}(8(N^3/M^3))$
		Space	-	$\mathcal{O}(\xi)$	$\mathcal{O}(2\xi + 2(N^2/M^2))$
Global	GD	Space	-	$\mathcal{O}(DM + 2M)$	$\mathcal{O}(DM + 2M)$
		Communication	-	$\mathcal{O}(s^{\text{end}}(DM + 2M))$	$\mathcal{O}(s^{\text{end}}(DM + 2M))$
	Final	Time	$\mathcal{O}(N^3)$	-	-
		Space	$\mathcal{O}(N^2 + DN)$	$\mathcal{O}(N^2/M)$	$\mathcal{O}(4(N^2/M))$
		Communication	-	$\mathcal{O}(N^2/M)$	$\mathcal{O}(4(N^2/M))$

$\xi = N^2/M^2 + D(N/M)$.

hyperparameter vector $\hat{\boldsymbol{\theta}}$, we store the inverse of the covariance matrix $\boldsymbol{C}^{-1}$ and N observations, which results in $\mathcal{O}(N^2+DN)$ space complexity. For agents with limited RAM memory capacity, the space complexity may be more restrictive than the time complexity. The prediction mean (4.6) and variance (4.7) yield $\mathcal{O}(N)$ and $\mathcal{O}(N^2)$ computations respectively for matrix multiplications.

4.1.3 Centralized Scalable Gaussian Processes

Let us consider a network of M agents that can perform local computations. Each agent i collects local observations to form the local dataset $\{\mathcal{D}_i = \{\boldsymbol{X}_i, \boldsymbol{y}_i\}\}_{i=1}^M$ corresponding to N_i observations for M robots with $\sum_{i=1}^M N_i = N$. Thus, the global dataset is composed as $\mathcal{D} = \cup_{i=1}^M \mathcal{D}_i$. All local datasets have equal number of observations, i.e. $N_i = N_j = N/M$ for all $i, j \in \mathcal{V}$ with $i \neq j$. For privacy reasons, for example, we presume that the local datasets $\mathcal{D}_i$ should not be communicated to other agents. In practice, even if all agents have access to the global dataset $\mathcal{D}$, the GP computational complexity (Section 4.1.2) is prohibitive if $\mathcal{D}$ is large. Furthermore, in case we assume a centralized topology, where every entity i communicates its dataset $\mathcal{D}_i$ to a central node with significant computational and storage resources, then we face several problems. These problems comprise of: i) *security*

and *robustness*, as the central node is vulnerable to malicious attacks or even failure; ii) *traffic network congestion*, when all agents communicate their local datasets with the central entity; and iii) *privacy*, because a single entity has access to the global dataset. In addition, for certain cases (e.g., autonomous vehicles and multi-robot systems), distant agents may be subject to communication range limitations. To this end, we make the following assumptions.

Assumption 4.2. *Every agent i can communicate only with agents in its neighborhood $\mathcal{N}_i$ and the communication shall not include any data exchange.*

Assumption 4.3. *Every agent i can communicate only with agents in its neighborhood $\mathcal{N}_i$ and the communication shall include partial exchange of the local dataset $\mathcal{D}_i$.*

Remark 4.4. Assumption 4.2 prohibits the communication of any observation, while Assumption 4.3 allows the communication of a subset of the local dataset $\mathcal{D}_i$. This distinction has been made to propose different methodologies in case that partial communication of the local dataset is permitted.

Factorized Training

The factorized GP training [30, 90], termed as FACT-GP, relies on the following assumption.

Assumption 4.5. *All local sub-models $\mathcal{M}_i$ are independent.*

The independence in Assumption 4.5 is invoked to result in the approximation of the global marginal likelihood as,

$$p(\boldsymbol{y} \mid \boldsymbol{X}) \approx \prod_{i=1}^{M} p_i(\boldsymbol{y}_i \mid \boldsymbol{X}_i), \tag{4.8}$$

where $p_i(\boldsymbol{y}_i \mid \boldsymbol{X}_i) \sim \mathcal{N}(0, \boldsymbol{C}_{\theta,i})$ is the local marginal likelihood of the i-th robot with local covariance matrix $\boldsymbol{C}_{\theta,i} = \boldsymbol{K}_i + \sigma_\epsilon^2 I_{N_i}$ and $\boldsymbol{K}_i = k(\boldsymbol{X}_i, \boldsymbol{X}_i) \in \mathbb{R}^{N_i \times N_i}$. The factorized approximation (4.8) implies that the covariance matrix is approximated by a block diagonal

matrix that results in $\boldsymbol{K}^{-1} \approx \text{diag}(\boldsymbol{K}_1^{-1}, \boldsymbol{K}_2^{-1}, \dots, \boldsymbol{K}_M^{-1})$. Subsequently, the global marginal log-likelihood is approximated by $\mathcal{L} \approx \sum_{i=1}^{M} \mathcal{L}_i$ which yields,

$$\ln p(\boldsymbol{y} \mid \boldsymbol{X}) \approx \sum_{i=1}^{M} \ln p_i(\boldsymbol{y}_i \mid \boldsymbol{X}_i),$$

with local marginal log-likelihood,

$$\begin{aligned} \mathcal{L}_i &= \ln p_i(\boldsymbol{y}_i \mid \boldsymbol{X}_i) \\ &= -\frac{1}{2}\Big(\boldsymbol{y}_i^\intercal \boldsymbol{C}_{\theta,i}^{-1}\boldsymbol{y}_i + \ln|\boldsymbol{C}_{\theta,i}| + N_i \ln 2\pi\Big). \end{aligned} \tag{4.9}$$

The gradient of the global marginal log-likelihood in factorized training is computed by $\nabla_{\boldsymbol{\theta}}\mathcal{L} = \sum_{i=1}^{M} \nabla_{\boldsymbol{\theta}}\mathcal{L}_i$ [135, 136]. The minimization problem utilizes the local negative marginal log-likelihood (LNLL) function and takes the form of,

$$\begin{aligned} \text{(P2)} \quad \hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \quad & \frac{1}{2}\sum_{i=1}^{M} \boldsymbol{y}_i^\intercal \boldsymbol{C}_{\theta,i}^{-1}\boldsymbol{y}_i + \ln|\boldsymbol{C}_{\theta,i}| + N_i \ln 2\pi \\ \text{s.to} \quad & \boldsymbol{\theta_i} > \mathbf{0}_{D+2}, \quad \forall i \in \mathcal{V}, \end{aligned} \tag{4.10}$$

where $\boldsymbol{\theta}_i = \{\theta_{1,i}, \dots, \theta_{D+2,i}\}$. Similarly to (P1), constraint (4.10) imposes positivity on the agreed hyperparameters.

Remark 4.6. A common approach to relax the positivity constraint (4.4) in (P1) and (4.10) in (P2) is to employ the logarithmic transformation on the hyperparameter vector that has strictly positive domain, i.e. $\ln(\boldsymbol{\theta}) : \mathbb{R}_{>0} \to \mathbb{R}$. After convergence the inverse transformation, by using the exponential, yields the hyperparameter vector.

The computation of (4.9) for the FACT-GP training (P2) yields $\mathcal{O}(N_i^3) = \mathcal{O}(N^3/M^3)$ time complexity for each local entity to invert the local covariance matrix $\boldsymbol{C}_{\theta,i}^{-1}$. Addi-

tionally, for the storage of the local inverted covariance matrix and the local observations $\mathcal{O}(N_i^2 + DN_i) = \mathcal{O}(N^2/M^2 + D(N/M))$ space is needed. The factorized training requires also communication from every node i to the central node. The communication complexity depends on the selection of the optimization algorithm. Provided that the central node implements gradient descent [135], every node communicates the local gradient of LNLL $\nabla_{\boldsymbol{\theta}}\mathcal{L}_i$ at every iteration s. That is $\mathcal{O}(s^{\text{end}}(D+2)M) = \mathcal{O}(s^{\text{end}}(DM+2M))$ total communications from all agents to the central node, where s^{end} is the total number of iterations to reach convergence. Additionally, the central node needs to store at each iteration: i) the hyperparameter vector on the previous iteration $\{\boldsymbol{\theta}_i^{(s)}\}_{i=1}^M$ from all M nodes; and ii) their gradient of LNLL $\{\nabla_{\boldsymbol{\theta}}\mathcal{L}_i\}_{i=1}^M$, which results in $\mathcal{O}((D+2)M + (D+2)) = \mathcal{O}(DM+2M)$ space complexity. Finally, after the optimization algorithm converges, each node communicates the local inverted covariance matrix $\boldsymbol{C}_{\theta,i}^{-1}$ that yields $\mathcal{O}(MN_i^2) = \mathcal{O}(N^2/M)$ communications to the central node. All local inverted covariance matrices need to be stored in the central node to form the block diagonal approximation for GP prediction, i.e. $\mathcal{O}(N^2/M)$ space complexity. A computational complexity comparison between FULL-GP and FACT-GP is provided in Table 4.1. Since $N_i = N/M < N$, the time and space complexity of FACT-GP (P2) is significantly less than the time and space complexity of FULL-GP (P1).

Aggregated Prediction

Provided the hyperparameter vector $\hat{\boldsymbol{\theta}}$ from Problem 2 or 3, we shall employ multiple aggregation of GP experts methods to perform joint prediction with local data. These are the PoE [51], gPoE [17], BCM [126], rBCM [30], grBCM [76], and NPAE [5, 107]. The main idea is that each local entity i develops a local GP sub-model $\mathcal{M}_i$ using its local dataset $\mathcal{D}_i$. Then, the agents communicate local models to make joint predictions. The local GP sub-model $\mathcal{M}_i$ conditioned on the local dataset is $p_i(y_* \mid \mathcal{D}_i, \boldsymbol{x}_i) \sim \mathcal{GP}(\mu_i(\boldsymbol{x}_i), \sigma_i^2(\boldsymbol{x}_i))$ with

local mean and local variance,

$$\mu_i(\boldsymbol{x}_*) = \boldsymbol{k}_{*,i}^\intercal \boldsymbol{C}_{\theta,i}^{-1} \boldsymbol{y}_i, \tag{4.11}$$

$$\sigma_i^2(\boldsymbol{x}_*) = \sigma_f^2(k_{**} - \boldsymbol{k}_{*,i}^\intercal \boldsymbol{C}_{\theta,i}^{-1} \boldsymbol{k}_{*,i}), \tag{4.12}$$

where $\boldsymbol{k}_{*,i} = k(\boldsymbol{x}_*, \boldsymbol{X}_i) \in \mathbb{R}^{N_i}$.

A useful definition to study the properties of joint mean predictions for local aggregation methods is provided below.

Definition 4.7. [107] Provided N observations, an aggregate GP method with joint prediction mean $\mu_{\mathcal{A}}$, variance $\sigma^2_{\mathcal{A}}$, full GP prediction mean μ_{full}, and variance σ^2_{full} is consistent if,

$$\lim_{N\to\infty} \mu_{\text{full}}(\boldsymbol{x}_*) - \mu_{\mathcal{A}}(\boldsymbol{x}_*) \to 0, \quad \forall \boldsymbol{x}_*,$$

$$\lim_{N\to\infty} \sigma^2_{\text{full}}(\boldsymbol{x}_*) - \sigma^2_{\mathcal{A}}(\boldsymbol{x}_*) \to 0, \quad \forall \boldsymbol{x}_*,$$

where subscript $\mathcal{A}$ denotes any aggregation method for GP prediction.

Proposition 4.7 implies that as the number of observations tends to infinity, the prediction mean and variance of full GP (4.6) and the aggregated prediction mean and variance are identical.

PoE Family: After computing the local mean (4.11) and the local variance (4.12), the

joint mean and precision of both PoE and gPoE $\mathcal{M}_{\mathcal{A}} = \mathcal{M}_{\text{(g)PoE}}$ is provided by,

$$\mu_{\text{(g)PoE}}(\boldsymbol{x}_*) = \sigma^2_{\text{(g)PoE}}(\boldsymbol{x}_*) \sum_{i=1}^{M} \beta_i \sigma_i^{-2}(\boldsymbol{x}_*) \mu_i(\boldsymbol{x}_*), \tag{4.13}$$

$$\sigma^{-2}_{\text{(g)PoE}}(\boldsymbol{x}_*) = \sum_{i=1}^{M} \beta_i \sigma_i^{-2}(\boldsymbol{x}_*), \tag{4.14}$$

where $\beta_i = 1$ for PoE, and $\beta_i = 1/M$ for gPoE. The original gPoE [17] considers weight β_i to be the difference in differential entropy, but this approach limits the computational graph to have a single layer. To allow multiple layer GP aggregation, an average weight β_i is proposed in [30]. Since for certain decentralized networks multiple layer GP aggregation is required, we find the average weight more suitable.

Proposition 4.8. *[76, Proposition 1] For a disjoint partition of local datasets $\mathcal{D}_i$, the constant weight of PoE results in overconfident joint variance as the number of observations N tends to infinity, i.e. $\lim_{N\to\infty} \sigma^2_{PoE}(\boldsymbol{x}_*) \to 0$. The average weight of gPoE produces a conservative, yet finite joint variance as the number of observations N tends to infinity, i.e. $\sigma^2_{full} < \lim_{N\to\infty} \sigma^2_{gPoE}(\boldsymbol{x}_*) < \sigma^2_{**}$, where σ^2_{full} is the target variance of a full GP and σ^2_{**} the prior variance.*

Remark 4.9. It is empirically observed that for a disjoint partition of local datasets $\mathcal{D}_i$, both PoE and gPoE produce inconsistent mean predictions [76]. Specifically, as the number of observations tends to infinity both methods recover the prior mean μ_{**}, i.e. $\lim_{N\to\infty} \mu_{\text{(g)PoE}}(\boldsymbol{x}_*) \to \mu_{**}$ for all $\boldsymbol{x}_*$.

Proposition 4.10. *The PoE and gPoE make identical mean predictions (4.13).*

Proof. The proof is provided in Appendix B.1.

The local time computational complexity of both PoE and gPoE is governed by the local

Table 4.2: Time, Space, and Communication Complexity for Centralized GP Aggregated Prediction

		FULL-GP	(g)PoE & BCM	rBCM [30]	grBCM [76]	NPAE [107]
Local	Time	-	$\mathcal{O}(\zeta)$	$\mathcal{O}(\zeta)$	$\mathcal{O}((5+N/M)\zeta)$	$\mathcal{O}(N\zeta)$
	Space	-	$\mathcal{O}(\xi)$	$\mathcal{O}(\xi)$	$\mathcal{O}(2\xi+2(N^2/M^2))$	$\mathcal{O}(\xi+DN)$
Global	Time	$\mathcal{O}(N^2)$	$\mathcal{O}(M)$	$\mathcal{O}(M)$	$\mathcal{O}(M)$	$\mathcal{O}(M^3)$
	Space	$\mathcal{O}(N^2+DN)$	$\mathcal{O}(2M)$	$\mathcal{O}(3M)$	$\mathcal{O}(5M)$	$\mathcal{O}(M^2)$
	Comm	-	$\mathcal{O}(2M)$	$\mathcal{O}(3M)$	$\mathcal{O}(5M)$	$\mathcal{O}(M^2)$

$\zeta = N^2/M^2$, $\xi = N^2/M^2 + D(N/M)$.

variance (4.12), that is $\mathcal{O}(N_i^2) = \mathcal{O}(N^2/M^2)$ for the multiplication of the quadratic term. Provided that the number of local observations is less than the observations from all local entities $N_i < N$, the PoE and gPoE alleviates the computations compared to the full GP $\mathcal{O}(N^2)$. The space complexity requires $\mathcal{O}(N_i^2 + DN_i) = \mathcal{O}(N^2/M^2 + D(N/M))$ capacity to store the inverse of the local inverted covariance matrix $\boldsymbol{C}_{\theta,i}^{-1}$ and the vector of local observations $\boldsymbol{y}_i$. Thus, the space requirement is relaxed compared to full GP that occupies $\mathcal{O}(N^2 + DN)$ memory. The total communication complexity from all entities to the central node is $\mathcal{O}(2M)$ to transmit all local mean μ_i and local variance σ_i^2 values. A comparison of PoE and gPoE with other aggregation methods is presented in Table 4.2.

BCM Family: The BCM, rBCM, and grBCM make an additional assumption other than Assumption 4.5.

Assumption 4.11. *The dataset of every agent i is conditionally independent from any other dataset of agent $j \neq i$ given the posterior distribution f_*, i.e. $\mathcal{D}_i \perp\!\!\!\perp \mathcal{D}_j \mid f_*$.*

After computing the local mean (4.11) and the local variance (4.12), the joint mean and

precision of the BCM and rBCM $\mathcal{M}_{\mathcal{A}} = \mathcal{M}_{(\mathrm{r})\mathrm{BCM}}$ is provided by,

$$\mu_{(\mathrm{r})\mathrm{BCM}}(\boldsymbol{x}_*) = \sigma^2_{(\mathrm{r})\mathrm{BCM}}(\boldsymbol{x}_*) \sum_{i=1}^{M} \beta_i \sigma_i^{-2}(\boldsymbol{x}_*)\mu_i(\boldsymbol{x}_*), \tag{4.15}$$

$$\sigma^2_{(\mathrm{r})\mathrm{BCM}}(\boldsymbol{x}_*) = \sum_{i=1}^{M} \beta_i \sigma_i^{-2}(\boldsymbol{x}_*) + (1 - \sum_{i=1}^{M} \beta_i)\sigma_{**}^{-2}, \tag{4.16}$$

where $\beta_i = 1$ for BCM, and $\beta_i = 0.5[\ln \sigma^2_{**} - \ln \sigma_i^2(\boldsymbol{x}_*)]$ for rBCM. For rBCM β_i describes the difference in the differential entropy between the prior and the posterior distribution.

Proposition 4.12. *[76, Proposition 1] For a disjoint partition of local datasets $\mathcal{D}_i$, the BCM and rBCM results in overconfident joint variance as the number of observations N tends to infinity, i.e.* $\lim_{N\to\infty} \sigma^2_{(r)BCM}(\boldsymbol{x}_*) \to 0$.

Remark 4.13. It is empirically observed that for a disjoint partition of local datasets $\mathcal{D}_i$, both BCM and rBCM produce inconsistent mean predictions [76]. However, the joint prediction mean of BCM and rBCM converges to the prior mean slower than the PoE and gPoE.

The time and space complexity of BCM is identical to PoE and gPoE. In addition, BCM requires similar communications with the PoE family. However, the rBCM entails $\mathcal{O}(3M)$ communication complexity to exchange the local mean μ_i, the local variance σ_i^2, and the difference in the differential entropy between the prior and posterior distribution β_i. Note that β_i in rBCM can be computed by the central node and recovers the communication complexity of PoE and gPoE. Yet, we prefer to express β_i as part of the communication exchange, because in the ensuing discussion the central node is removed. A comparison of BCM and rBCM with other aggregation methods is illustrated in Table 4.2.

grBCM: The main idea of grBCM is to equip every agent with a new dataset that has global information on the underlying latent function to ensure consistency (Definition 4.7).

Every agent i selects randomly without replacement N_i/M data from its local dataset $\mathcal{D}_i$ to form the *local sample dataset* $\mathcal{D}_{-i} \in \mathbb{R}^{N_i/M} \subset \mathcal{D}_i$. Then, the local sample datasets are communicated to every other agent (Assumption 4.3) to compose the *communication dataset* $\mathcal{D}_\mathrm{c} = \{\mathcal{D}_{\text{-i}}\}_{i=1}^M = \{\boldsymbol{X}_\mathrm{c}, \boldsymbol{y}_\mathrm{c}\}$. Next, every agent i fuses the communication dataset $\mathcal{D}_\mathrm{c} \in \mathbb{R}^{N_i}$ with its local dataset $\mathcal{D}_i$ to form the *local augmented dataset* $\mathcal{D}_{+i} = \mathcal{D}_i \cup \mathcal{D}_\mathrm{c} \in \mathbb{R}^{2N_i}$. The local augmented dataset $\mathcal{D}_{+i}$ is a new dataset for every agent i that includes the local dataset $\mathcal{D}_i$ and the communication dataset $\mathcal{D}_\mathrm{c}$, providing a global perspective. Note that in [76], the authors of grBCM suggest to select randomly the communication dataset $\mathcal{D}_\mathrm{c}$ from the full dataset $\mathcal{D}$. In this paper, we consider a slight variation that does not violate any result towards a network implementation. Thus, the communication dataset $\mathcal{D}_\mathrm{c}$ is selected by the local datasets $\mathcal{D}_i$ and then fused through information exchange.

The agents shall use the local augmented dataset $\mathcal{D}_{+i}$ to compute the augmented local mean μ_{+i} (4.11) and the augmented local variance σ^2_{+i} (4.12). In addition, grBCM requires the computation of the communication local mean μ_c (4.11) and the communication local variance σ^2_c (4.12) using exclusively the communication dataset $\mathcal{D}_\mathrm{c}$. The joint mean and precision of grBCM $\mathcal{M}_\mathcal{A} = \mathcal{M}_\mathrm{grBCM}$ yield,

$$\mu_\mathrm{grBCM}(\boldsymbol{x}_*) = \sigma^2_\mathrm{grBCM}(\boldsymbol{x}_*)\Big[\sum_{i=1}^M \beta_i \sigma^{-2}_{+i}(\boldsymbol{x}_*)\mu_{+i}(\boldsymbol{x}_*) - \Big(\sum_{i=1}^M \beta_i - 1\Big)\sigma_\mathrm{c}^{-2}(\boldsymbol{x}_*)\mu_\mathrm{c}(\boldsymbol{x}_*)\Big], \tag{4.17}$$

$$\sigma^{-2}_\mathrm{grBCM}(\boldsymbol{x}_*) = \sum_{i=1}^M \beta_i \sigma^{-2}_{+i}(\boldsymbol{x}_*) + (1 - \sum_{i=1}^M \beta_i)\sigma_\mathrm{c}^{-2}(\boldsymbol{x}_*), \tag{4.18}$$

where $\beta_1 = 1$ and $\beta_i = 1/2[\ln \sigma^2_\mathrm{c}(\boldsymbol{x}_*) - \ln \sigma^2_{+i}(\boldsymbol{x}_*)]$ for $i > 2$.

Proposition 4.14. *[76, Proposition 3] For any collection of aggregated prediction mean values $\mu_1(\boldsymbol{x}_*), \ldots, \mu_M(\boldsymbol{x}_*)$ the grBCM is consistent.*

The local time complexity for grBCM includes the computation of the augmented local variance σ^2_{+i}, the local communication variance σ^2_{c}, and the inversion of the communication dataset covariance matrix $\boldsymbol{K}_{\mathrm{c}} = k(\boldsymbol{X}_{\mathrm{c}}, \boldsymbol{X}_{\mathrm{c}})$. That is $\mathcal{O}((2N_i)^2 + N_i^2 + N_i^3) = \mathcal{O}(N^2/M^2(5 + N/M))$. Then, the inverse of the local augmented covariance matrix $\boldsymbol{C}^{-1}_{\theta,+i}$ and the local augmented dataset $\mathcal{D}_{+i}$ occupy $\mathcal{O}((2N_i)^2 + D(2N_i)) = \mathcal{O}(2(N^2/M^2 + DN/M) + 2N^2/M^2)$ space. The total communications from all entities to the central node is $\mathcal{O}(5M)$ to transmit all local augmented means μ_{+i}, local augmented variances σ^2_{+i}, local communication means μ_{c}, local communication variances σ^2_{c}, and all differences in the differential entropy β_i to the central node. A comparison of grBCM with other aggregation methods is shown in Table 4.2.

The local augmented dataset $\mathcal{D}_{+i}$ is used in factorized GP training (Section 4.1.3) to relax the block diagonal matrix approximation induced by Assumption 4.5, and produce better estimates of the hyperparameter vector [76]. The time, space, and communication complexity of the generalized factorized GP (g-FACT-GP) training is more demanding than the FACT-GP training, yet remains more affordable than the FULL-GP training. A comparison of all centralized GP training methods is presented in Table 4.1.

NPAE: The main idea of NPAE is to use covariance between sub-models $\mathcal{M}_i$ to ensure consistency. The local computations of NPAE for agent i include: i) local prediction mean $\mu_i(\boldsymbol{x}_*) \in \mathbb{R}$ (4.11); ii) i-th entry of the cross-covariance vector $[\boldsymbol{k}_A(\boldsymbol{x}_*)]_i \in \mathbb{R}$; and iii) i-th row of the covariance $\mathrm{row}_i\{\boldsymbol{C}_{\theta,A}(\boldsymbol{x}_*)\} \in \mathbb{R}^M$. Thus, NPAE requires the local computation of two additional quantities other than (4.11). These are the cross-covariance and the covariance for each agent i,

$$[\boldsymbol{k}_A(\boldsymbol{X}_i, \boldsymbol{x}_*)]_i = \boldsymbol{k}^{\intercal}_{i,*}\boldsymbol{C}^{-1}_{\theta,i}\boldsymbol{k}_{i,*}, \tag{4.19}$$

$$[\mathrm{row}_i\{\boldsymbol{C}_{\theta,A}(\boldsymbol{X}_i, \boldsymbol{X}_j, \boldsymbol{x}_*)\}]_j = \boldsymbol{k}^{\intercal}_{i,*}\boldsymbol{C}^{-1}_{\theta,i}\boldsymbol{C}_{\theta,ij}\boldsymbol{C}^{-1}_{\theta,j}\boldsymbol{k}_{j,*}, \tag{4.20}$$

where $\boldsymbol{C}_{\theta,ij} = k(\boldsymbol{X}_i, \boldsymbol{X}_j) + \sigma_\epsilon^2 I_{N_i} \in \mathbb{R}^{N_i \times N_i}$, $\boldsymbol{k}_{i,*} = (\boldsymbol{X}_i, \boldsymbol{x}_*) \in \mathbb{R}^{N_i}$, and $\boldsymbol{k}_{j,*} = (\boldsymbol{X}_j, \boldsymbol{x}_*) \in \mathbb{R}^{N_i}$ for all $j \neq i$. The next step is to aggregate the local sub-models and obtain the joint prediction mean and variance,

$$\mu_{\text{NPAE}}(\boldsymbol{x}_*) = \boldsymbol{k}_A^\intercal \boldsymbol{C}_{\theta,A}^{-1} \boldsymbol{\mu}, \tag{4.21}$$

$$\sigma_{\text{NPAE}}^2(\boldsymbol{x}_*) = \sigma_f^2 (k_{**} - \boldsymbol{k}_A^\intercal \boldsymbol{C}_{\theta,A}^{-1} \boldsymbol{k}_A), \tag{4.22}$$

where $\boldsymbol{C}_{\theta,A} = \{\text{row}_i\{\boldsymbol{C}_{\theta,A}\}\}_{i=1}^M \in \mathbb{R}^{M \times M}$, $\boldsymbol{k}_A = \{\boldsymbol{k}_A(\boldsymbol{X}_i, \boldsymbol{x}_*)\}_{i=1}^M \in \mathbb{R}^M$, and $\boldsymbol{\mu} = \{\mu_i\}_{i=1}^M \in \mathbb{R}^M$.

Proposition 4.15. *[5, Proposition 2] For any collection of aggregated prediction mean values $\mu_1(\boldsymbol{x}_*), \ldots, \mu_M(\boldsymbol{x}_*)$ the NPAE is consistent.*

The local time complexity for NPAE is governed by the computation of all local inverted covariance matrices for every other agent $\boldsymbol{C}_{\theta,j}^{-1}$, $j \neq i$ (4.19) which yields $\mathcal{O}((M-1)N_i^3) = \mathcal{O}(N^3/M^2)$ computations. Additionally, the aggregated covariance $\boldsymbol{C}_{\theta,A}$ needs to be inverted on the central node that entails $\mathcal{O}(M^3)$ computations. The local memory footprint is $\mathcal{O}(N_i^2 + DN_i + MDN_i) = \mathcal{O}(N^2/M^2 + D(N/M) + DN)$, including the local inverted covariance matrix $\boldsymbol{C}_{\theta,i}^{-1}$, the local dataset $\mathcal{D}_i$, and the inputs of all other datasets $\boldsymbol{X}_j$ for all $j \neq i$. The total communication complexity yields $\mathcal{O}((M+1)M) = \mathcal{O}(M^2)$ governed by the transmission of the row aggregated covariance $\text{row}_i\{\boldsymbol{C}_{\theta,A}\}$, for all $i \in \mathcal{V}$. A major disadvantage of NPAE is the high global time complexity, yet in the ensuing discussion we remove this expensive computation by using decentralized iterative techniques. A comparison of NPAE with other aggregation methods is provided in Table 4.2.

4.1.4 Problem Definition

In this section we define the problems we seek to address.

Problem 2. Under Assumption 4.1, 4.2, and 4.5, solve the optimization problem (P2) to estimate the hyperparmeters $\hat{\boldsymbol{\theta}}$ of the GP for a decentralized network topology.

Problem 3. Under Assumption 4.1, 4.3, and 4.5, solve the optimization problem (P2) to estimate the hyperparmeters $\hat{\boldsymbol{\theta}}$ of the GP for a decentralized network topology.

The difference between Problem 2 and 3 is that the latter allows partial data exchange while the former prohibits any data exchange between agents. Both problems assume a strongly connected network of agents (Assumption 4.1) and make the independence approximation assumption between local datasets (Assumption 4.5). Recent advancements in distributed GP hyperparameter training [135, 136] have addressed the centralized version of Problem 2. The focus of this paper is on decentralized networks without requiring a central coordinator with massive computational and storage capabilities. The decentralized setup is imposed by Assumption 4.2 and 4.3.

Problem 4. Let Assumption 4.1, 4.2, 4.5, and 4.11 hold, decentralize the computation of the PoE, gPoE, BCM, rBCM, and NPAE using expertise from all agents. In addition, replace Assumption 4.2 with 4.3 and decentralize the computation of grBCM.

Problem 5. Let Assumption 4.1, 4.2, 4.5, and 4.11 hold, decentralize the computation of the PoE, gPoE, BCM, rBCM, and NPAE using expertise from statistically correlated agents. In addition, replace Assumption 4.2 with 4.3 and decentralize the grBCM.

The difference between Problem 4 and 5 is that the latter involves the joint prediction of only statistically correlated agents. That is because we seek to reduce the communication from distant entities with insignificant statistic correlation to the aggregation.

Table 4.3: Time, Space, and Communication Complexity of Centralized Factorized GP Training with ADMM-based Methods

			c-GP [136]	apx-GP [135]	gapx-GP (proposed)
Local		Time	$\mathcal{O}(s^{\text{end}}_{\text{nest}}(N^3/M^3))$	$\mathcal{O}(N^3/M^3)$	$\mathcal{O}(8(N^3/M^3))$
		Space	$\mathcal{O}(\xi)$	$\mathcal{O}(\xi)$	$\mathcal{O}(2\xi + 2(N^2/M^2))$
Global	ADMM	Comm	$\mathcal{O}(s^{\text{end}}_{\text{c-GP}}M(D+2))$	$\mathcal{O}(s^{\text{end}}_{\text{apx-GP}}M(D+2))$	$\mathcal{O}(s^{\text{end}}_{\text{gapx-GP}}M(D+2))$
	Final	Comm	$\mathcal{O}(N^2/M)$	$\mathcal{O}(N^2/M)$	$\mathcal{O}(4(N^2/M))$

$s^{\text{end}}_{\text{c-GP}} < s^{\text{end}}_{\text{apx-GP}} \approx s^{\text{end}}_{\text{gapx-GP}}$, $\xi = N^2/M^2 + D(N/M)$.

4.2 Centralized GP Training

In this section, we discuss existing centralized methods and propose a centralized technique to address the factorized GP training problem (P2) based on the alternating direction method of multipliers (ADMM) [12].

The following Assumption is required for first-order approximation methods.

Assumption 4.16. *A function $f : \mathbb{R}^{\mathbb{N}} \to \mathbb{R}$ is Lipschitz continuous with positive parameter $L > 0$ if it satisfies,*

$$\|\nabla f(\boldsymbol{x}) - \nabla f(\boldsymbol{y})\|_2 \leq L\|\boldsymbol{x} - \boldsymbol{y}\|_2, \quad \forall \boldsymbol{x}, \boldsymbol{y}. \tag{4.23}$$

4.2.1 Existing Centralized GP Training Methods

To address the centralized factorized GP training problem (P2) an exact consensus ADMM (c-ADMM) and an inexact proximal consensus ADMM (px-ADMM) have been used in [135, 136]. Using the relaxation of the positivity constraint in Remark 4.6, (P2) can be expressed

as,

$$\text{(P3)} \quad \hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \quad \frac{1}{2}\sum_{i=1}^{M} \boldsymbol{y}_i^{\intercal}\boldsymbol{C}_{\theta,i}^{-1}\boldsymbol{y}_i + \ln|\boldsymbol{C}_{\theta,i}| + N_i \ln 2\pi$$
$$\text{s.to} \quad \boldsymbol{\theta}_i = \boldsymbol{z}, \quad \forall i \in \mathcal{V}, \tag{4.24}$$

where $\boldsymbol{\theta}_i = \{\theta_{1,i}, \ldots, \theta_{D+2,i}\}$ is the local vector of hyperparameters and $\boldsymbol{z} \in \mathbb{R}^{D+2}$ is an auxiliary variable. In other words, constraint (4.24) implies that every agent i is allowed to have its own opinion for the hyperparameters $\boldsymbol{\theta}_i$, yet at the end of the optimization all agents must agree on the global vector value $\boldsymbol{z}$. Recognize that (P3) has the same problem formulation with the c-ADMM problem. After formulating the augmented Lagrangian, the c-GP iterative scheme [12] takes the form,

$$\boldsymbol{z}^{(s+1)} = \frac{1}{M}\sum_{i=1}^{M}\left(\boldsymbol{\theta}_i^{(s)} + \frac{1}{\rho}\boldsymbol{\psi}_i^{(s)}\right), \tag{4.25a}$$
$$\boldsymbol{\theta}_i^{(s+1)} = \arg\min_{\boldsymbol{\theta}_i}\left\{\mathcal{L}_i(\boldsymbol{\theta}_i) + (\boldsymbol{\psi}_i^{(s)})^{\intercal}(\boldsymbol{\theta}_i - \boldsymbol{z}^{(s+1)}) + \frac{\rho}{2}\|\boldsymbol{\theta}_i - \boldsymbol{z}^{(s+1)}\|_2^2\right\}, \tag{4.25b}$$
$$\boldsymbol{\psi}_i^{(s+1)} = \boldsymbol{\psi}_i^{(s)} + \rho(\boldsymbol{\theta}_i^{(s+1)} - \boldsymbol{z}^{(s+1)}), \tag{4.25c}$$

where $\boldsymbol{\psi}_i \in \mathbb{R}^{D+2}$ is the vector of dual variables of the i-th node, $s \in \mathbb{Z}_{\geq 0}$ is the iteration number, and $\rho > 0$ is the penalty constant term of the augmented Lagrangian. The steps of c-GP are the following: i) all agents transmit their $\boldsymbol{\theta}_i^{(s)}$ to the central node; ii) the central node updates the global hyperparameter vector $\boldsymbol{z}^{(s+1)}$ (4.25a); iii) the central node scatters the updated $\boldsymbol{z}^{(s+1)}$ vector; iv) every agent i solves locally the nested optimization problem (4.25b) to find the local hyperparameter vector $\boldsymbol{\theta}_i^{(s+1)}$; and v) every agent i updates the local dual vector $\boldsymbol{\psi}_i^{(s+1)}$ (4.25c).

Let $s_{\text{nest}}^{\text{end}}$ be the number of iterations required from the nested optimization problem (4.25b) to converge. The computational complexity of c-GP is cubic in the number of local observations

$\mathcal{O}(s_{\text{nest}}^{\text{end}} N_i^3) = \mathcal{O}(s_{\text{nest}}^{\text{end}}(N^3/M^3))$. More specifically, the nested optimization problem (4.25b) requires the evaluation of the local log-likelihood $\mathcal{L}_i(\boldsymbol{\theta}_i)$ at every internal iteration s_{nest} which entails cubic computations to invert the local covariance matrix $\boldsymbol{C}_{\theta,i}^{-1}$ (4.9). The communication complexity to transmit all local hyperparameter vectors yields $\mathcal{O}(s^{\text{end}} M(D+2))$. After convergence, every agent i transmits the local inverted covariance matrix $\boldsymbol{C}_{\theta,i}^{-1}$ which yields $\mathcal{O}(MN_i^2) = \mathcal{O}(N^2/M)$ communications. Every agent i occupies $\mathcal{O}(N_i^2 + 3(D+2) + D(N/M))) = \mathcal{O}(N^2/M^2 + DN/M)$ memory to store the local inverted covariance matrix $\boldsymbol{C}_{\theta,i}^{-1}$, the three quantities of c-GP at the previous iteration $\boldsymbol{\theta}_i^{(s)}$, $\boldsymbol{z}^{(s)}$, $\boldsymbol{\psi}_i^{(s)}$, and the local dataset $\mathcal{D}_i$.

The major disadvantage of c-GP is the time complexity of the nested optimization problem (4.25b). To address this issue, the authors in [135] employed the inexact px-ADMM [53] and derived an analytical solution for the case of centralized factorized GP training to form the analytical px-ADMM-GP (apx-GP). Note that apx-GP employs a first-order approximation (linearization) on the local log-likelihood $\mathcal{L}_i$ around $\boldsymbol{z}^{(s+1)}$ which yields,

$$\mathcal{L}_i(\boldsymbol{\theta}_i) \approx \nabla_{\boldsymbol{\theta}}^{\intercal} \mathcal{L}_i(\boldsymbol{z}^{(s+1)})(\boldsymbol{\theta}_i - \boldsymbol{z}^{(s+1)}) + \frac{L_i}{2} \|\boldsymbol{\theta}_i - \boldsymbol{z}^{(s+1)}\|_2^2, \tag{4.26}$$

where $L_i > 0$ is a positive Lipschitz constant that satisfies Assumption 4.16 of the local log-likelihood function $\mathcal{L}_i$ for all $i \in \mathcal{V}$. The apx-GP iteration steps are given by,

$$\boldsymbol{z}^{(s+1)} = \frac{1}{M} \sum_{i=1}^{M} \left(\boldsymbol{\theta}_i^{(s)} + \frac{1}{\rho} \boldsymbol{\psi}_i^{(s)} \right), \tag{4.27a}$$

$$\boldsymbol{\theta}_i^{(s+1)} = \boldsymbol{z}^{(s+1)} - \frac{1}{\rho + L_i} \left(\nabla_{\boldsymbol{\theta}} \mathcal{L}_i(\boldsymbol{z}^{(s+1)}) + \boldsymbol{\psi}_i^{(s)} \right) \tag{4.27b}$$

$$\boldsymbol{\psi}_i^{(s+1)} = \boldsymbol{\psi}_i^{(s)} + \rho(\boldsymbol{\theta}_i^{(s+1)} - \boldsymbol{z}^{(s+1)}), \tag{4.27c}$$

where the gradient of the local log-likelihood $\nabla_{\boldsymbol{\theta}} \mathcal{L}_i$ has similar structure to the the gradient

of the log-likelihood (4.5). The only difference on the workflow of apx-GP and c-GP is that step iv) is computed analytically (4.27b), while the former incorporates a nested optimization problem (4.25b) at every ADMM-iteration.

The space and communication complexity of apx-GP is identical to c-GP. However, c-GP converges faster than apx-GP, i.e. $s^{\text{end}}_{\text{c-GP}} < s^{\text{end}}_{\text{apx-GP}}$. The time complexity of apx-GP entails $\mathcal{O}(N_i^3) = \mathcal{O}(N^3/M^3)$ computations, significantly reduced from $\mathcal{O}(s^{\text{end}}_{\text{nest}} N^3/M^3)$ of c-GP. In other words, there is no nested optimization problem in apx-GP (4.27) and thus requires just one inversion of the local covariance matrix $\boldsymbol{C}_{\theta,i}^{-1}$ per ADMM-iteration instead of $s^{\text{end}}_{\text{nest}}$ inversions per ADMM-iteration in c-GP. Both c-GP and apx-GP inherit the convergence properties of c-ADMM [12] and px-ADMM [53] which result in much faster convergence than gradient descent.

A disadvantage of both centralized methods (4.25) and (4.27) is that they are based on factorized GP training and thus they inherit poor approximation capabilities when the number of nodes increases. More specifically, for a bounded space of interest, Assumption 4.5 is violated as we increase the number of sub-models $\mathcal{M}_i$.

4.2.2 Proposed Centralized GP Training

The first method we propose is a centralized factorized GP training technique that extends apx-GP with a local augmented datatset $\mathcal{D}_{+i}$ for all $i \in \mathcal{V}$. The goal is to limit the approximation error of factorized GP training inherited by Assumption 4.5 at the cost of allowing partial local data exchange (Assumption 4.3). A larger dataset entails more computations, thus we build on the computationally affordable apx-GP method. We term our methodology as generalized apx-GP (gapx-GP).

Let the communication dataset to be formed as discussed in Section 4.1.3-grBCM. Then,

Algorithm 2 GAPX-GP

Input: $\mathcal{D}_i(\boldsymbol{X}_i, \boldsymbol{y}_i)$, $k(\cdot,\cdot)$, ρ, L_i, $\mathcal{N}_i$, $\mathcal{V}$, TOL_{ADMM}
Output: $\hat{\boldsymbol{\theta}}$, $\boldsymbol{C}_\theta^{-1}$, $\mathcal{D}_{+i}$

for each $i \in \mathcal{V}$ **do** ▷ Local Sample Dataset
  $\mathcal{D}_{\text{c},i} \leftarrow \texttt{Sample}(\mathcal{D}_i)$
  communicate $\mathcal{D}_{\text{c},i}$ to central node
end for
scatter $\mathcal{D}_\text{c} = \{\mathcal{D}_{\text{c},i}\}_{i=1}^M$ from central node to every agent
for each $i \in \mathcal{V}$ **do** ▷ Local Augmented Dataset
  $\mathcal{D}_{+i} \leftarrow \mathcal{D}_i \cup \mathcal{D}_\text{c}$
end for
repeat ▷ ADMM Optimization
  communicate $\boldsymbol{\theta}_i^{(s)}$ to central node
  $\boldsymbol{z}^{(s+1)} \leftarrow \texttt{primal\_2}(\boldsymbol{\theta}_i^{(s)}, \boldsymbol{\psi}_i^{(s)}, \text{card}(\mathcal{V}))$ (4.27a)
  scatter $\boldsymbol{z}^{(s+1)}$ from central node to every agent
  for each $i \in \mathcal{V}$ **do**
    $\boldsymbol{\theta}_i^{(s+1)} \leftarrow \texttt{primal\_1}(\boldsymbol{\theta}_i^{(s)}, \boldsymbol{z}^{(s+1)}, \boldsymbol{\psi}_i^{(s)}, \rho, L_i, \mathcal{D}_{+i})$ (4.27b)
    $\boldsymbol{\psi}_i^{(s+1)} \leftarrow \texttt{dual}(\boldsymbol{\theta}_i^{(s+1)}, \boldsymbol{z}^{(s+1)}, \boldsymbol{\psi}_i^{(s)}, \rho)$ (4.27c)
  end for
until $\|\boldsymbol{\theta}_i^{(s+1)} - \boldsymbol{z}^{(s+1)}\|_2 < \text{TOL}_{\text{ADMM}}$, for all $i \in \mathcal{V}$
for each $i \in \mathcal{V}$ **do** ▷ Local Augmented Covariance Inversion
  $\hat{\boldsymbol{\theta}} \leftarrow \boldsymbol{\theta}_i^{\text{end}}$
  $\boldsymbol{C}_{\theta,+i}^{-1} \leftarrow \texttt{invert}(k, \boldsymbol{X}_{+i}, \hat{\boldsymbol{\theta}})$
  communicate $\boldsymbol{C}_{\theta,+i}^{-1}$ to central node
end for
$\boldsymbol{C}_\theta^{-1} \leftarrow \texttt{diag}(\boldsymbol{C}_{\theta,+1}^{-1}, \boldsymbol{C}_{\theta,+2}^{-1}, \ldots, \boldsymbol{C}_{\theta,+M}^{-1})$ ▷ Block Diagonal
Return $\hat{\boldsymbol{\theta}}$, $\boldsymbol{C}_\theta^{-1}$, $\mathcal{D}_{+i}$

every agent i has access to a local augmented dataset which is the union of the corresponding local dataset and the communication dataset $\mathcal{D}_{+i} = \mathcal{D}_i \cup \mathcal{D}_\text{c} \in \mathbb{R}^{2N_i}$. Next, we implement the apx-GP (4.27), but now every agent is equipped with the local augmented dataset $\mathcal{D}_{+i}$. The implementation details are provided in Algorithm 2.

The local time complexity of gapx-GP yields $\mathcal{O}((2N_i)^3) = \mathcal{O}(8(N^3/M^3))$ computations to invert the local augmented covariance matrix $\boldsymbol{C}_{\theta,+i} = \boldsymbol{K}_{+i} + \sigma_\epsilon^2 I_{2N_i} \in \mathbb{R}^{2N_i \times 2N_i}$.The total communication overhead is the same with c-GP and apx-GP. After convergence, each agent i communicates the local augmented covariance matrix $\boldsymbol{C}_{\theta,+i}^{-1}$ that entails $\mathcal{O}(M(2N_i)^2) =$

$\mathcal{O}(4(N^2/M))$ communications. The space complexity of every agent i yields $\mathcal{O}((2N_i)^2 + 3(D+2) + D(2N_i)) = \mathcal{O}(4(N^2/M^2) + 2D(N/M))$ to store the local augmented covariance matrix, the optimization variables at the previous iteration, and the local augmented dataset.

In Table 4.3, we list the time, space, and communication complexity for all centralized factorized GP training methods based on ADMM. The proposed method is more demanding in space and communication than c-GP and in all complexity aspects than apx-GP. In terms of time complexity, gapx-GP is more affordable than c-GP, because the nested optimization of the latter (4.25b) takes on average more than eight iterations to converge, i.e., $s^{\text{end}}_{\text{nest}} > 8$. The proposed method supports Assumption 4.5, and thus we expect to produce more accurate hyperparameters.

Proposition 4.17. *Let Assumption 4.5 and 4.18 hold for the local sub-model $\mathcal{M}_i$, then the gapx-GP converges, i.e., $\lim_{s\to\infty}\|\boldsymbol{\theta}_i^{(s)} - \boldsymbol{z}^{(s)}\|_2 = 0$ for all $i \in \mathcal{V}$, to a stationary solution $(\boldsymbol{\theta}_i^\star, \boldsymbol{z}^\star, \boldsymbol{\psi}_i^\star)$ of (P3).*

Proof. The proof is direct consequence of [53, Theorem 2.10].

4.3 Proposed Decentralized GP Training

In this section, we propose solutions for Problem 2 and 3 based on the *edge formulation* of ADMM [110] that yields parallel updates and decentralizes the factorized GP training. Let

Assumption 4.1 hold, then (P3) can be expressed as,

$$\text{(P4)} \quad \hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \quad \frac{1}{2}\sum_{i=1}^{M} \boldsymbol{y}_i^\intercal \boldsymbol{C}_{\theta,i}^{-1}\boldsymbol{y}_i + \ln|\boldsymbol{C}_{\theta,i}| + N_i \ln 2\pi$$

$$\text{s.to} \quad \boldsymbol{\theta}_i = \boldsymbol{\tau}_{ij}, \quad \forall i \in \mathcal{V},\, j \in \mathcal{N}_i, \tag{4.28}$$

$$\boldsymbol{\theta}_j = \boldsymbol{\tau}_{ij}, \quad \forall i \in \mathcal{V},\, j \in \mathcal{N}_i, \tag{4.29}$$

where $\boldsymbol{\tau}_{ij}$ are auxiliary variables. Constraints (4.28) and (4.29) imply that every agent i is allowed to have its own opinion for the hyperparameters $\boldsymbol{\theta}_i$, yet at the end of the optimization all agents in the neighborhood $\mathcal{N}_i$ must agree on the neighborhood values $\boldsymbol{\tau}_{ij}$. The edge formulation requires each node i to store and update variables for all of its neighbors $\mathcal{N}_i$. Conversely, one can employ the *node formulation* that relaxes the storage capacity, as each agent i is required to store and update variables of itself [80].

Let us introduce an additional Assumption to study the convergence properties of the proposed methods.

Assumption 4.18. *A function $f : \mathbb{R}^{\mathbb{N}} \to \mathbb{R}$ is strongly convex with positive parameter $m > 0$ if it satisfies,*

$$(\nabla f(\boldsymbol{x}) - \nabla f(\boldsymbol{y}))^\intercal (\boldsymbol{x} - \boldsymbol{y}) \geq m\|\boldsymbol{x} - \boldsymbol{y}\|_2^2, \quad \forall \boldsymbol{x}, \boldsymbol{y}. \tag{4.30}$$

4.3.1 DEC-c-GP

This method is based on the decentralized consensus ADMM [82]. After rendering the augmented Lagrangian for (P4) we obtain the decentralized consensus ADMM iterative

Algorithm 3 DEC-c-GP

Input: $\mathcal{D}_i(\boldsymbol{X}_i, \boldsymbol{y}_i)$, $k(\cdot,\cdot)$, ρ, $\mathcal{N}_i$, $s^{\text{end}}_{\text{DEC-c-GP}}$
Output: $\hat{\boldsymbol{\theta}}$, $\boldsymbol{C}^{-1}_{\theta,i}$

initialize $\boldsymbol{p}_i^{(0)} = \mathbf{0}$
for $s = 1$ to $s^{\text{end}}_{\text{DEC-c-GP}}$ **do** ▷ ADMM Optimization
  for each $i \in \mathcal{V}$ **do**
    communicate $\boldsymbol{\theta}_i^{(s)}$ to neighbors $\mathcal{N}_i$
    $\boldsymbol{p}_i^{(s+1)} \leftarrow \texttt{duals}(\boldsymbol{p}_i^{(s)}, \boldsymbol{\theta}_i^{(s)}, \{\boldsymbol{\theta}_j^{(s)}\}_{j\in\mathcal{N}_i}, \rho)$ (4.31a)
    $\boldsymbol{\theta}_i^{(s+1)} \leftarrow \texttt{primal}(\boldsymbol{p}_i^{(s+1)}, \boldsymbol{\theta}_i^{(s)}, \{\boldsymbol{\theta}_j^{(s)}\}_{j\in\mathcal{N}_i}, \rho, \mathcal{D}_i)$ (4.31b)
  end for
end for
for each $i \in \mathcal{V}$ **do** ▷ Local Covariance Inversion
  $\hat{\boldsymbol{\theta}} \leftarrow \boldsymbol{\theta}_i^{\text{end}}$
  $\boldsymbol{C}^{-1}_{\theta,i} \leftarrow \texttt{invert}(k, \boldsymbol{X}_i, \hat{\boldsymbol{\theta}})$
end for
Return $\hat{\boldsymbol{\theta}}$, $\boldsymbol{C}^{-1}_{\theta,i}$

scheme,

$$\boldsymbol{p}_i^{(s+1)} = \boldsymbol{p}_i^{(s)} + \rho \sum_{j\in\mathcal{N}_i} \left(\boldsymbol{\theta}_i^{(s)} - \boldsymbol{\theta}_j^{(s)}\right), \tag{4.31a}$$

$$\boldsymbol{\theta}_i^{(s+1)} = \arg\min_{\boldsymbol{\theta}_i} \left\{ \mathcal{L}_i(\boldsymbol{\theta}_i) + \boldsymbol{\theta}_i^{\intercal} \boldsymbol{p}_i^{(s+1)} + \rho \sum_{j\in\mathcal{N}_i} \left\| \boldsymbol{\theta}_i - \frac{\boldsymbol{\theta}_i^{(s)} + \boldsymbol{\theta}_j^{(s)}}{2} \right\|_2^2 \right\}, \tag{4.31b}$$

where $\rho > 0$ is the penalty term of the augmented Lagrangian and $\boldsymbol{p}_i^{(s)} = \sum_{j\in\mathcal{N}_i}(\boldsymbol{u}_{ij}^{(s)} + \boldsymbol{v}_{ij}^{(s)})$ is the sum of the dual variables $\boldsymbol{u}_{ij}^{(s)}$ and $\boldsymbol{v}_{ij}^{(s)}$ corresponding to constraints (4.28) and (4.29). Note that (4.31a) imposes initial values $\boldsymbol{p}_i^{(0)} = \mathbf{0}$.

The workflow is as follows. Every agent i communicates to its neighbors $j \in \mathcal{N}_i$ the current estimate of the hyperparameters $\boldsymbol{\theta}_i^{(s)}$. After each agent gathers all $\boldsymbol{\theta}_j^{(s)}$ vectors from its neighborhood, then the sum of the dual variables vector is updated (4.31a) to obtain $\boldsymbol{p}_i^{(s+1)}$. Next, every agent i solves a nested optimization problem (4.31b) to compute $\boldsymbol{\theta}_i^{(s+1)}$. The method iterates until it reaches a predefined maximum iteration number $s^{\text{end}}_{\text{DEC-c-GP}}$. The main routine of DEC-c-GP is provided in Algorithm 3. The proposed method is decentral-

Table 4.4: Time, Space, and Communication Complexity of Decentralized Factorized GP Training with ADMM-based Methods

		DEC-C-GP	DEC-APX-GP	DEC-GAPX-GP
Local	Time	$\mathcal{O}(s^{\text{end}}_{\text{nest}}(N^3/M^3))$	$\mathcal{O}(N^3/M^3)$	$\mathcal{O}(8(N^3/M^3))$
	Space	$\mathcal{O}(\xi)$	$\mathcal{O}(\xi)$	$\mathcal{O}(2\xi + 2(N^2/M^2))$
	Communication	$\mathcal{O}(s^{\text{end}}_{\text{DEC-c-GP}}(D+2))$	$\mathcal{O}(s^{\text{end}}_{\text{DEC-apx-GP}}(D+2))$	$\mathcal{O}(s^{\text{end}}_{\text{DEC-gapx-GP}}(D+2))$

$s^{\text{end}}_{\text{DEC-c-GP}} < s^{\text{end}}_{\text{DEC-apx-GP}} \approx s^{\text{end}}_{\text{DEC-gapx-GP}}$, $\xi = N^2/M^2 + D(N/M)$.

ized (executed in parallel), requiring exclusively neighbor-wise communication as shown in Fig. 4.2-(a). Note that the inter-agent communications do not involve any data exchange which satisfies Assumption 4.2. Provided that the graph topology is connected (Assumption 4.1), then DEC-c-GP (4.31) addresses Problem 2.

Let the total number of iterations for the nested optimization problem (4.31b) be $s^{\text{end}}_{\text{nest}}$. The time complexity of every agent i is dominated by the inverse of the local covariance matrix $\boldsymbol{C}^{-1}_{\theta,i}$ for every iteration of the nested optimization problem (4.31b), which results in $\mathcal{O}(s^{\text{end}}_{\text{nest}} N_i^3) = \mathcal{O}(s^{\text{end}}_{\text{nest}}(N^3/M^3))$ computations. The gradient for the nested optimization is provided in Appendix A.2. Moreover, every agent i occupies $\mathcal{O}(N_i^2 + DN_i + (D+2) + (\text{card}(\mathcal{N}_i)+1)(D+2)) = \mathcal{O}(N^2/M^2 + D(N/M) + (\text{card}(\mathcal{N}_i)+2)(D+2))$ memory to store the local inverted covariance matrix $\boldsymbol{C}^{-1}_{\theta,i}$, the local dataset $\mathcal{D}_i$, the sum of dual variables vector at the previous iteration $\boldsymbol{p}_i^{(s)}$, the hyperparameter vector at the previous iteration $\boldsymbol{\theta}_i^{(s)}$, and the hyperparameter vectors of all neighbors at the previous iteration $\{\boldsymbol{\theta}_j^{(s)}\}_{j\in\mathcal{N}_i}$. The total number of communications for each agent is $\mathcal{O}(s^{\text{end}}_{\text{DEC-c-GP}}(D+2))$ to transmit the hyperparameters to its neighbors.

Proposition 4.19. *Under Assumptions 4.1, 4.2, 4.5, 4.18 for the local sub-model $\mathcal{M}_i$, then DEC-c-GP (4.31) converges to a stationary solution $\lim_{s\to\infty} \boldsymbol{\theta}_i^{(s)} = \boldsymbol{\theta}^\star$ of (P4) for all $i \in \mathcal{V}$.*

Proof. The proof is direct application of [82, Proposition 2].

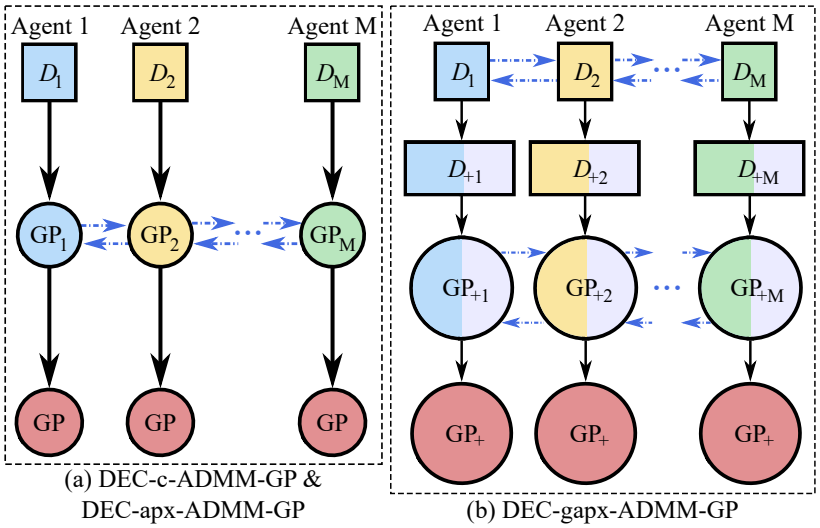


Figure 4.2: The structure of the proposed decentralized factorized GP training methods. Blue dotted lines correspond to communication (strongly connected). a) Every agent i has access to the local dataset $\mathcal{D}_i$. The agents are allowed to have their own opinion on the hyperparameter $\boldsymbol{\theta}_i$ using exclusively $\mathcal{D}_i$, but after communicating they all agree on the same hyperparameters $\boldsymbol{\theta}$. b) Every agent i has access to $\mathcal{D}_i$. Next, they communicate to form the local augmented dataset $\mathcal{D}_{+i}$ which comprises of $\mathcal{D}_i$ (local color) and the global communication dataset $\mathcal{D}_{\text{c}}$ (gray color). The agents are allowed to have their own opinion on the hyperparameter $\boldsymbol{\theta}_i$ using exclusively $\mathcal{D}_{+i}$, but after communicating they all agree on the same hyperparameters $\boldsymbol{\theta}$.

Remark 4.20. The main disadvantage of the proposed DEC-c-GP method is the cubic computations on the number of local observations for every iteration of the nested optimization (4.31b), which results in a computationally demanding process.

4.3.2 DEC-apx-GP

To address the computational complexity problem of DEC-c-GP (Remark 4.20) we consider an inexact proximal step based on a first-order approximation on the local log-likelihood $\mathcal{L}_i$ around $\boldsymbol{\theta}^{(s)}$ which yields,

$$\mathcal{L}_i(\boldsymbol{\theta}_i) \approx \nabla_{\boldsymbol{\theta}}^{\intercal} \mathcal{L}_i(\boldsymbol{\theta}_i^{(s)})(\boldsymbol{\theta}_i - \boldsymbol{\theta}_i^{(s)}) + \frac{\kappa_i}{2} \|\boldsymbol{\theta}_i - \boldsymbol{\theta}_i^{(s)}\|_2^2, \tag{4.32}$$

Algorithm 4 DEC-apx-GP

Input: $\mathcal{D}_i(\boldsymbol{X}_i, \boldsymbol{y}_i)$, $k(\cdot,\cdot)$, ρ, $\mathcal{N}_i$, κ_i, $s^{\text{end}}_{\text{DEC-apx-GP}}$

Output: $\hat{\boldsymbol{\theta}}$, $\boldsymbol{C}_{\theta,i}^{-1}$

1: Identical to Algorithm 3 with (4.31b) replaced by (4.35b)

where $\kappa_i > 0$ is a positive constant for all $i \in \mathcal{V}$. To this end, we obtain the DEC-px-ADMM [19] iterative scheme,

$$\boldsymbol{p}_i^{(s+1)} = \boldsymbol{p}_i^{(s)} + \rho \sum_{j \in \mathcal{N}_i} \left(\boldsymbol{\theta}_i^{(s)} - \boldsymbol{\theta}_j^{(s)} \right), \tag{4.33a}$$

$$\begin{aligned} \boldsymbol{\theta}_i^{(s+1)} = \arg\min_{\boldsymbol{\theta}_i} \Big\{ & \nabla_{\boldsymbol{\theta}}^{\intercal} \mathcal{L}_i(\boldsymbol{\theta}_i^{(s)})(\boldsymbol{\theta}_i - \boldsymbol{\theta}_i^{(s)}) + \frac{\kappa_i}{2} \|\boldsymbol{\theta}_i - \boldsymbol{\theta}_i^{(s)}\|_2^2 \\ & + \boldsymbol{\theta}_i^{\intercal} \boldsymbol{p}_i^{(s+1)} + \rho \sum_{j \in \mathcal{N}_i} \left\| \boldsymbol{\theta}_i - \frac{\boldsymbol{\theta}_i^{(s)} + \boldsymbol{\theta}_j^{(s)}}{2} \right\|_2^2 \Big\}. \end{aligned} \tag{4.33b}$$

Essentially, the linearization (4.32) allows the evaluation of the local log-likelihood function $\mathcal{L}_i$ (4.9) at a fixed point $\boldsymbol{\theta}_i^{(s)}$ and not at the optimizing variable $\boldsymbol{\theta}_i$. To this end, the nested optimization of (4.33b) entails significantly less computations than (4.31b). For the special case of factorized GP training problem (P4), an analytical solution of (4.33b) can be derived.

Theorem 4.21. *Let Assumption 4.1, 4.2, 4.5, 4.16, and 4.18 hold for the local sub-model $\mathcal{M}_i$. Suppose that the penalty term of the first-order approximation κ_i is sufficiently large,*

$$\kappa_i > \frac{L_i^2}{m_i^2} - \rho \underline{\lambda}(\boldsymbol{D} + \boldsymbol{A}) > 0. \tag{4.34}$$

Then, the hyperparameter update (4.33b) admits a closed-form solution, resulting in the

iterative scheme of DEC-apx-GP,

$$\boldsymbol{p}_i^{(s+1)} = \boldsymbol{p}_i^{(s)} + \rho \sum_{j \in \mathcal{N}_i} \left(\boldsymbol{\theta}_i^{(s)} - \boldsymbol{\theta}_j^{(s)} \right), \tag{4.35a}$$

$$\boldsymbol{\theta}_i^{(s+1)} = \frac{1}{\kappa_i + 2\text{card}(\mathcal{N}_i)\rho} \Big(\rho \sum_{j \in \mathcal{N}_i} \boldsymbol{\theta}_j^{(s)} - \nabla_{\boldsymbol{\theta}} \mathcal{L}_i(\boldsymbol{\theta}_i^{(s)}) + (\kappa_i + \text{card}(\mathcal{N}_i)\rho)\boldsymbol{\theta}_i^{(s)} - \boldsymbol{p}_i^{(s+1)} \Big), \tag{4.35b}$$

that converges to a stationary solution $(\boldsymbol{\theta}_i^\star, \boldsymbol{p}^\star)$ *of (P4) for all local entities* $i \in \mathcal{V}$.

Proof. The proof is provided in Appendix B.2.

The condition to select the penalty parameter κ_i (4.34) depends on the graph topology. This implies that the stronger the network the faster the convergence.

The workflow of DEC-apx-GP is identical to DEC-c-GP, yet the hyperparameter update step (4.35b) is performed analytically without requiring a nested optimization update as in (4.31b) or (4.33b). Implementation details are given in Algorithm 4 and the structure is illustrated in Fig. 4.2-(a). The gradient of the local log-likelihood $\nabla_{\boldsymbol{\theta}} \mathcal{L}_i$ can be computed similarly to (4.5). The proposed iterative method (4.35) tackles Problem 2.

The local time complexity of DEC-apx-GP is reduced to $\mathcal{O}(N_i^3) = \mathcal{O}(N^3/M^3)$ for the inversion of the local covariance matrix $\boldsymbol{C}_{\theta,i}^{-1}$ just once at every ADMM iteration. The space complexity is identical to DEC-c-GP and the total communications entail $\mathcal{O}(s^{\text{end}}_{\text{DEC-apx-GP}}(D+2))$ messages.

Remark 4.22. A disadvantage of both decentralized methods DEC-c-GP and DEC-apx-GP is the poor approximation capabilities when the number of agents increases, similarly to Section 4.2.1. In particular, Assumption 4.5 is violated as we increase the number of sub-models $\mathcal{M}_i$.

Algorithm 5 DEC-gapx-GP

Input: $\mathcal{D}_i(\boldsymbol{X}_i, \boldsymbol{y}_i)$, $k(\cdot,\cdot)$, ρ, $\mathcal{N}_i$, κ_i, $s^{\text{end}}_{\text{DEC-gapx-GP}}$
Output: $\hat{\boldsymbol{\theta}}$, $\boldsymbol{C}^{-1}_{\theta,+i}$, $\mathcal{D}_{+i}$

1: **for each** $i \in \mathcal{V}$ **do**
2: $\quad \mathcal{D}_{\text{c},i} \leftarrow \texttt{Sample}(\mathcal{D}_i)$
3: $\quad \mathcal{D}_{\text{c}} \leftarrow \texttt{flooding}(\mathcal{D}_{\text{c},i})$
4: $\quad \mathcal{D}_{+i} = \mathcal{D}_i \cup \mathcal{D}_{\text{c}}$
5: **end for**
6: $\boldsymbol{C}^{-1}_{\theta,+i} \leftarrow \texttt{DEC-apx-GP}(\mathcal{D}_{+i}, k, \rho, \mathcal{N}_i, \kappa_i, s^{\text{end}}_{\text{DEC-gapx-GP}})$
7: Return $\hat{\boldsymbol{\theta}}$, $\boldsymbol{C}^{-1}_{\theta,+i}$, $\mathcal{D}_{+i}$

4.3.3 DEC-gapx-GP

We propose to extend the computationally efficient DEC-apx-GP method with a local augmented dataset $\mathcal{D}_{+i}$ for all $i \in \mathcal{V}$ to address the poor approximation capabilities of (4.31) and (4.35) when the network has large number of nodes (Remark 4.22). The idea is similar to the centralized gapx-GP method as presented in Section 4.2.2. In order to reduce the approximation error, we relax Assumption 4.2 by allowing exchange of local subsets of data Assumption 4.3. We termed the proposed method as generalized DEC-apx-GP (DEC-gapx-GP).

Since the network has a decentralized topology, flooding [125] is employed to broadcast the local sample datasets $\mathcal{D}_{\text{c},i}$ and form the communication dataset $\mathcal{D}_{\text{c}}$. The rest is a direct application of DEC-apx-GP with the local augmented datatset $\mathcal{D}_{+i}$ for all $i \in \mathcal{V}$. Algorithm 5 presents the implementation details of DEC-gapx-GP. In Fig. 4.2-(b) the structure of the proposed method is illustrated. Larger circular objects indicate that the augmented covariance matrices $\boldsymbol{C}_{\theta,+i}$ have double dimension, i.e., $2N_i \times 2N_i$ for all $i \in \mathcal{V}$. In addition, the larger rectangular blocks represent the double size local augmented datasets $\mathcal{D}_{+i} \in \mathbb{R}^{2N_i}$. The proposed method addresses Problem 3.

The local time complexity of gapx-GP entails $\mathcal{O}((2N_i)^3) = \mathcal{O}(8(N^3/M^3))$ computations

to invert the local augmented covariance matrix $\boldsymbol{C}_{\theta,+i} = \boldsymbol{K}_{+i} + \sigma_\epsilon^2 I_{2N_i} \in \mathbb{R}^{2N_i \times 2N_i}$. The proposed method requires $\mathcal{O}((2N_i)^2 + D(2N_i) + (\text{card}(\mathcal{N}_i) + 2)(D+2)) = \mathcal{O}(4(N^2/M^2) + 2D(N/M) + (\text{card}(\mathcal{N}_i) + 2)(D+2))$ space to store the local augmented covariance matrix $\boldsymbol{C}_{\theta,+i}^{-1}$, the local augmented dataset $\mathcal{D}_{+i}$, the sum of dual variables vector at the previous iteration $\boldsymbol{p}_i^{(s)}$, the hyperparameter vector at the previous iteration $\boldsymbol{\theta}_i^{(s)}$, and the hyperparameter vectors of all neighbors at the previous iteration $\{\boldsymbol{\theta}_j^{(s)}\}_{j \in \mathcal{N}_i}$. The total communication overhead is $\mathcal{O}(s^{\text{end}}_{\text{DEC-gapx-GP}}(D+2))$.

In Table 4.4, we list the time, space, and communication complexity for the proposed decentralized factorized GP training methods. The DEC-c-GP is the most computationally expensive method, but it requires less communications than the other methods to converge. Therefore, the DEC-c-GP method favors applications with significant computational resources on the local nodes. Note that this method can also be extended with local augmented dataset $\mathcal{D}_{+i}$ for all $i \in \mathcal{V}$. Next, the DEC-apx-GP is the computationally most affordable method. The DEC-gapx-GP stands between the two former methods on time complexity, but requires more space. However, the latter can produce more accurate estimates of the hyperparameters.

Remark 4.23. Assumption 4.18 requires the local log-likelihood function $\mathcal{L}_i$ to be strongly convex. Similarly to the global log-likelihood $\mathcal{L}$, this is not guaranteed for the local log-likelihoods $\mathcal{L}_i$ for all $i \in \mathcal{V}$. Usually $\mathcal{L}_i$ is nonconvex with respect to the hyperparameters $\boldsymbol{\theta}_i$ [79, 96, 104]. This is a well known issue of GP hyperparameter training with MLE. A common trick to address the nonconvexity problem is to use multiple starting points to the optimization problem [8, 21, 104]. Consequently, in this work we follow a similar approach. Note that as we increase the observations the local log-likelihoods tend to be unimodal distributions around the hyperparameters, and thus Assumption 4.18 is satisfied [79].

Remark 4.24. There is no condition to evaluate the termination of the decentralized al-

gorithms, c-GP, apx-GP, and gapx-GP. To this end, we resort to predetermined number of iterations s^{end} which imposes additional computations, storage, and neighbor-wise communications from each agent.

4.4 Proposed Decentralized GP Prediction

In this section, we discuss the discrete-time average consensus (DAC) method [93], the Jacobi over-relaxation method (JOR) [10, Ch. 2.4], and a distributed algorithm to solve systems of linear equations (DALE) [78, 130]. In addition, we introduce a technique to identify statistically correlated agents for the location of interest. We combine these tools to approximate the aggregation of GP experts methods in a decentralized fashion.

4.4.1 Decentralized Aggregation Methods

DAC: The DAC is an iterative and parallel method to compute the average of a vector $\boldsymbol{w} \in \mathbb{R}^{\mathbb{M}}$ within a network. More specifically, every agent i has access to one element $w_i \in \mathbb{R}$ and the goal is to compute the average $\bar{\boldsymbol{w}} = (1/M)\sum_{i=1}^{M} w_i$. The DAC update law yields,

$$w_i^{(s+1)} = w_i^{(s)} + \epsilon \sum_{j \in \mathcal{N}_i(t)} a_{ij}(t)(w_j^{(s)} - w_i^{(s)}), \tag{4.36}$$

where ϵ is the parameter of the Perron matrix and $a_{ij}(t)$ is the (i,j)-th entry of the adjacency matrix. Use of consensus protocols implicitly requires that each node can distributively determine convergence in the network. In other words, just because an agent converged, that does not imply that the network has reached consensus. We employ a maximin stopping criterion [138] to locally detect convergence in the network. An additional assumption is required to implement the DAC.

Assumption 4.25. *The total number of agents M is known.*

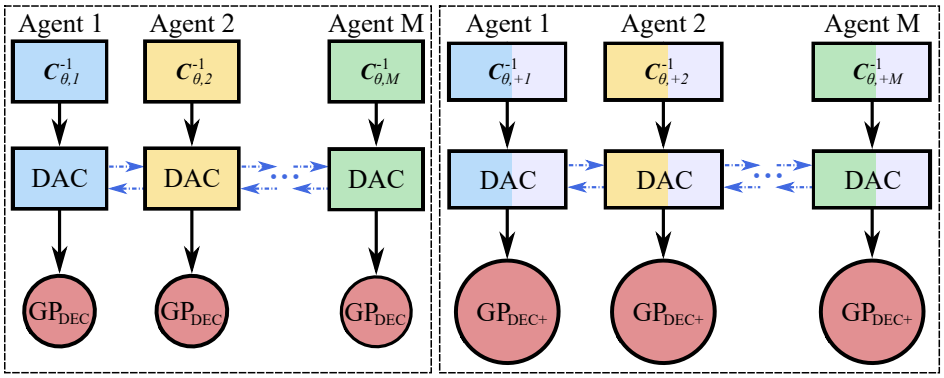
Lemma 4.26. *[93, Theorem 2], [94, Corollary 5.2] Let Assumption 4.1 hold. If $\epsilon \in (0, 1/\Delta)$, then the DAC (4.36) converges to the average $\bar{\boldsymbol{w}}$ for any initialization $w_i^{(0)}$ with convergence time $T_M(\epsilon) = \mathcal{O}(M^3 \log(M/\epsilon))$.*

JOR: The JOR is an iterative and parallel method to solve a system of linear algebraic equations in the form of $\boldsymbol{H}\boldsymbol{q} = \boldsymbol{b}$, where $\boldsymbol{H} = [h_{ij}] \in \mathbb{R}^{M \times M}$ is a known non-singular matrix with non-zero diagonal entries $h_{ii} \neq 0$, $\boldsymbol{b} \in \mathbb{R}^M$ is a known vector, and $\boldsymbol{q} \in \mathbb{R}^M$ is an unknown vector. More specifically, the i-th node knows: i) the i-th row of the known matrix $\text{row}_i\{\boldsymbol{H}\} \in \mathbb{R}^{1 \times M}$; and ii) the i-th element of the known vector $b_i \in \mathbb{R}$. The objective is to find $q_i \in \mathbb{R}$, the i-th element of the unknown vector $\boldsymbol{q}$. The JOR iterative scheme yields,

$$q_i^{(s+1)} = (1 - \omega) q_i^{(s)} + \frac{\omega}{h_{ii}} \left(b_i - \sum_{j \neq i} h_{ij} q_j^{(s)} \right), \tag{4.37}$$

where $\omega \in (0, 1)$ the relaxation parameter.

Remark 4.27. The limit of the summation in (4.37) requires communication with all agents, as it is computed over j other than i. This means that each agent must know the update value (4.37) of every other agent $\{\boldsymbol{q}_j^{(s)}\}_{j \neq i}$, i.e. $j \neq i \implies j \in \mathcal{V} \backslash i$. That is a major restriction, as it imposes a strongly complete graph topology (Figure 4.1). Although in [23, 24, 25] JOR is used for distributed networks, it is unrealistic for many network applications due to limited communication. However, we evaluate the use of JOR, as in some applications with small fleet size, strongly complete networks are feasible. For not strongly complete network topologies, distributed flooding is required at every iteration to obtain $\{\boldsymbol{q}_j^{(s)}\}_{j \neq i}$ and implement (4.37). The number of inter-agent communications for distributed flooding is the diameter of the graph $\text{diam}(\mathcal{G})$. Thus, the total number of iterations yields $s_{\text{JOR}} = \text{diam}(\mathcal{G}) s_{\text{JOR}}^{\text{end}}$.



(a) DEC-(g)PoE, DEC-(r)BCM (b) DEC-grBCM

Figure 4.3: The structure of the proposed DEC-PoE and DEC-BCM families. Blue dotted lines correspond to communication (strongly connected). Every agent implements discrete-time average consensus (DAC) methods.

Lemma 4.28. *[127, Theorem 2] Let the graph $\mathcal{G}$ be time-invariant and strongly complete. If $\boldsymbol{H}$ is symmetric and PD, and $\omega < 2/M$, then the JOR converges to the solution for any initialization $q_i^{(0)}$.*

Lemma 4.29. *[127, Theorem 4] Let the graph $\mathcal{G}$ be time-invariant and strongly complete. If $\boldsymbol{H}$ is symmetric and PD, and $\omega^\star = 2/(\overline{\lambda}(\boldsymbol{R}) + \underline{\lambda}(\boldsymbol{R}))$ where $\boldsymbol{R} = \mathrm{diag}(\boldsymbol{H})^{-1}\boldsymbol{H}$, then the JOR converges to the solution for any initialization $q_i^{(0)}$ with the optimal rate.*

Remark 4.30. The difference between Lemma 4.28 and 4.29 is that the latter employs the optimal relaxation factor $\omega^\star$ which is characterized by the eigenvalues of $\boldsymbol{R}$. In principle, the smaller the relaxation factor ω the slower the convergence speed [45]. Since $\omega^* > 2/M$, the optimal relaxation leads to faster convergence of JOR to the solution. To compute $\omega^\star$ in a network of agents, additional communication is required to distributively estimate the maximum and minimum eigenvalues of $\boldsymbol{R}$. However, the sufficient condition for ω of Lemma 4.28 can be locally computed with no communication. Let the distributed method for the computation of $\omega^\star$ entail $s_{\omega^\star}^{\text{end}}$ iterations, JOR with ω from Lemma 4.28 converge after $s_{\text{JOR}}^{\text{end}}$ iterations, and JOR with $\omega^\star$ from from Lemma 4.29 converge after $s_{\text{JOR}^\star}^{\text{end}}$ iterations. Then, $\omega^\star$ is communication-wise more efficient in decentralized networks when $s_{\omega^\star}^{\text{end}} + s_{\text{JOR}^\star}^{\text{end}} < s_{\text{JOR}}^{\text{end}}$.

PM: The optimal relaxation factor $\omega^\star$ involves the maximum eigenvalue $\overline{\lambda}(\boldsymbol{R})$ and minimum eigenvalue $\underline{\lambda}(\boldsymbol{R})$ (Lemma 4.29). We employ the power method (PM) to compute $\overline{\lambda}(\boldsymbol{R})$ and the inverse power method (IPM) to compute $\underline{\lambda}(\boldsymbol{R})$. The PM is a two step iterative algorithm that follows,

$$\boldsymbol{g}^{(s+1)} = \boldsymbol{R}\boldsymbol{e}^{(s)} \tag{4.38a}$$

$$\boldsymbol{e}^{(s+1)} = \boldsymbol{g}^{(s+1)}/\|\boldsymbol{g}^{(s+1)}\|_\infty, \tag{4.38b}$$

where $\|\cdot\|_\infty$ denotes the infinity norm. As the PM algorithm converges $\|\boldsymbol{e}^{(s)} - \boldsymbol{e}^{(s-1)}\|_2 \to 0$, the infinity norm approximates the dominant eigenvalue $\|\boldsymbol{g}^{(s)}\|_\infty \approx \overline{\lambda}(\boldsymbol{R})$. After obtaining $\overline{\lambda}(\boldsymbol{R})$, we formulate the spectral shift of $\boldsymbol{R}$, that is $\boldsymbol{B} = \boldsymbol{R} - \overline{\lambda}(\boldsymbol{R})I_M$. The IPM is the application of PM (4.38) on $\boldsymbol{B}$. Then, we derive the minimum eigenvalue as $\underline{\lambda}(\boldsymbol{R}) = |\overline{\lambda}(\boldsymbol{B}) - \overline{\lambda}(\boldsymbol{R})|$. In order to obtain both $\overline{\lambda}(\boldsymbol{R})$ and $\underline{\lambda}(\boldsymbol{R})$, we need to execute the PM algorithm (4.38) two sequential times. Let the first PM algorithm to converge after $s^{\text{end}}_{\text{PM}}$ iterations and the second after $s^{\text{end}}_{\text{IPM}}$ iterations. The use of the optimal relaxation is communication-wise more efficient if $s^{\text{end}}_{\text{PM}} + s^{\text{end}}_{\text{IPM}} + s^{\text{end}}_{\text{JOR}^\star} < s^{\text{end}}_{\text{JOR}}$ (Remark 4.30). Note that if $\boldsymbol{H}$ is symmetric, then $\boldsymbol{R} = \text{diag}(\boldsymbol{H})^{-1}\boldsymbol{H}$ is also symmetric, as the only changes occur in the diagonal elements, with $\text{diag}(\boldsymbol{H})^{-1} = \{\boldsymbol{H}_{ii}^{-1}\}_{i=1}^M$.

Lemma 4.31. *[40, Chapter 8] Let the graph $\mathcal{G}$ be time-invariant and strongly complete. If $\boldsymbol{H}$ is symmetric, then the PM converges to the dominant real eigenvalue $\overline{\lambda}(\boldsymbol{R})$ with convergence rate $\mathcal{O}((\lambda_2/\overline{\lambda})^{s^{end}_{PM}})$, where λ_2 is the second largest eigenvalue.*

DEC-PoE Family

The decentralized PoE (DEC-PoE) method makes use of two DAC algorithms (FIgure 4.3-(a)). The first DAC computes the average $(1/M)\sum_{i=1}^M \beta_i\sigma_i^{-2}$ and the second DAC the aver-

Table 4.5: Communication Complexity of Decentralized GP Aggregations

Method	Graph	Communication Complexity
DEC-PoE	SC	$\mathcal{O}(2\chi)$
DEC-gPoE	SC	$\mathcal{O}(2\chi)$
DEC-BCM	SC	$\mathcal{O}(2\chi)$
DEC-rBCM	SC	$\mathcal{O}(3\chi)$
DEC-grBCM	SC	$\mathcal{O}(3\chi)$
DEC-NPAE	SCC	$\mathcal{O}(2Ms^{\text{end}}_{\text{JOR}} + 2\chi + M\xi)$
DEC-NPAE*	SCC	$\mathcal{O}(2M(s^{\text{end}}_{\text{JOR}^\star} + s^{\text{end}}_{\text{PM}}) + 2\chi + M\xi + M^2)$

SC: strongly connected, SCC: strongly complete connected, $\chi = s^{\text{end}}_{\text{DAC}}\text{card}(\mathcal{N}_i)$, $\xi = N^2/M^2 + D(N/M)$.

Algorithm 6 DEC-PoE

Input: $\mathcal{D}_i(\boldsymbol{X}_i, \boldsymbol{y}_i)$, $\hat{\boldsymbol{\theta}}$, $\boldsymbol{C}^{-1}_{\theta,i}$, $\mathcal{N}_i$, k, M, $\boldsymbol{x}_*$, Δ
Output: $\mu_{\text{DEC-PoE}}$, $\sigma^{-2}_{\text{DEC-PoE}}$

$\epsilon = 1/\Delta$
for each $i \in \mathcal{V}$ **do**
  $\mu_i \leftarrow \texttt{localMean}(\boldsymbol{x}_*, k, \hat{\boldsymbol{\theta}}, \mathcal{D}_i, \boldsymbol{C}^{-1}_{\theta,i})$ (4.11)
  $\sigma_i^{-2} \leftarrow \texttt{localVariance}(\boldsymbol{x}_*, k, \hat{\boldsymbol{\theta}}, \mathcal{D}_i, \boldsymbol{C}^{-1}_{\theta,i},)$ (4.12)
  initialize $w^{(0)}_{\mu,i} = \beta_i\sigma_i^{-2}\mu_i$, $w^{(0)}_{\sigma^{-2},i} = \beta_i\sigma_i^{-2}$, $\beta_i = 1$
  repeat
    communicate $w^{(s)}_{\mu,i}$, $w^{(s)}_{\sigma^{-2},i}$ to agents in $\mathcal{N}_i$
    $w^{(s+1)}_{\mu,i} \leftarrow \texttt{DAC}(\epsilon, w^{(s)}_{\mu,i}, \{\boldsymbol{w}^{(s)}_{\mu,j}\}_{j\in\mathcal{N}_i}, \mathcal{N}_i)$ (4.36) ▷ DAC1
    $w^{(s+1)}_{\sigma^{-2},i} \leftarrow \texttt{DAC}(\epsilon, w^{(s)}_{\sigma^{-2},i}, \{\boldsymbol{w}^{(s)}_{\sigma^{-2},j}\}_{j\in\mathcal{N}_i}, \mathcal{N}_i)$ (4.36) ▷ DAC2
  until maximin stopping criterion
  $\sigma^{-2}_{\text{DEC-PoE}} = Mw^{(\text{end})}_{\sigma^{-2},i}$ (4.14)
  $\mu_{\text{DEC-PoE}} = \sigma^2_{\text{DEC-PoE}}Mw^{(\text{end})}_{\mu,i}$ (4.13)
end for

age $(1/M)\sum_{i=1}^{M}\beta_i\sigma_i^{-2}\mu_i$, where $\beta_i = 1$. At every iteration of DAC each agent communicates both computed values $w^{(s)}_{\mu,i}$, $w^{(s)}_{\sigma^{-2},i}$ to its neighbors $\mathcal{N}_i$. After convergence, each DAC average is multiplied by the number of nodes M and follow (4.13), (4.14) to recover the DEC-PoE prediction mean and precision. The implementation details are given in Algorithm 6. The time and space complexity are identical to the local time and space complexity of the PoE family as listed in Table 4.2. Let $s^{\text{end}}_{\text{DAC}}$ be the maximum number of iterations of the two DAC

Algorithm 7 DEC-gPoE

Input: $\mathcal{D}_i(\boldsymbol{X}_i, \boldsymbol{y}_i)$, $\hat{\boldsymbol{\theta}}$, $\boldsymbol{C}_{\theta,i}^{-1}$, $\mathcal{N}_i$, k, M, $\boldsymbol{x}_*$, Δ
Output: $\mu_{\text{DEC-gPoE}}$, $\sigma^{-2}_{\text{DEC-gPoE}}$
1: Identical to Algorithm 6 with $\beta_i = 1/M$ instead of $\beta_i = 1$ (line 5)

Algorithm 8 DEC-BCM

Input: $\mathcal{D}_i(\boldsymbol{X}_i, \boldsymbol{y}_i)$, $\hat{\boldsymbol{\theta}}$, $\boldsymbol{C}_{\theta,i}^{-1}$, $\mathcal{N}_i$, k, M, $\boldsymbol{x}_*$, Δ
Output: $\mu_{\text{DEC-BCM}}$, $\sigma^{-2}_{\text{DEC-BCM}}$
1: $\epsilon = 1/\Delta$
2: **for each** $i \in \mathcal{V}$ **do**
3: $\mu_i \leftarrow \texttt{localMean}(\boldsymbol{x}_*, k, \hat{\boldsymbol{\theta}}, \mathcal{D}_i, \boldsymbol{C}_{\theta,i}^{-1})$ (4.11)
4: $\sigma_i^{-2} \leftarrow \texttt{localVariance}(\boldsymbol{x}_*, k, \hat{\boldsymbol{\theta}}, \mathcal{D}_i, \boldsymbol{C}_{\theta,i}^{-1},)$ (4.12)
5: $\sigma_{**}^2 = k(\boldsymbol{x}_*, \boldsymbol{x}_*)$
6: initialize $w_{\mu,i}^{(0)} = \beta_i \sigma_i^{-2} \mu_i$, $w_{\sigma^{-2},i}^{(0)} = \beta_i \sigma_i^{-2}$, $\beta_i = 1$
7: **repeat**
8: communicate $w_{\mu,i}^{(s)}$, $w_{\sigma^{-2},i}^{(s)}$ to agents in $\mathcal{N}_i$
9: $w_{\mu,i}^{(s+1)} \leftarrow \texttt{DAC}(\epsilon, w_{\mu,i}^{(s)}, \{\boldsymbol{w}_{\mu,j}^{(s)}\}_{j\in\mathcal{N}_i}, \mathcal{N}_i)$ (4.36) ▷ DAC1
10: $w_{\sigma^{-2},i}^{(s+1)} \leftarrow \texttt{DAC}(\epsilon, w_{\sigma^{-2},i}^{(s)}, \{\boldsymbol{w}_{\sigma^{-2},j}^{(s)}\}_{j\in\mathcal{N}_i}, \mathcal{N}_i)$ (4.36) ▷ DAC2
11: **until** maximin stopping criterion
12: $\sigma^{-2}_{\text{DEC-BCM}} = M w_{\sigma^{-2},i}^{(\text{end})} + (1 - \sum_{i=1}^{M} \beta_i)\sigma_{**}^{-2}$ (4.16)
13: $\mu_{\text{DEC-BCM}} = \sigma^2_{\text{DEC-BCM}} M w_{\mu,i}^{(\text{end})}$ (4.15)
14: **end for**

to converge. The total communications are $\mathcal{O}(2s_{\text{DAC}}^{\text{end}}\text{card}(\mathcal{N}_i))$ for all $i \in \mathcal{V}$ (Table 4.5).

Next, we form the decentralized gPoE (DEC-gPoE) (Figure 4.3-(a)). The DEC-gPoE is identical to the DEC-PoE, but $\beta_i = 1/M$ instead of $\beta_i = 1$ (Algorithm 7). The time, space, and communication complexity are identical to the DEC-PoE. Both DEC-PoE and DEC-gPoE methods address Problem 4.

DEC-BCM Family

The decentralized BCM (DEC-BCM) method employs two DAC algorithms (Figure 4.3-(a)). The first DAC computes the average $(1/M)\sum_{i=1}^{M} \beta_i \sigma_i^{-2}$ and the second DAC the average $(1/M)\sum_{i=1}^{M} \beta_i \sigma_i^{-2} \mu_i$, where $\beta_i = 1$. At every iteration of DAC each agent communicates

Algorithm 9 DEC-rBCM

Input: $\mathcal{D}_i(\boldsymbol{X}_i, \boldsymbol{y}_i)$, $\hat{\boldsymbol{\theta}}$, $\boldsymbol{C}_{\theta,i}^{-1}$, $\mathcal{N}_i$, k, M, $\boldsymbol{x}_*$, Δ
Output: $\mu_{\text{DEC-rBCM}}$, $\sigma^{-2}_{\text{DEC-rBCM}}$

$\epsilon = 1/\Delta$
for each $i \in \mathcal{V}$ **do**
  $\mu_i \leftarrow \texttt{localMean}(\boldsymbol{x}_*, k, \hat{\boldsymbol{\theta}}, \mathcal{D}_i, \boldsymbol{C}_{\theta,i}^{-1})$ (4.11)
  $\sigma_i^{-2} \leftarrow \texttt{localVariance}(\boldsymbol{x}_*, k, \hat{\boldsymbol{\theta}}, \mathcal{D}_i, \boldsymbol{C}_{\theta,i}^{-1})$ (4.12)
  $\sigma_{**}^2 = k(\boldsymbol{x}_*, \boldsymbol{x}_*)$
  initialize $w_{\mu,i}^{(0)} = \beta_i \sigma_i^{-2} \mu_i$, $w_{\sigma^{-2},i}^{(0)} = \beta_i \sigma_i^{-2}$, $w_{\beta_i}^{(0)} = \beta_i$, $\beta_i = 0.5[\ln \sigma_{**}^2 - \ln \sigma_i^2]$
  repeat
    communicate $w_{\mu,i}^{(s)}$, $w_{\sigma^{-2},i}^{(s)}$, $w_{\beta_i}^{(s)}$ to agents in $\mathcal{N}_i$
    $w_{\mu,i}^{(s+1)} \leftarrow \texttt{DAC}(\epsilon, w_{\mu,i}^{(s)}, \{\boldsymbol{w}_{\mu,j}^{(s)}\}_{j\in\mathcal{N}_i}, \mathcal{N}_i)$ (4.36) ▷ DAC1
    $w_{\sigma^{-2},i}^{(s+1)} \leftarrow \texttt{DAC}(\epsilon, w_{\sigma^{-2},i}^{(s)}, \{\boldsymbol{w}_{\sigma^{-2},j}^{(s)}\}_{j\in\mathcal{N}_i}, \mathcal{N}_i)$ (4.36) ▷ DAC2
    $w_{\beta_i}^{(s+1)} \leftarrow \texttt{DAC}(\epsilon, w_{\beta_i}^{(s)}, \{\boldsymbol{w}_{\beta_j}^{(s)}\}_{j\in\mathcal{N}_i}, \mathcal{N}_i)$ (4.36) ▷ DAC3
  until maximin stopping criterion
  $\sigma^{-2}_{\text{DEC-rBCM}} = M w_{\sigma^{-2},i}^{(\text{end})} + (1 - M w_{\beta_i}^{(\text{end})}) \sigma_{**}^{-2}$ (4.16)
  $\mu_{\text{DEC-rBCM}} = \sigma^2_{\text{DEC-rBCM}} M w_{\mu,i}^{(\text{end})}$ (4.15)
end for

both computed values $w_{\mu,i}^{(s)}$, $w_{\sigma^{-2},i}^{(s)}$ to its neighbors $\mathcal{N}_i$. After convergence, each DAC average is multiplied by the number of nodes M and follow (4.15), (4.16) to recover the DEC-BCM mean and precision. The implementation details are provided in Algorithm 8. The time, space, and communication complexity are identical to the DEC-PoE family. The DEC-BCM addresses Problem 4.

We introduce the decentralized rBCM (DEC-rBCM) technique that utilizes three DAC algorithms to compute the averages $(1/M)\sum_{i=1}^M \beta_i \sigma_i^{-2}$, $(1/M)\sum_{i=1}^M \beta_i \sigma_i^{-2}\mu_i$, and $(1/M)\sum_{i=1}^M \beta_i$, where $\beta_i = 0.5[\ln \sigma_{**}^2 - \ln \sigma_i^2]$. At every iteration of DAC each agent communicates $w_{\mu,i}^{(s)}$, $w_{\sigma^{-2},i}^{(s)}$, $w_{\beta_i}^{(s)}$ to its neighbors $\mathcal{N}_i$. After convergence, each DAC average is multiplied by the number of nodes M and follow (4.15), (4.16) to recover the DEC-rBCM prediction mean and precision. Implementation details are given in Algorithm 9. The local time and space complexity are identical to the rBCM (Table 4.2). Let $s_{\text{DAC}}^{\text{end}}$ be the maximum number of iterations of the three DAC to converge. The total communications are $\mathcal{O}(3 s_{\text{DAC}}^{\text{end}} \text{card}(\mathcal{N}_i))$

Algorithm 10 DEC-grBCM

Input: $\mathcal{D}_{+i}(\boldsymbol{X}_{+i}, \boldsymbol{y}_{+i})$, $\hat{\boldsymbol{\theta}}$, $\boldsymbol{C}_{\theta,+i}^{-1}$, $\mathcal{N}_i$, k, M, $\boldsymbol{x}_*$, Δ
Output: $\mu_{\text{DEC-grBCM}}$, $\sigma^{-2}_{\text{DEC-grBCM}}$

$\epsilon = 1/\Delta$
for each $i \in \mathcal{V}$ **do**
  $\mu_{+i} \leftarrow \texttt{localMean}(\boldsymbol{x}_*, k, \hat{\boldsymbol{\theta}}, \mathcal{D}_{+i}, \boldsymbol{C}_{\theta,+i}^{-1})$ (4.11)
  $\sigma_{+i}^{-2} \leftarrow \texttt{localVariance}(\boldsymbol{x}_*, k, \hat{\boldsymbol{\theta}}, \mathcal{D}_{+i}, \boldsymbol{C}_{\theta,+i}^{-1})$ (4.12)
  $\sigma_{\text{c}}^2 = k(\boldsymbol{X}_{\text{c}}, \boldsymbol{X}_{\text{c}})$
  initialize $w_{\mu,i}^{(0)} = \beta_i \sigma_{+i}^{-2} \mu_{+i}$, $w_{\sigma^{-2},i}^{(0)} = \beta_i \sigma_{+i}^{-2}$, $w_{\beta_i}^{(0)} = \beta_i$, $\beta_i = 0.5[\ln \sigma_{\text{c}}^2 - \ln \sigma_{+i}^2]$
  repeat
    communicate $w_{\mu,i}^{(s)}$, $w_{\sigma^{-2},i}^{(s)}$, $w_{\beta_i}^{(s)}$ to agents in $\mathcal{N}_i$
    $w_{\mu,i}^{(s+1)} \leftarrow \texttt{DAC}(\epsilon, w_{\mu,i}^{(s)}, \{\boldsymbol{w}_{\mu,j}^{(s)}\}_{j\in\mathcal{N}_i}, \mathcal{N}_i)$ (4.36) ▷ DAC1
    $w_{\sigma^{-2},i}^{(s+1)} \leftarrow \texttt{DAC}(\epsilon, w_{\sigma^{-2},i}^{(s)}, \{\boldsymbol{w}_{\sigma^{-2},j}^{(s)}\}_{j\in\mathcal{N}_i}, \mathcal{N}_i)$ (4.36) ▷ DAC2
    $w_{\beta_i}^{(s+1)} \leftarrow \texttt{DAC}(\epsilon, w_{\beta_i}^{(s)}, \{\boldsymbol{w}_{\beta_j}^{(s)}\}_{j\in\mathcal{N}_i}, \mathcal{N}_i)$ (4.36) ▷ DAC3
  until maximin stopping criterion
  $\sigma^{-2}_{\text{DEC-grBCM}} = M w_{\sigma^{-2},i}^{(\text{end})} + (1 - M w_{\beta_i}^{(\text{end})}) \sigma_{\text{c}}^{-2}$ (4.18)
  $\mu_{\text{DEC-grBCM}} = \sigma^2_{\text{DEC-grBCM}} (M w_{\mu,i}^{(\text{end})} - (M w_{\beta_i}^{(\text{end})} - 1) \sigma_{\text{c}}^{-2} \mu_{\text{c}})$ (4.17)
end for

for all $i \in \mathcal{V}$ (Table 4.5). The DEC-rBCM addresses Problem 4.

We propose the decentralized grBCM (DEC-grBCM) method which employs three DAC algorithms (Figure 4.3-(b)) to compute the averages $(1/M)\sum_{i=1}^M \beta_i \sigma_{+i}^{-2}$, $(1/M)\sum_{i=1}^M \beta_i \sigma_{+i}^{-2}\mu_i$, and $(1/M)\sum_{i=1}^M \beta_i$, where $\beta_i = 0.5[\ln \sigma_{\text{c}}^2 - \ln \sigma_{+i}^2]$. At every iteration of DAC each agent communicates $w_{\mu,i}^{(s)}$, $w_{\sigma^{-2},i}^{(s)}$, $w_{\beta_i}^{(s)}$ to its neighbors $\mathcal{N}_i$. After convergence, each DAC average is multiplied by the number of nodes M and follow (4.17), (4.18) to recover the prediction mean and precision. Implementation details are given in Algorithm 10. The local time and space complexity are identical to the grBCM (Table 4.2). Let $s_{\text{DAC}}^{\text{end}}$ be the maximum number of iterations of the three DAC to converge. The total communications are $\mathcal{O}(3 s_{\text{DAC}}^{\text{end}} \text{card}(\mathcal{N}_i))$ for all $i \in \mathcal{V}$ (Table 4.5). The DEC-grBCM addresses Problem 5.

Proposition 4.32. *Let the Assumption 4.1, 4.3, 4.5, 4.11, 4.25 hold throughout the approximation. If $\omega < 2/M$ then the DEC-grBCM is consistent for any initialization.*

Proof. The proof is a direct consequence of Proposition 4.14 and Proposition 4.26.

DEC-NPAE Family

An additional assumption is required to implement the DEC-NPAE family methods.

Assumption 4.33. *The graph topology is strongly complete, i.e. every agent i can communicate with every other node $j \neq i$.*

Remark 4.34. Assumption 4.33 is conservative, but mandatory for the implementation of the PM and JOR algorithms. In order to use the DEC-NPAE family with strongly connected graph topologies, flooding is required (Remark 4.27).

We present DEC-NPAE which combines JOR and DAC to decentralize the computations (4.21), (4.22) of NPAE (Figure 4.4-(a)). We execute two parallel JOR algorithms with known matrix $\boldsymbol{H} = \boldsymbol{C}_{\theta,A}$ and known vectors: i) $\boldsymbol{b} = \boldsymbol{\mu}$; and ii) $\boldsymbol{b} = \boldsymbol{k}_A$. The first JOR is associated with the prediction mean (4.21) and the second with the variance (4.22). Note that $\boldsymbol{C}_{\theta,A}$ is a symmetric and PD covariance matrix. Implementation details are provided in Algorithm 11. We split up the computation in two parts. First, each entity computes three quantities: i) the local mean μ_i (4.11); ii) the local cross covariance $[\boldsymbol{k}_A]_i$ (4.19); and iii) the local row covariance $\text{row}_i\{\boldsymbol{C}_{\theta,A}\}$ (4.20). For the local computation of (4.20) the agents must know the inputs $\{\boldsymbol{X}_j\}_{j\neq i}$ of all other agents, to find $\boldsymbol{C}_{\theta,ij}$ and $\boldsymbol{k}_{j,*}$. The inputs $\{\boldsymbol{X}_j\}_{j\neq i}$ are communicated between agents. The local inverted covariance matrices of all other agents $\{\boldsymbol{C}_{\theta,j}^{-1}\}_{j\neq i}$ can be locally computed, but it is computationally very expensive to invert $M-1$ matrices, i.e. $\mathcal{O}(MN_i^3) = \mathcal{O}(N^3/M^2)$. Since every agent i has already stored its local covariance matrix from the training step (Section 4.3), we select to exchange $\{\boldsymbol{C}_{\theta,j}^{-1}\}_{j\neq i}$ between agents (Algorithm 11-[Line 3]). After every JOR iteration, each agent i communicates the computed values $q_{\mu,i}^{(s)}$, $q_{\sigma^2,i}^{(s)}$ to its neighbors $\mathcal{N}_i$ (Algorithm 11-[line 10]).

Algorithm 11 DEC-NPAE

Input: $\mathcal{D}_i(\boldsymbol{X}_i, \boldsymbol{y}_i)$, $\boldsymbol{X}$, $\hat{\boldsymbol{\theta}}$, $\boldsymbol{C}_{\theta,i}^{-1}$, $\mathcal{N}_i$, k, M, $\boldsymbol{x}_*$, Δ
Output: $\mu_{\text{DEC-NPAE}}$, $\sigma^2_{\text{DEC-NPAE}}$

1: initialize $\omega = 2/M$; $\epsilon = 1/\Delta$
2: **for each** $i \in \mathcal{V}$ **do**
3: communicate $\boldsymbol{C}_{\theta,i}^{-1}$, $\boldsymbol{X}_i$ to agents in $\mathcal{V}\backslash i$
4: $\mu_i \leftarrow \texttt{localMean}(\boldsymbol{x}_*, k, \hat{\boldsymbol{\theta}}, \mathcal{D}_i, \boldsymbol{C}_{\theta,i}^{-1})$ (4.11)
5: $[\boldsymbol{k}_A]_i \leftarrow \texttt{crossCov}(\boldsymbol{x}_*, k, \hat{\boldsymbol{\theta}}, \boldsymbol{X}_i, \boldsymbol{C}_{\theta,i}^{-1})$ (4.19)
6: $\text{row}_i\{\boldsymbol{C}_{\theta,A}\} \leftarrow \texttt{localCov}(\boldsymbol{x}_*, k, \hat{\boldsymbol{\theta}}, \boldsymbol{X}, \boldsymbol{C}_{\theta,i}^{-1}, \{\boldsymbol{C}_{\theta,j}^{-1}\}_{j\neq i})$ (4.20)
7: $[\boldsymbol{H}]_i = \text{row}_i\{\boldsymbol{C}_{\theta,A}\}$; $b_{\mu,i} = \mu_i$; $b_{\sigma^2,i} = [\boldsymbol{k}_A]_i$
8: initialize $q_{\mu,i}^{(0)} = b_{\mu,i}/[\boldsymbol{H}]_{ii}$, $q_{\sigma^2,i}^{(0)} = b_{\sigma^2,i}/[\boldsymbol{H}]_{ii}$
9: **repeat** ▷ 2×JOR
10: communicate $q_{\mu,i}^{(s)}$, $q_{\sigma^2,i}^{(s)}$ to agents in $\mathcal{V}\backslash i$
11: $q_{\mu,i}^{(s+1)} \leftarrow \texttt{JOR}(\omega, [\boldsymbol{H}]_i, b_{\mu,i}, q_{\mu,i}^{(s)}, \{\boldsymbol{q}_{\mu,j}^{(s)}\}_{j\neq i})$ (4.37)
12: $q_{\sigma^2,i}^{(s+1)} \leftarrow \texttt{JOR}(\omega, [\boldsymbol{H}]_i, b_{\sigma^2,i}, q_{\sigma^2,i}^{(s)}, \{\boldsymbol{q}_{\sigma^2,j}^{(s)}\}_{j\neq i})$ (4.37)
13: **until** maximin stopping criterion
14: initialize $w_{\mu,i}^{(0)} = [\boldsymbol{k}_A]_i q_{\mu,i}^{(\text{end})}$, $w_{\sigma^2,i}^{(0)} = [\boldsymbol{k}_A]_i q_{\sigma^2,i}^{(\text{end})}$
15: **repeat**
16: communicate $w_{\mu,i}^{(s)}$, $w_{\sigma^2,i}^{(s)}$ to agents in $\mathcal{N}_i$
17: $w_{\mu,i}^{(s+1)} \leftarrow \texttt{DAC}(\epsilon, w_{\mu,i}^{(s)}, \{\boldsymbol{w}_{\mu,j}^{(s)}\}_{j\in\mathcal{N}_i}, \mathcal{N}_i)$ (4.36) ▷ DAC1
18: $w_{\sigma^2,i}^{(s+1)} \leftarrow \texttt{DAC}(\epsilon, w_{\sigma^2,i}^{(s)}, \{\boldsymbol{w}_{\sigma^2,j}^{(s)}\}_{j\in\mathcal{N}_i}, \mathcal{N}_i)$ (4.36) ▷ DAC2
19: **until** maximin stopping criterion
20: $\mu_{\text{DEC-NPAE}} = M w_{\mu,i}^{(\text{end})}$
21: $\sigma^2_{\text{DEC-NPAE}} = \sigma_f^2(k_{**} - M w_{\sigma^2,i}^{(\text{end})})$
22: **end for**

Next, we compute an element of the unknown vectors $q_{\mu,i} = [\boldsymbol{C}_{\theta,A}^{-1}\boldsymbol{\mu}]_i$, $q_{\sigma^2,i} = [\boldsymbol{C}_{\theta,A}^{-1}\boldsymbol{k}_A]_i$ (Algorithm 11-[lines 11, 12]) with the JOR method. When JOR converges, each agent computes locally the i-th element of the resulting summation from: i) the multiplication between the vectors $\boldsymbol{k}_A^\intercal$ and $\boldsymbol{C}_{\theta,A}^{-1}\boldsymbol{\mu}$ (4.21), that is $w_{\mu,i} = [\boldsymbol{k}_A]_i q_{\mu,i}^{(\text{end})}$; and ii) the multiplication between the vectors $\boldsymbol{k}_A^\intercal$ and $\boldsymbol{C}_{\theta,A}^{-1}\boldsymbol{k}_A$ (4.22), that is $w_{\sigma^2,i} = [\boldsymbol{k}_A]_i q_{\sigma^2,i}^{(\text{end})}$. Second, since all agents have stored a part of the summations $w_{\mu,i}$, $w_{\sigma^2,i}$, we use the DAC to compute the averages $(1/M)\sum_{i=1}^M [\boldsymbol{k}_A]_i q_{\mu,i}^{(\text{end})}$ and $(1/M)\sum_{i=1}^M [\boldsymbol{k}_A]_i q_{\sigma^2,i}^{(\text{end})}$. After every DAC iteration, each agent i communicates the computed values $w_{\mu,i}^{(s)}$, $w_{\sigma^2,i}^{(s)}$ to its neighbors $\mathcal{N}_i$. When both DAC converge, each agent follows (4.21), (4.22) to recover the DEC-NPAE mean and variance.

Algorithm 12 POWERMETHOD

Input: $\boldsymbol{R}$, $\mathcal{N}_i$, M, η_{PM}
Output: $\overline{\lambda}(\boldsymbol{R})$

initialize $\boldsymbol{e}^{(0)} = 1/M$
repeat
  $g_i^{(s+1)} = \text{row}_i\{\boldsymbol{R}\}\boldsymbol{e}^{(s)}$ (4.38a)
  communicate $g_i^{(s+1)}$ to agents in $\mathcal{V}\backslash i$
  $\|\boldsymbol{g}^{(s+1)}\|_\infty = \max\{|\boldsymbol{g}^{s+1}|\}$
  $\boldsymbol{e}^{(s+1)} = \boldsymbol{g}^{(s+1)}/\|\boldsymbol{g}^{(s+1)}\|_\infty$ (4.38b)
until $\|\boldsymbol{e}^{(s+1)} - \boldsymbol{e}^{(s)}\|_2 < \eta_{\text{PM}}$
$\overline{\lambda}(\boldsymbol{R}) = \|\boldsymbol{g}^{(\text{end})}\|_\infty$

Algorithm 13 DEC-NPAE*

Input: $\mathcal{D}_i(\boldsymbol{X}_i, \boldsymbol{y}_i)$, $\boldsymbol{X}$, $\hat{\boldsymbol{\theta}}$, $\boldsymbol{C}_{\theta,i}^{-1}$, $\mathcal{N}_i$, k, M, $\boldsymbol{x}_*$, Δ, η_{PM}
Output: $\mu_{\text{DEC-NPAE}^\star}$, $\sigma^2_{\text{DEC-NPAE}^\star}$

for each $i \in \mathcal{V}$ **do**
  communicate $\text{row}_i\{\boldsymbol{C}_{\theta,A}\}$ to agents in $\mathcal{V}\backslash i$
  $\text{diag}(\boldsymbol{C}_{\theta,A})^{-1} = \text{diag}(\{\boldsymbol{C}_{\theta,A}\}_{ii}^{-1})$
  $\boldsymbol{R} = \text{diag}(\boldsymbol{C}_{\theta,A})^{-1}\boldsymbol{C}_{\theta,A}$
  $\overline{\lambda}(\boldsymbol{R}) \leftarrow$ `PowerMethod`$(\boldsymbol{R}, \mathcal{N}_i, M, \eta_{\text{PM}})$ ▷ PM1
  $\boldsymbol{B} = \boldsymbol{R} - \overline{\lambda}(\boldsymbol{R})I_M$
  $\overline{\lambda}(\boldsymbol{B}) \leftarrow$ `PowerMethod`$(\boldsymbol{B}, \mathcal{N}_i, M, \eta_{\text{PM}})$ ▷ PM2
  $\underline{\lambda}(\boldsymbol{R}) = |\overline{\lambda}(\boldsymbol{B}) - \overline{\lambda}(\boldsymbol{R})|$
  $\omega^\star = 2/(\overline{\lambda}(\boldsymbol{R}) + \underline{\lambda}(\boldsymbol{R}))$
end for
`DEC-NPAE`$(\mathcal{D}_i, \boldsymbol{X}, \hat{\boldsymbol{\theta}}, \boldsymbol{C}_{\theta,i}^{-1}, \mathcal{N}_i, k, M, \boldsymbol{x}_*, \Delta, \omega^\star)$

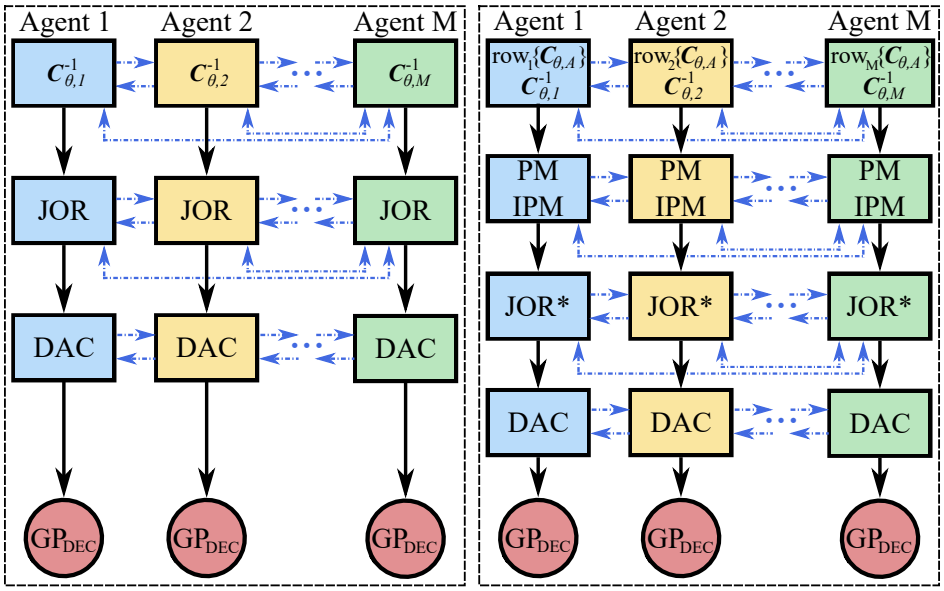
The local time and space complexity are identical to the local NPAE as shown in Table 4.2. Let $s_{\text{JOR}}^{\text{end}}$ and $s_{\text{DAC}}^{\text{end}}$ be the maximum number of iterations of the JOR and DAC to converge respectively. The total communications for a strongly complete topology yields $\mathcal{O}(2s_{\text{JOR}}^{\text{end}}M + 2s_{\text{DAC}}^{\text{end}}\text{card}(\mathcal{N}_i) + MN_i^2 + MDN_i) = \mathcal{O}(2s_{\text{JOR}}^{\text{end}}M + 2s_{\text{DAC}}^{\text{end}}\text{card}(\mathcal{N}_i) + M(N^2/M^2 + DN/M)$ for all $i \in \mathcal{V}$ as listed in Table 4.5.

The decentralized NPAE* (DEC-NPAE*) method (Figure 4.4-(b)) is similar to the DEC-NPAE, but includes an additional routine (Algorithm 12) to compute the optimal relaxation factor $\omega^\star$ (Lemma 4.29). More specifically, we employ the PM iterative scheme (4.38) to estimate the largest $\overline{\lambda}$ and smallest $\underline{\lambda}$ eigenvalues of $\boldsymbol{R}$. The workflow is as follows. To

compute the matrix of interest $\boldsymbol{R} = \text{diag}(\boldsymbol{C}_{\theta,A})^{-1}\boldsymbol{C}_{\theta,A}$, each agent i constructs $\boldsymbol{C}_{\theta,A}$ after exchanging $\{\text{row}_j\{\boldsymbol{C}_{\theta,A}\}\}_{j\neq i}$ (Algorithm 13-[Line 2]). Next, each agent i executes the PM (Algorithm 12) to obtain the maximium eigenvalue $\overline{\lambda}(\boldsymbol{R})$. Then, the spectral shift matrix $\boldsymbol{B}$ is composed (Algorithm 13-[line 6]). Using $\boldsymbol{B}$ as an input to the PM algorithm, its maximum eigenvalue is obtained $\overline{\lambda}(\boldsymbol{B})$. To this end, the minimum eigenvalue of $\boldsymbol{R}$ can be computed (Algorithm 12-[Line 8]). Subsequently, the optimal relaxation $\omega^\star$ is computed according to Lemma 4.29. Provided $\omega^\star$, the DEC-NPAE (Algorithm 11) is executed. Let $s_{\text{PM}}^{\text{end}}$ be the iterations required for the PM to converge. Then, the total communications are $\mathcal{O}(2s_{\text{PM}}^{\text{end}}M + M^2)+\mathcal{O}(\text{DEC-NPAE})$ to exchange: i) the $g_i^{(s)}$ for two PM routines (Algorithm 12-[Line 4]); ii) the $\text{row}_i\{\boldsymbol{C}_{\theta,A}\}$ (Algorithm 13-[Line 2]); and iii) the quantities of DEC-NPAE. A comparison of the communication complexity for all decentralized GP aggregation methods is presented in Table 4.5. In Figure 4.4 we illustrate the structure of the DEC-NPAE family. Both methods of the DEC-NPAE family address Problem 5.

Proposition 4.35. *Let the graph $\mathcal{G}$ be strongly complete during the JOR and PM iterations (Assumption 4.33), and strongly connected during the DAC iterations (Assumption 4.1). In addition, let Assumption 4.3, 4.5, 4.25 hold throughout the approximation. If $\omega < 2/M$, $\epsilon \in (0, 1/\Delta)$, then the DEC-NPAE is consistent for any initialization. Provided that the conditions for JOR hold for the PM iterations and that $\omega^\star = 2/(\overline{\lambda}(\boldsymbol{R}) + \underline{\lambda}(\boldsymbol{R}))$, then the DEC-NPAE* is consistent for any initial conditions.*

Proof. The proof for DEC-NPAE is a direct consequence of Proposition 4.15 and Proposition 4.26, 4.28. Similarly for DEC-NPAE*, the proof follows from Proposition 4.15 and Proposition 4.26, 4.29.



(a) DEC-NPAE (b) DEC-NPAE*

Figure 4.4: The structure of the DEC-NPAE family. Blue dotted lines correspond to communication (strongly complete). (a) DEC-NPAE incorporates Jacobi over-relaxation (JOR) and discrete-time average consensus (DAC). (b) DEC-NPAE* makes use of the power method (PM) to obtain the optimal relaxation factor and execute JOR*, and DAC.

4.4.2 Nearest Neighbor Decentralized Aggregation Methods

DALE: An alternative method to solve a linear system of algebraic equations, but for strongly connected (Assumption 4.1) and not strongly complete topology (Assumption 4.33) is DALE. The latter is an iterative method with identical setup to JOR $\boldsymbol{H}\boldsymbol{q} = \boldsymbol{b}$, where $\boldsymbol{H}$ is a known matrix, $\boldsymbol{b}$ a known vector, and $\boldsymbol{q}$ an unknown vector. The i-th node knows: i) i-th row of $\boldsymbol{H}_i = \text{row}_i\{\boldsymbol{H}\} \in \mathbb{R}^{1\times M}$; and ii) i-th entry of $b_i \in \mathbb{R}$. In addition, DALE is formulated as a consensus problem, where the goal for all agents is to obtain the same solution $\boldsymbol{q}_i \in \mathbb{R}^M$ and not just an element of the unknown vector as in JOR. The DALE follows,

$$\boldsymbol{q}_i^{(s+1)} = \boldsymbol{H}_i^\intercal(\boldsymbol{H}_i\boldsymbol{H}_i^\intercal)^{-1}b_i + \frac{1}{\text{card}(\mathcal{N}_i(t))}\boldsymbol{P}_i \sum_{j\in\mathcal{N}_i(t)} \boldsymbol{q}_j^{(s)}, \tag{4.39}$$

where $\boldsymbol{P}_i = I_M - \boldsymbol{H}_i^\intercal(\boldsymbol{H}_i\boldsymbol{H}_i^\intercal)^{-1}\boldsymbol{H}_i \in \mathbb{R}^{M\times M}$ is the orthogonal projection onto the kernel of $\boldsymbol{H}_i$. In addition, DALE can be executed in a time-varying network under Assumption 4.1.

Assumption 4.36. *Matrix $\boldsymbol{H}$ is full row rank.*

Lemma 4.37. *[78, Theorem 3] Let Assumption 4.1, 4.36 hold. There exists a constant $\phi \in (0,1)$ such that all $\boldsymbol{q}_i^{(s)}$ converge to the solution for any initialization $\boldsymbol{q}_i^{(0)}$ with worst case convergence speed ϕ^s.*

Remark 4.38. The convergence speed constant ϕ depends on the number of robots M and the diameter of the graph $\text{diam}(\mathcal{G})$. The larger the fleet size and the diameter the slower the convergence.

Remark 4.39. The i-th node using DALE (4.39) exchanges information only with its neighbors $j \in \mathcal{N}_i$ and not with the whole network (see in contrast Proposition 4.27 for JOR). In addition, DALE is concurrently a consensus algorithm and updates the whole vector $\boldsymbol{q}_i^{(s)} \in \mathbb{R}^M$, while JOR updates just the corresponding entry $[\boldsymbol{q}_i^{(s)}]_i \in \mathbb{R}$. Thus, DALE is equivalent to the operation of both JOR and DAC.

CBNN: To identify statistically correlated agents for a location of interest $\boldsymbol{x}_*$ we introduce the covariance-based nearest neighbor (CBNN) method. Let every agent i to have its own opinion for the location of interest $\{\mu_1, \ldots, \mu_M\}$, where $\mu_i = \mathrm{E}[y(\boldsymbol{x}_*) \mid \mathcal{D}_i, \boldsymbol{\theta}]$ computed as a GP local mean (4.11). In other words, every agent makes a prediction μ_i for the location of interest $\boldsymbol{x}_*$ based on its local dataset $\mathcal{D}_i$. Then, we use the local mean values to form the *mean dataset* $\mathcal{D}_\mu = (\{\boldsymbol{X}_i\}_{i=1}^M, \{\mu_i\}_{i=1}^M) = (\boldsymbol{X}, \boldsymbol{\mu})$, where $\boldsymbol{X}_i \in \mathbb{R}^{D\times N_i}$, $\boldsymbol{X} \in \mathbb{R}^{D\times N}$, $\mu_i \in \mathbb{R}$, and $\boldsymbol{\mu} \in \mathbb{R}^M$.

Definition 4.40. Let the vector of random variables $(\mu_1(\boldsymbol{x}_*), \ldots, \mu_M(\boldsymbol{x}_*), y(\boldsymbol{x}_*))^\intercal \in \mathbb{R}^{M+1}$ to form a random process, where the first two moments exist with zero mean $\mu_\mu = 0$ and a finite covariance $\boldsymbol{C}_{\theta,\mu}$.

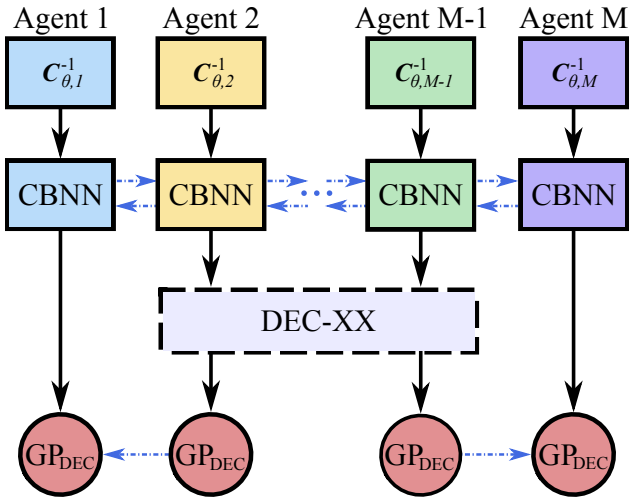


Figure 4.5: The structure of the proposed nearest neighbor decentralized aggregation methods. Blue dotted lines correspond to communication (strongly connected). The covariance-based nearest neighbor (CBNN) method identifies statistically correlated agents—in this illustration the CBNN set is $\mathcal{V}_{\text{NN}} \in [2, M-1]$. Next, a decentralized aggregation method among the DEC-PoE and DEC-BCM families is executed within the $\mathcal{V}_{\text{NN}}$ nodes. After convergence, the predicted values are communicated to the rest agents of the network.

Algorithm 14 DEC-NN-PoE

Input: $\mathcal{D}_i(\boldsymbol{X}_i, \boldsymbol{y}_i)$, $\hat{\boldsymbol{\theta}}$, $\boldsymbol{C}_{\theta,i}^{-1}$, $\mathcal{N}_i$, k, M, $\boldsymbol{x}_*$, Δ, η_{NN}
Output: $\mu_{\text{DEC-NN-PoE}}$, $\sigma^{-2}_{\text{DEC-NN-PoE}}$

for each $i \in \mathcal{V}$ **do**
  $[\boldsymbol{k}_{\mu,*}]_i \leftarrow \texttt{CrossCovCBNN}(\boldsymbol{x}_*, k, \hat{\boldsymbol{\theta}}, \boldsymbol{X}_i, \boldsymbol{C}_{\theta,i}^{-1})$ (4.40)
  for each $j \in \mathcal{N}_i$ **do**
    if $[\boldsymbol{k}_{\mu,*}]_j < \eta_{\text{NN}}$ **then**
      $\mathcal{N}_{\text{NN},i} = \mathcal{N}_i \backslash j$
      communicate j to all agents in $\mathcal{V} \backslash i$
      $\mathcal{V}_{\text{NN}} = \mathcal{V} \backslash j$
    end if
  end for
  $M_{\text{NN}} = \text{card}(\mathcal{V}_{\text{NN}})$
end for
$\texttt{DEC-PoE}(\mathcal{D}_i, \hat{\boldsymbol{\theta}}, \boldsymbol{C}_{\theta,i}^{-1}, \mathcal{N}_{\text{NN},i}, k, M_{\text{NN}}, \boldsymbol{x}_*, \Delta)$
communicate $\mu_{\text{DEC-NN-PoE}}$ and $\sigma^2_{\text{DEC-NN-PoE}}$ to agents in $\mathcal{V} \backslash \mathcal{V}_{\text{NN}}$

Proposition 4.41. *[5, Proposition 3] The random process (Definition 4.40) approximates a GP,* $(\mu_1(\boldsymbol{x}_*), \ldots, \mu_M(\boldsymbol{x}_*), y(\boldsymbol{x}_*))^\intercal \sim \mathcal{GP}(\mu_\mu, \boldsymbol{C}_{\theta,\mu})$ *as* $N \to \infty$.

Algorithm 15 DEC-NN-gPoE

Input: $\mathcal{D}_i(\boldsymbol{X}_i, \boldsymbol{y}_i)$, $\hat{\boldsymbol{\theta}}$, $\boldsymbol{C}_{\theta,i}^{-1}$, $\mathcal{N}_i$, k, M, $\boldsymbol{x}_*$, Δ, η_{NN}
Output: $\mu_{\text{DEC-gPoE}}$, $\sigma^{-2}_{\text{DEC-gPoE}}$
1: Identical to Algorithm 14 with routine `DEC-PoE` replaced by `DEC-gPoE`

The covariance of the new GP (Proposition 4.41) yields,

$$\boldsymbol{C}_{\theta,\mu} = \text{Cov}[\boldsymbol{\mu}(\boldsymbol{x}_*), y(\boldsymbol{x}_*)] = \begin{bmatrix} \boldsymbol{K}_\mu & \boldsymbol{k}_{\mu,*}^\intercal \\ \boldsymbol{k}_{\mu,*} & k_{**} \end{bmatrix},$$

where $\boldsymbol{k}_{\mu,*}^\intercal \in \mathbb{R}^M$ is the cross-covariance. Interestingly, the cross-covariance elements $[\boldsymbol{k}_{\mu,*}]_i \in \mathbb{R}_{\geq 0}$ represent the correlation of a local dataset $\mathcal{D}_i$ with the location of interest $\boldsymbol{x}_*$. Essentially, this means that when the corresponding entry tends to zero $\{\boldsymbol{k}_{\mu,*}\}_i \to 0$, then agent i is statistically uncorrelated to the location of interest $\boldsymbol{x}_*$. Every agent i can compute locally its cross-covariance element as,

$$[\boldsymbol{k}_{\mu,*}]_i = \boldsymbol{k}_{i,*}^\intercal \boldsymbol{C}_{\theta,i}^{-1} \boldsymbol{k}_{i,*}, \tag{4.40}$$

where $\boldsymbol{k}_{i,*} = k(\boldsymbol{X}_i, \boldsymbol{x}_*)$. The workflow of CBNN is as follows. Every agent i computes its cross-covariance $[\boldsymbol{k}_{\mu,*}]_i$ (4.40). When the correlation of agent i to the location of interest is below a threshold $[\boldsymbol{k}_{\mu,*}]_i < \eta_{\text{NN}}$, then the agent does not take place to the aggregation of GP experts. In other words, the agent is not allowed to have an opinion for $\boldsymbol{x}_*$. After all agents compute their correlation, the nearest neighbor subset of nodes is derived $\mathcal{V}_{\text{NN}} \subseteq \mathcal{V}$ with $M_{\text{NN}} = \text{card}(\mathcal{V}_{\text{NN}}) \leq M$.

Lemma 4.42. *The exclusion of agents from the aggregation using CBNN preserves network connectivity.*

Proof. The proof is provided in the Appendix.

Algorithm 16 DEC-NN-BCM

Input: $\mathcal{D}_i(\boldsymbol{X}_i, \boldsymbol{y}_i)$, $\hat{\boldsymbol{\theta}}$, $\boldsymbol{C}_{\theta,i}^{-1}$, $\mathcal{N}_i$, k, M, $\boldsymbol{x}_*$, Δ, η_{NN}
Output: $\mu_{\text{DEC-BCM}}$, $\sigma^{-2}_{\text{DEC-BCM}}$
1: Identical to Algorithm 14 with routine `DEC-PoE` replaced by `DEC-BCM`

Algorithm 17 DEC-NN-rBCM

Input: $\mathcal{D}_i(\boldsymbol{X}_i, \boldsymbol{y}_i)$, $\hat{\boldsymbol{\theta}}$, $\boldsymbol{C}_{\theta,i}^{-1}$, $\mathcal{N}_i$, k, M, $\boldsymbol{x}_*$, Δ, η_{NN}
Output: $\mu_{\text{DEC-rBCM}}$, $\sigma^{-2}_{\text{DEC-rBCM}}$
1: Identical to Algorithm 14 with routine `DEC-PoE` replaced by `DEC-rBCM`

The advantages of using CBNN to identify statistically correlated agents are: i) the selection of nearest neighbors is justified through a covariance not just by using an arbitrary radius; ii) only the local dataset $\mathcal{D}_i$ is required to compute (4.40) with no data exchange, which satisfies Assumption 4.2; iii) the total communications are reduced, as a subset of the agents takes part to the aggregation $\mathcal{V}_{\text{NN}}$; iv) the DAC converges faster (Lemma 4.26); and v) the DALE can be employed as $\boldsymbol{H}$ is ensured to be full row rank.

DEC-NN-PoE Family

The decentralized nearest neighbor PoE (DEC-NN-PoE) family is identical to the DEC-PoE family with a CBNN selection as shown in Figure 4.5. The implementation details for DEC-NN-PoE are given in Algorithm 14 and for DEC-NN-gPoE in Algorithm 15. The workflow is as follows. Every agent i computes the local cross-covariance of CBNN $[\boldsymbol{k}_{\mu,*}]_i$ (4.40) and evaluates its involvement to the aggregation (Algorithm 14-[Line 4]). After the CBNN terminates, the remaining agents $\mathcal{V}_{\text{NN}}$ run the DEC-PoE family routines (Algorithm 6, 7). Finally, the predicted values are transmitted to the agents that did not take part to the aggregation $\mathcal{V}\backslash\mathcal{V}_{\text{NN}}$. The time and space computational complexity is identical to the local PoE family (Table 4.2). The communication complexity for both methods is $\mathcal{O}(2s^{\text{end}}_{\text{DAC}}\text{card}(\mathcal{N}_{\text{NN},i}))$. Both methods address Problem 4.

Algorithm 18 DEC-NN-grBCM

Input: $\mathcal{D}_{+i}(\boldsymbol{X}_{+i}, \boldsymbol{y}_{+i})$, $\hat{\boldsymbol{\theta}}$, $\boldsymbol{C}^{-1}_{\theta,+i}$, $\mathcal{N}_i$, k, M, $\boldsymbol{x}_*$, Δ, η_{NN}

Output: $\mu_{\text{DEC-NN-grBCM}}$, $\sigma^{-2}_{\text{DEC-NN-grBCM}}$

1: Identical to Algorithm 14 with routine `DEC-PoE` replaced by `DEC-grBCM`

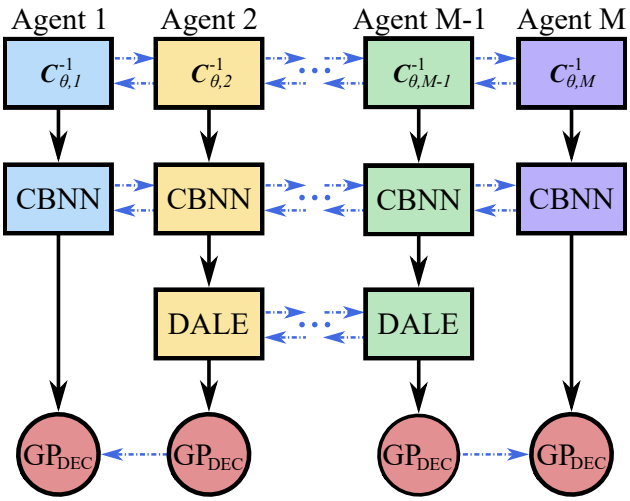


Figure 4.6: The structure of the proposed nearest neighbor decentralized aggregation methods. Blue dotted lines correspond to communication (strongly connected). The covariance-based nearest neighbor (CBNN) method identifies statistically correlated agents—in this illustration the CBNN set is $\mathcal{V}_{\text{NN}} \in [2, M-1]$. Next, a distributed algorithm for solving a linear system of equations (DALE) is executed within the $\mathcal{V}_{\text{NN}}$ nodes. After convergence, the predicted values are communicated to the rest agents of the network.

DEC-NN-BCM Family

The decentralized nearest neighbor BCM (DEC-NN-BCM) family is identical to the DEC-BCM family with a CBNN selection (Figure 4.5). The implementation details for DEC-NN-BCM are given in Algorithm 16, for DEC-NN-rBCM in Algorithm 17, and for DEC-NN-grBCM in Algorithm 18. The time and space complexity is identical to the local complexity of the DEC-BCM family (Table 4.2). The communication complexity for DEC-NN-BCM and DEC-NN-rBCM is $\mathcal{O}(2s^{\text{end}}_{\text{DAC}}\text{card}(\mathcal{N}_{\text{NN},i}))$, while for DEC-NN-grBCM is $\mathcal{O}(3s^{\text{end}}_{\text{DAC}}\text{card}(\mathcal{N}_{\text{NN},i}))$. The DEC-NN-BCM and DEC-NN-rBCM methods address Problem 4, while DEC-NN-grBCM addresses Problem 5.

Algorithm 19 DEC-NN-NPAE

Input: $\mathcal{D}_i(\boldsymbol{X}_i, \boldsymbol{y}_i)$, $\boldsymbol{X}$, $\hat{\boldsymbol{\theta}}$, $\boldsymbol{C}_{\theta,i}^{-1}$, $\mathcal{N}_i$, k, M, $\boldsymbol{x}_*$, Δ, η_{NN}
Output: $\mu_{\text{DEC-NN-NPAE}}$, $\sigma^2_{\text{DEC-NN-NPAE}}$

for each $i \in \mathcal{V}$ **do**
  $[\boldsymbol{k}_A]_i \leftarrow \texttt{crossCov}(\boldsymbol{x}_*, k, \hat{\boldsymbol{\theta}}, \boldsymbol{X}_i, \boldsymbol{C}_{\theta,i}^{-1})$ (4.19)
  $[\boldsymbol{k}_{\mu,*}]_i \leftarrow \texttt{CrossCovCBNN}(\boldsymbol{x}_*, k, \hat{\boldsymbol{\theta}}, \boldsymbol{X}_i, \boldsymbol{C}_{\theta,i}^{-1})$ (4.40)
  for each $j \in \mathcal{N}_i$ **do**
    if $[\boldsymbol{k}_{\mu,*}]_j < \eta_{\text{NN}}$ **then**
      $\mathcal{N}_{\text{NN},i} = \mathcal{N}_i \backslash j$; $\mathcal{V}_{\text{NN}} = \mathcal{V} \backslash j$
      communicate j to all agents in $\mathcal{V}_{\text{NN}} \backslash i$
    else
      communicate $[\boldsymbol{k}_A]_j$ to all agents in $\mathcal{V}_{\text{NN}} \backslash i$
    end if
  end for
end for
for each $i \in \mathcal{V}_{\text{NN}}$ **do**
  $\mu_i \leftarrow \texttt{localMean}(\boldsymbol{x}_*, k, \hat{\boldsymbol{\theta}}, \mathcal{D}_i, \boldsymbol{C}_{\theta,i}^{-1})$ (4.11)
  communicate $\boldsymbol{C}_{\theta,i}^{-1}$, $\boldsymbol{X}_i$ to agents in $\mathcal{V}_{\text{NN}} \backslash i$
  $\boldsymbol{k}_{\text{NN},A} = [\boldsymbol{k}_A]_i \cup \{[\boldsymbol{k}_A]_j\}_{j \in \mathcal{V}_{\text{NN}}}$
  $\text{row}_{\text{NN},i}\{\boldsymbol{C}_{\theta,A}\} \leftarrow \texttt{localCov}(\boldsymbol{x}_*, k, \boldsymbol{X}, \hat{\boldsymbol{\theta}}, \mathcal{V}_{\text{NN}})$ (4.20)
  $\boldsymbol{H}_i = \text{row}_{\text{NN},i}\{\boldsymbol{C}_{\theta,A}\}$; $M_{\text{NN}} = \text{card}(\mathcal{V}_{\text{NN}})$
  $b_{\mu,i} = \mu_{\text{NN},i}$; $\boldsymbol{b}_{\sigma^2} = \boldsymbol{k}_{\text{NN},A}$
  $\boldsymbol{P}_i = I_{M_{\text{NN}}} - \boldsymbol{H}_i^\intercal (\boldsymbol{H}_i \boldsymbol{H}_i^\intercal)^{-1} \boldsymbol{H}_i$
  initialize $\boldsymbol{q}_{\mu,i}^{(0)} = b_{\mu,i} \oslash \boldsymbol{H}_i$; $\boldsymbol{q}_{\sigma^2,i}^{(0)} = \boldsymbol{b}_{\sigma^2} \oslash \boldsymbol{H}_i$
  repeat ▷ 2×DALE
    communicate $\boldsymbol{q}_{\mu,i}^{(s)}$, $\boldsymbol{q}_{\sigma^2,i}^{(s)}$ to neighbors $\mathcal{N}_{\text{NN},i}$
    $\boldsymbol{q}_{\mu,i}^{(s+1)} \leftarrow \texttt{DALE}(\boldsymbol{P}_i, \boldsymbol{H}_i, b_{\mu,i}, \{\boldsymbol{q}_{\mu,j}^{(s)}\}_{j \in \mathcal{N}_{\text{NN},i}}, \mathcal{N}_{\text{NN},i})$ (4.39)
    $\boldsymbol{q}_{\sigma^2,i}^{(s+1)} \leftarrow \texttt{DALE}(\boldsymbol{P}_i, \boldsymbol{H}_i, b_{\sigma^2,i}, \{\boldsymbol{q}_{\sigma^2,j}^{(s)}\}_{j \in \mathcal{N}_{\text{NN},i}}, \mathcal{N}_{\text{NN},i})$ (4.39)
  until maximin stopping criterion
  $\mu_{\text{DEC-NN-NPAE}} = \boldsymbol{k}_{\text{NN},A}^\intercal \boldsymbol{q}_{\mu,i}^{\text{end}}$
  $\sigma^2_{\text{DEC-NN-NPAE}} = \sigma_f^2 (k(\boldsymbol{x}_*, \boldsymbol{x}_*) - \boldsymbol{k}_{\text{NN},A}^\intercal \boldsymbol{q}_{\sigma^2,i}^{\text{end}})$
end for

Proposition 4.43. *Let the Assumption 4.1, 4.3, 4.5, 4.11, 4.25 hold throughout the approximation. If $\omega < 2/M$ then the DEC-NN-grBCM is consistent for any initialization.*

Proof. The proof is a direct consequence of Proposition 4.14 and Proposition 4.26, 4.42.

DEC-NN-NPAE

We introduce the decentralized nearest neighbor NPAE (DEC-NN-NPAE) method to distribute the computations (4.21), (4.22) of NPAE (Figure 4.6). The DEC-NN-NPAE employs the CBNN and DALE (4.39) methods. By using CBNN, one can satisfy Assumption 4.36 and use the DALE. Thus, the DEC-NN-NPAE relaxes the strongly complete topology (Assumption 4.33) to a time-invariant strongly connected topology (Assumption 4.1) as discussed in Remark 4.39. Implementation details are given in Algorithm 19. The workflow is as follows. First, each entity computes: i) the local cross covariance $[\boldsymbol{k}_A]_i$ (4.19); and ii) the cross-covariance of CBNN $[\boldsymbol{k}_{\mu,*}]_i$ (4.40). Next, we execute the CBNN routine to select the nearest neighbors. During the CBNN, if a criterion is met for an agent j to stay in idle (Algorithm 19-[Line 5]), it is removed from the list of agents $\mathcal{V}_{\text{NN}} = \mathcal{V} \backslash j$; and if not, the corresponding element of the local cross covariance $[\boldsymbol{k}_A]_j$ is communicated to all other agents $\mathcal{V}_{\text{NN}} \backslash i$. When the CBNN routine terminates, we execute the DALE method on the nearest neighbors $\mathcal{V}_{\text{NN}}$. Similarly to DEC-NPAE, the inputs $\{\boldsymbol{X}_j\}_{j \neq i}$ and the local inverted covariance matrices $\{\boldsymbol{C}_{\theta,j}^{-1}\}_{j \neq i}$ are communicated between CBNN agents. Next, we execute two parallel DALE algorithms with known matrix $\boldsymbol{H} = \boldsymbol{C}_{\theta,A}$ and known vectors: i) $\boldsymbol{b} = \boldsymbol{\mu}$; and ii) $\boldsymbol{b} = \boldsymbol{k}_A$. The first DALE is associated with the prediction mean $\mu_{\text{DEC-NN-NPAE}}$ (Algorithm 19-[Line 24]) and the second with the variance $\sigma^2_{\text{DEC-NN-NPAE}}$ (Algorithm 19-[Line 25]). After every DALE iteration, each agent i communicates the computed vectors $\boldsymbol{q}_{\mu,i}^{(s)}$, $\boldsymbol{q}_{\sigma^2,i}^{(s)}$ to its neighbors $\mathcal{N}_{\text{NN},i}$ (Algorithm 19-[Line 23]). Next, we update the vectors $\boldsymbol{q}_{\mu,i}$, $\boldsymbol{q}_{\sigma^2,i}$ (Algorithm 11-[Lines 24, 25]) with the DALE method. When both DALE converge, each agent follows (4.21), (4.22) to recover the DEC-NN-NPAE mean and variance. The local time and space complexity are identical to the local NPAE as shown in Table 4.2. Let $s^{\text{end}}_{\text{DALE}}$ be the maximum number of iterations of DALE to converge. The total communications during the CBNN yields $\mathcal{O}(M_{\text{NN}})$ and during DALE $\mathcal{O}(2s^{\text{end}}_{\text{DALE}}\text{card}(\mathcal{N}_{\text{NN},i}) + M_{\text{NN}}N_i^2 + M_{\text{NN}}DN_i) =$

$\mathcal{O}(2s^{\text{end}}_{\text{DALE}}\text{card}(\mathcal{N}_{\text{NN},i}) + M_{\text{NN}}(N^2/M^2_{\text{NN}} + DN/M_{\text{NN}}))$ for all $i \in \mathcal{V}_{\text{NN}}$. DEC-NN-NPAE addresses Problem 5.

Proposition 4.44. *Let Assumption 4.1, 4.3, 4.5, 4.25 hold throughout the approximation. Then, the DEC-NN-NPAE is consistent for any initialization of DALE.*

Proof. The proof is a direct consequence of Proposition 4.15 and Proposition 4.37, 4.42

4.5 Numerical Experiments

We perform numerical experiments to illustrate the efficiency of the proposed methods. Synthetic data with known hyper-parameters values are employed to evaluate the GP training methods in four aspects: i) hyper-parameter estimation accuracy; ii) computation time per agent; iii) communications per agent; and iv) comparison with centralized GP training techniques. A real-world dataset of sea surface temperature (SST) [22, 59] and the kin40k dataset [17, 30, 76, 90, 113, 123] are used to assess the GP prediction algorithms in four aspects: i) prediction accuracy; ii) uncertainty quantification; iii) communications per agent; and iv) comparison with aggregation of GP experts methods. All numerical experiments are conducted in MATLAB using the GPML package [103] on an Intel Core i7-6700 CPU @3.40 GHz with 32.0 GB memory RAM. Demonstration code can be found at: github.com/gkontoudis/decentralized-GP.

4.5.1 Decentralized GP Training

We generate two sets of data with total size $N = 8,100$ and $N = 32,400$ using the observation model (4.1) and the separable squared exponential covariance function (4.2) with hyper-parameter values $\boldsymbol{\theta} = (l_1, l_2, \sigma_f, \sigma_\epsilon)^\intercal = (1.2, 0.3, 1.3, 0.1)^\intercal$. For every set of random

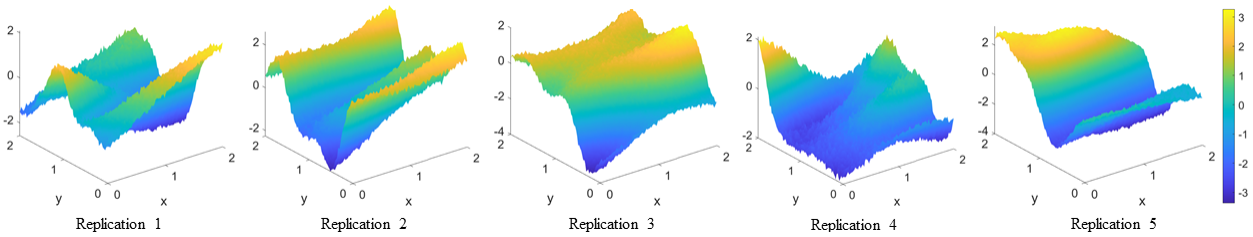


Figure 4.7: Five replications of the synthetic GP with known hyper-parameter values $\boldsymbol{\theta} = (1.2, 0.3, 1.3, 0.1)^\intercal$ for $N = 8,100$ data.

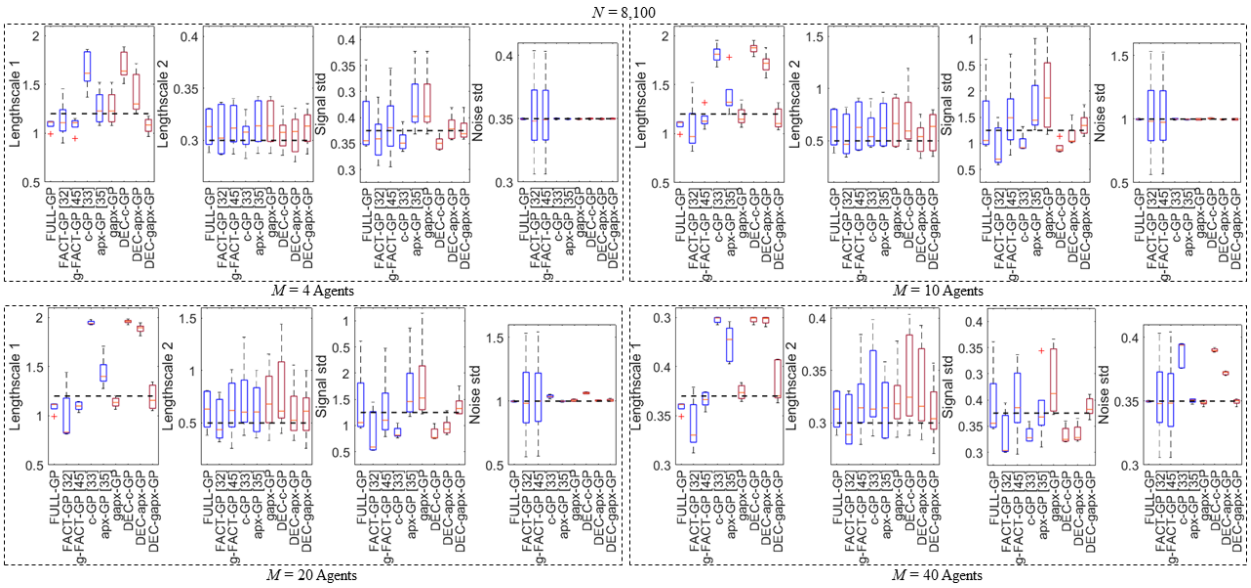


Figure 4.8: Accuracy of GP hyper-parameter training using $N = 8,100$ data for four fleet sizes and 10 replications. The true values are demonstrated with a black dotted line. The existing GP training methods are shown in blue boxes (FULLGP, FACT-GP [30], g-FACT-GP [76], c-GP [136], apx-GP [135]) and the proposed in maroon boxes (gapx-GP, DEC-c-GP, DEC-apx-GP, and DEC-gapx-GP).

functions we perform 10 replications to avoid random assignment of data. An example of five replications for $N = 8,100$ data is presented in Figure 4.7. Note that the smaller the length-scale l, the more wiggly is the random function. Since $l_2 < l_1$, the profile of the produced random functions is more uneven along the y-axis rather than the x-axis. Next, we equally partition the space of interest $\mathbb{S} = [0, 2]^2$ along the x-axis according to fleet sizes $M = \{4, 10, 20, 40\}$, and assign local datasets that lie in the corresponding local space, e.g., for $M = 10$ agents see Figure 4.10-(b). We compare the centralized GP training methods

Table 4.6: Time & Communication Rounds of GP Training Methods

M	Method	$N = 8,100$		$N = 32,400$	
		Time [s]	Comms s^{end}	Time [s]	Comms s^{end}
	FULL-GP	2,114.2	-	-	-
4	FACT-GP [30]	75.9	186.0	2,361.9	196.0
	g-FACT-GP [76]	332.1	160.0	¿3,000	-
	c-GP [136]	404.1	141.4	-	-
	apx-GP [135]	**26.8**	43.6	**817.6**	45.2
	gapx-GP	67.3	**39.7**	2,074.2	**42.1**
	DEC-c-GP	414.1	100	-	-
	DEC-apx-GP	**61.9**	100	**1,821.3**	100
	DEC-gapx-GP	328.1	100	¿3,000	-
10	FACT-GP [30]	9.8	179.6	228.2	194.2
	g-FACT-GP [76]	31.8	131.8	1,035.6	155.2
	c-GP [136]	92.1	193.8	-	-
	apx-GP [135]	**3.8**	47.8	**88.8**	46.8
	gapx-GP	15.1	**42.2**	522.2	**44.3**
	DEC-c-GP	82.4	100	-	-
	DEC-apx-GP	**8.4**	100	**188.8**	100
	DEC-gapx-GP	38.5	100	1,123.4	100
20	FACT-GP [30]	2.6	172.6	46.6	226.2
	g-FACT-GP [76]	7.0	127.2	199.4	167.6
	c-GP [136]	31.4	127.8	-	-
	apx-GP [135]	**1.3**	56.2	**18.3**	49.8
	gapx-GP	4.1	**50.6**	85.8	**45.6**
	DEC-c-GP	30.4	100	-	-
	DEC-apx-GP	**2.2**	100	**36.9**	100
	DEC-gapx-GP	8.1	100	185.8	100
40	FACT-GP [30]	0.5	139.6	9.1	160.0
	g-FACT-GP [76]	1.8	112.2	30.9	128.6
	c-GP [136]	8.9	66.6	-	-
	apx-GP [135]	**0.3**	56.4	**4.6**	54.4
	gapx-GP	1.2	**51.2**	17.9	**49.2**
	DEC-c-GP	9.1	100	-	-
	DEC-apx-GP	**0.5**	100	**8.2**	100
	DEC-gapx-GP	2.5	100	36.4	100

FACT-GP [30], g-FACT-GP [76], c-GP [136], and apx-GP [135] to the proposed gapx-GP. In addition, we include in the comparison the proposed decentralized GP training methods DEC-c-GP, DEC-apx-GP, and DEC-gapx-GP. All decentralized GP training methods fol-

low a path graph topology as depicted in Figure 4.1. Thus, the maximum degree of the graph is $\Delta = 2$ and its diameter $\text{diam}(\mathcal{G}) = M - 1$. All methods start from the same initial vector value $(l_1^{(0)}, l_2^{(0)}, \sigma_f^{(0)}, \sigma_\epsilon^{(0)})^\intercal = (2, 0.5, 1, 1)^\intercal$. The penalty parameter of the augmented Lagrangian is set to $\rho = 500$, the decentralized ADMM tolerance for convergence $\text{TOL}_{\text{ADMM}} = 10^{-3}$, the positive Lipschitz constant of the approximation (4.26) $L_i = 5,000$, and the regulation positive constant of the approximation (4.32) $\kappa_i = 5,000$ for all $i \in \mathcal{V}$. For the nested optimization problem of c-GP (4.25b) and DEC-c-GP (4.31b) we use gradient descent with step size $\alpha = 10^{-5}$. All decentralized GP training methods terminate after $s^{\text{end}} = 100$ predetermined communication rounds (Remark 4.24), yielding identical communication complexity (Table 4.4). Any algorithm that takes over 3,000 s to be executed is terminated.

In Figure 4.8, we show the boxplots of the estimated hyper-parameters for all GP training methods and all fleet sizes using $N = 8,100$ data. Blue boxes illustrate existing GP training methods and maroon boxes represent the proposed GP training methods. The corresponding average computation time per agent and the communication rounds are shown in Table 4.6. Provided the communication rounds s^{end}, the communication complexity can be computed according to Table 4.1, 4.3. For the case of $M = 4$ agents, all centralized methods provide accurate hyper-parameters estimates except of the c-GP on l_1. In terms of computation time, c-GP is the more demanding method, while FACT-GP, apx-GP, and gapx-GP convergence very fast, outperforming FULL-GP two orders of magnitude for similar or even better level of accuracy. The least communication rounds are achieved by the proposed methodology gapx-GP which results in the lowest communication complexity. Regarding the decentralized methods, both DEC-apx-GP and DEC-gapx-GP produce accurate hyper-parameter estimates, while DEC-c-GP is inaccurate on l_1. DEC-apx-GP requires less computation time per agent than the other two decentralized methods. As we increase the

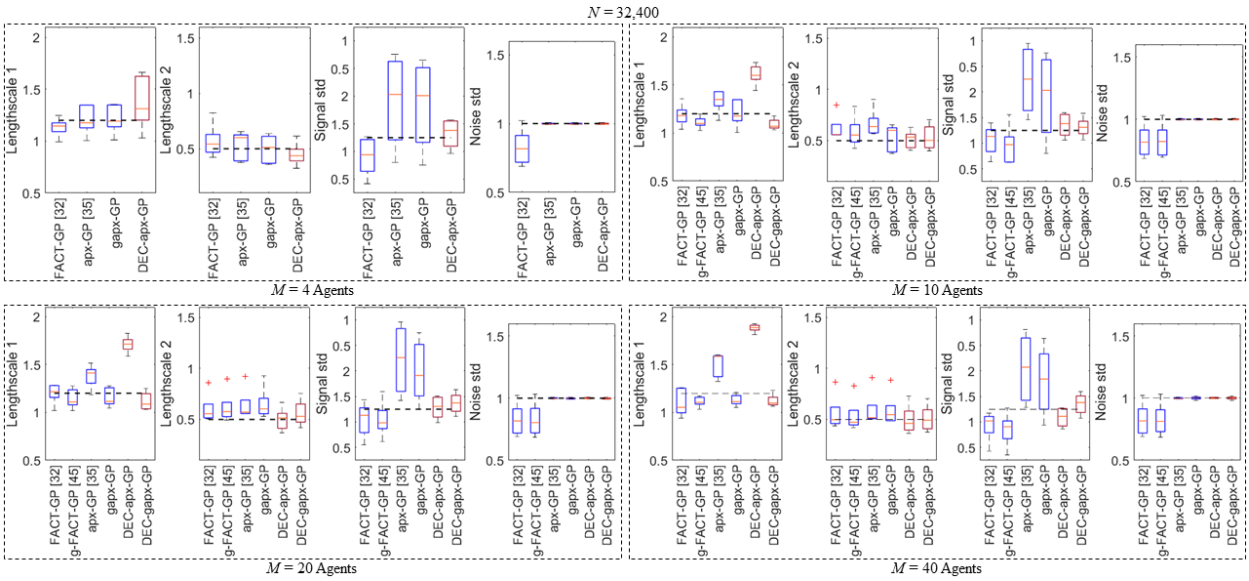


Figure 4.9: Accuracy of GP hyper-parameter training using $N = 32,400$ data for four fleet sizes and 10 replications. The true values are demonstrated with a black dotted line. The existing GP training methods are shown in blue boxes (FACT-GP [30], g-FACT-GP [76], apx-GP [135]) and the proposed in maroon boxes (DEC-apx-GP, and DEC-gapx-GP).

number of agents ($M = 10$ and $M = 20$ agents), the hyper-parameter estimation accuracy deteriorates for all centralized methods except of the proposed gapx-GP. In addition, gapx-GP results in the lowest communication complexity and in competitive computation time per agents, outperformed only by apx-GP. Regarding the decentralized GP training methods, the hyper-parameter estimation of DEC-gapx-GP is the most accurate. Both DEC-apx-GP and DEC-c-GP provide reasonable estimates for all hyper-parameters other than l_1 which is inaccurate. The lowest computation per entity is measured for DEC-apx-GP, while the most accurate method DEC-gapx-GP requires four times more computations than DEC-apx-GP. For $M = 40$ agents, the proposed gapx-GP produces the most accurate hyper-parameter estimates with only g-FACT-GP competing with reasonable accuracy. However, g-FACT-GP requires more computation time per agent and exchanges double the amount of messages to converge than the proposed gapx-GP. From the proposed decentralized methods, only DEC-gapx-GP is accurate (Remark 4.22) and requires reasonable computation per agent.

We present the boxplots of the estimated hyper-parameters using $N = 32,400$ data and for all fleet sizes in Figure 4.9, while in Table 4.6 we list the corresponding computation time per agent as well as the communication rounds. The FULL-GP, c-GP, and DEC-c-GP methods are not implemented for $N = 32,400$ data, as we expect significantly high computation time (Remark 4.20). For $M = 4$ agents, both g-FACT-GP and DEC-gapx-GP exceeded the time limit (3,000 s) for convergence. Among the feasible centralized methods for $N = 32,400$ data, apx-GP and gapx-GP are more accurate than FACT-GP. All methods are computationally expensive as each agent i is assigned with $N_i = 32,400/4 = 8,100$ data, yet apx-GP is the fastest. Regarding the decentralized methods, DEC-apx-GP is the only feasible method and produces accurate hyper-parameter estimates. As we increase the number of agents ($M = 10$ and $M = 20$ agents), the number of data is distributed to local agents, and thus g-FACT-GP and DEC-gapx-GP can be implemented. Since the number of data is high, all centralized methods produce accurate hyper-parameters estimates. Yet, apx-GP is computationally more efficient. Although the proposed gapx-GP requires more time to converge, the communication overhead is the least. Among the decentralized methods, DEC-gapx-GP is more accurate, but computationally more demanding than DEC-apx-GP. For the case of $M = 40$ agents, the most accurate centralized hyper-parameter estimator is the gapx-GP with the lowest information exchange requirements. The fastest centralized method is the apx-GP, yet its accuracy is moderate. Regarding the decentralized methods, DEC-gapx-GP remains accurate and requires reasonable computation time.

Overall, for $N = 8,100$ the proposed gapx-GP is the most accurate centralized GP training method, especially as the fleet size increases. Moreover, gapx-GP requires reasonable computations and it is the most efficient method with respect to communication. Among the proposed decentralized GP training methods, DEC-gapx-GP is the most accurate method, yet DEC-apx-GP produces competitive hyper-parameter estimates for medium and small

fleet size. DEC-apx-GP is the fastest decentralized GP training method, while DEC-gapx-GP is more demanding, yet requires reasonable computational resources. In principle, as we increase the number of agents, the computation is distributed and thus yields lower computation time per agent. Note that the hyper-parameter estimation accuracy improves as we obtain more data which leads to higher accuracy for $N = 32,400$ data (Remark 4.23). Some techniques are not scalable for the larger dataset $N = 32,400$, especially when the fleet size is small $M = 4$. However, for larger fleet size the distribution of data facilitates the execution of most methods. Among the centralized methods, apx-GP is accurate and requires significantly less computational time for small fleet size, but as we increase the number of agents the proposed gapx-GP becomes computationally more efficient and remains accurate. Similarly, DEC-apx-GP is accurate and computationally less demanding for small fleet size, but DEC-gapx-GP becomes more computationally efficient as we distribute the data to more agents.

4.5.2 Decentralized GP Prediction

We use a real-world dataset of sea surface temperature (SST) [22, 59]. We extract $122,500$ SST values from $(36.4^{\text{o}}, -73.0^{\text{o}})$ to $(40.0^{\text{o}}, -69.4^{\text{o}})$ measured in Kelvins. The area corresponds to $400\,\text{km} \times 400\,\text{km}$ of the Atlantic ocean and for demonstration is normalized over $[0,1]^2$ (Figure 4.10-(a)). Additionally, we add iid noise $\epsilon \sim \mathcal{N}(0, 0.25)$ to the observations (4.1). We use $20,000$ observations, equally distributed for four fleet sizes $M = \{4, 10, 20, 40\}$. An example of data distribution assignment for $M = 10$ agents is shown in Figure 4.10-(b). The GP training of the hyper-parameters is performed with the DEC-gapx-GP method. We employ 13 techniques over $N_{\text{t}} = 100$ prediction points: i) DEC-PoE with path graph; ii) DEC-NN-PoE with path graph; iii) DEC-gPoE with path graph; iv) DEC-NN-gPoE with path graph; v) DEC-BCM with path graph; vi) DEC-NN-BCM with path graph; vii) DEC-

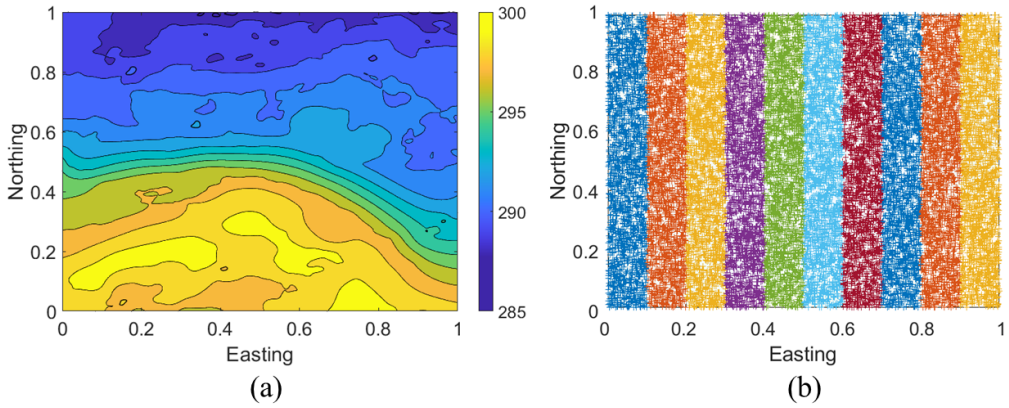


Figure 4.10: (a) SST field [59]; (b) Observations of each agent for $M = 10$.

rBCM with path graph; viii) DEC-NN-rBCM with path graph; ix) DEC-grBCM with path graph; x) DEC-NN-grBCM with path graph; xi) DEC-NPAE with strongly complete graph; xii) DEC-NPAE* with path graph and strongly complete graph; and xiii) DEC-NN-NPAE with path graph, where the graph types are shown in Figure 4.1. For every scenario we perform 15 replications to avoid random assignment of data.

The quality assessment is accomplished with two metrics. The root mean square error RMSE $= [1/N \sum_{i=1}^{N} (\mu(\boldsymbol{x}_*) - y(\boldsymbol{x}_*))^2]^{1/2}$ assesses the prediction mean. The negative log predictive density NLPD $= -1/N \sum_{i=1}^{N} \log p(\hat{\boldsymbol{y}}_* \mid \mathcal{D}, \boldsymbol{x}_*)$ characterizes the prediction mean and variance, where $p(\hat{\boldsymbol{y}}_* \mid \mathcal{D}, \boldsymbol{x}_*)$ is the predictive distribution [102].

In Figure 4.11, we show the average RMSE and NLPD values for four fleet sizes and 15 replications using the decentralized PoE-based methods. Since the proposed algorithms DEC-PoE, DEC-NN-PoE; DEC-gPoE, and DEC-NN-gPoE approximate the PoE; and gPoE respectively, the optimal RMSE and NLPD values are that of PoE [51] and gPoE [17]. All PoE-based methods produce identical RMSE accuracy, illustrating that the proposed decentralized methods converge with almost zero approximation error. Indeed, PoE and gPoE have identical mean prediction values, validating Proposition 4.10. In terms of uncertainty quantification, DEC-PoE and DEC-NN-PoE. Moreover, they all fail to report NLPD val-

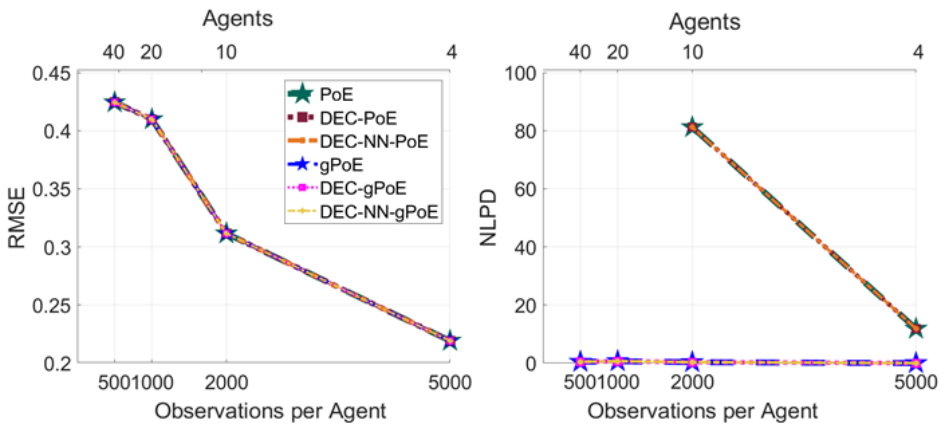


Figure 4.11: Average RMSE and NLPD values for four fleet sizes and 15 replications with the PoE-based methods on a path graph topology.

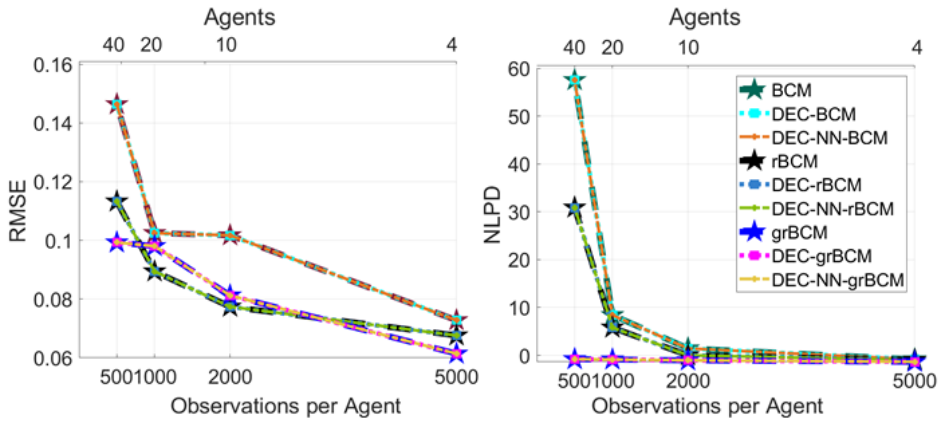


Figure 4.12: Average RMSE and NLPD values for four fleet sizes and 15 replications with the BCM-based methods on a path graph topology.

ues for larger fleet sizes ($M = 20$ and $M = 40$ agents), as the predictive variance of PoE (4.14) is additive and leads to overconfident results which subsequently yield infinite values of NLPD. Similarly, DEC-gPoE and DEC-NN-gPoE report identical NLPD values with the gPoE. Both nearest neighbor methods (DEC-NN-PoE and DEC-NN-gPoE) produce results that are indistinguishable to PoE and gPoE respectively, even though 42.5% of agents were excluded on average from the aggregation (Table 4.7).

In Figure 4.12, we present the average RMSE and NLPD values for four fleet sizes and 15

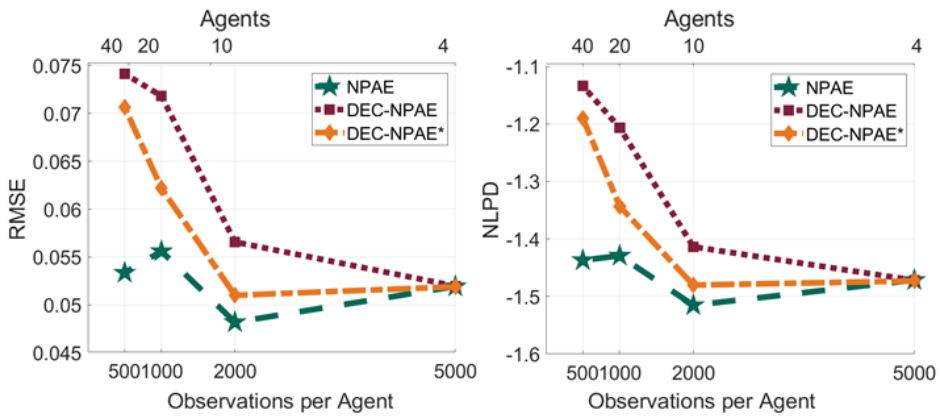


Figure 4.13: Average RMSE and NLPD values for four fleet sizes and 15 replications with the NPAE-based methods on a strongly complete topology.

replications using the decentralized BCM-based methods. Since the proposed algorithms DEC-BCM, DEC-NN-BCM; DEC-rBCM, DEC-NN-rBCM; DEC-grBCM, and DEC-NN-grBCM approximate the BCM [126]; rBCM [30]; and grBCM [76] respectively, the optimal RMSE and NLPD values are that of BCM, rBCM and grBCM. We observe that DEC-BCM and DEC-NN-BCM converge to BCM as they report identical RMSE and NLPD values. Similarly, for the rest decentralized methods, i.e., DEC-rBCM, DEC-NN-rBCM converge to rBCM, and DEC-grBCM, DEC-NN-grBCM converge to grBCM with almost zero approximation error. All nearest neighbor methods DEC-NN-BCM, DEC-NN-rBCM, and DEC-NN-grBCM make identical predictions to BCM, rBCM, and grBCM, although a subset of agents are selected to participate in the prediction.

The average RMSE and NLPD values for four fleet sizes and 15 replications using the decentralized NPAE-based methods are presented in Figure 4.13, 4.14. The difference between Figure 4.13 and Figure 4.14 is that the latter demonstrates methods in a strongly connected network topology (path graph), while the former methods in a strongly complete network topology (see Figure 4.1 for differences). Since the proposed algorithms DEC-NPAE, DEC-NPAE*, and DEC-NN-NPAE approximate the NPAE, the optimal RMSE and NLPD val-

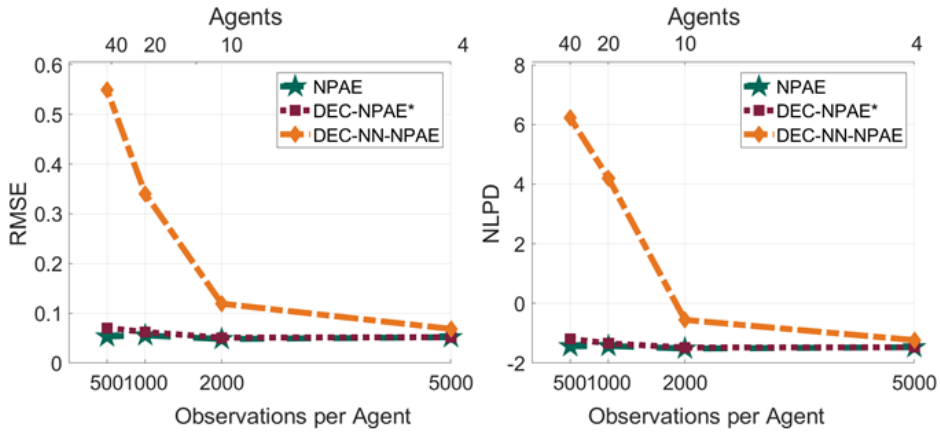


Figure 4.14: Average RMSE and NLPD values for four fleet sizes and 15 replications with the NPAE-based methods on a path graph topology.

ues are that of NPAE [107]. The main difference of DEC-NPAE and DEC-NPAE* lies in the estimation of the optimal relaxation factor rather than selecting the relaxation factor based on the fleet size (Remark 4.30). All DEC-NPAE, DEC-NPAE*, and DEC-NN-NPAE produce an approximation error, and thus they do not converge to the optimal RMSE and NLPD values of NPAE. More specifically, DEC-NPAE* has the smallest approximation error (Figure 4.13), while DEC-NN-NPAE reports high approximation error (Figure 4.14). Since $s^{\text{end}}_{\omega^\star} + s^{\text{end}}_{\text{JOR}^\star} < s^{\text{end}}_{\text{JOR}}$, the DEC-NPAE* converges faster than DEC-NPAE. This advocates that the proposed scheme to estimate the optimal relaxation factor before implementing the JOR, is more efficient both in speed and accuracy. Thus, the proposed method outperforms other approaches that employ the JOR [23, 24, 25].

In Table 4.7, we compare the average computation time for each agent and the communication rounds of all nearest neighbor methods on a path graph network topology. In addition, we compute the average number of nearest neighbors from 15 replications that participate in the prediction M_{NN} for all $N_{\text{t}} = 100$ prediction points of each fleet size. The results reveal a 42.5% agent reduction with no approximation error for DEC-NN-PoE, DEC-NN-gPoE, DEC-NN-BCM, DEC-NN-rBCM, and DEC-NN-grBCM (Figure 4.11, 4.12); and significant

Table 4.7: Decentralized CBNN Aggregation Methods

M	Method	Nearest Neighbors M_{NN}	Time per Agent [s]	Comms s^{end}
4	DEC-NN-PoE	2.3±0.1	0.0312	3.6
	DEC-NN-gPoE		0.0339	3.6
	DEC-NN-BCM		0.0323	3.6
	DEC-NN-rBCM		0.0469	3.6
	DEC-NN-grBCM		0.0516	3.6
	DEC-NN-NPAE		1.1683	69.4
10	DEC-NN-PoE	5.7±0.2	0.0122	7.2
	DEC-NN-gPoE		0.0121	7.2
	DEC-NN-BCM		0.0126	7.2
	DEC-NN-rBCM		0.0172	7.2
	DEC-NN-grBCM		0.0167	7.2
	DEC-NN-NPAE		0.4844	247.2
20	DEC-NN-PoE	11.3±0.4	0.0087	6.8
	DEC-NN-gPoE		0.0083	6.8
	DEC-NN-BCM		0.0086	6.8
	DEC-NN-rBCM		0.0126	7.0
	DEC-NN-grBCM		0.0124	7.0
	DEC-NN-NPAE		0.2698	625.6
40	DEC-NN-PoE	23.6±1.1	0.0052	7.4
	DEC-NN-gPoE		0.0049	7.4
	DEC-NN-BCM		0.0051	7.4
	DEC-NN-rBCM		0.0073	7.4
	DEC-NN-grBCM		0.0071	7.4
	DEC-NN-NPAE		2.7009	1,824.1

approximation error for DEC-NN-NPAE (Figure 4.14). The computation time per agent and the communication rounds are similar for all methods other than the DEC-NN-NPAE. Thus, DEC-NN-NPAE is insufficient in accuracy, computation time per agent, and communications, while all other decentralized nearest neighbor methods report optimal RMSE and NLPD values, scalable computation time, and require little information exchange.

In Figure 4.15, we compare five out of the 13 proposed methods that produce accurate predictions and properly quantify the uncertainty. The comparison includes the DEC-NN-

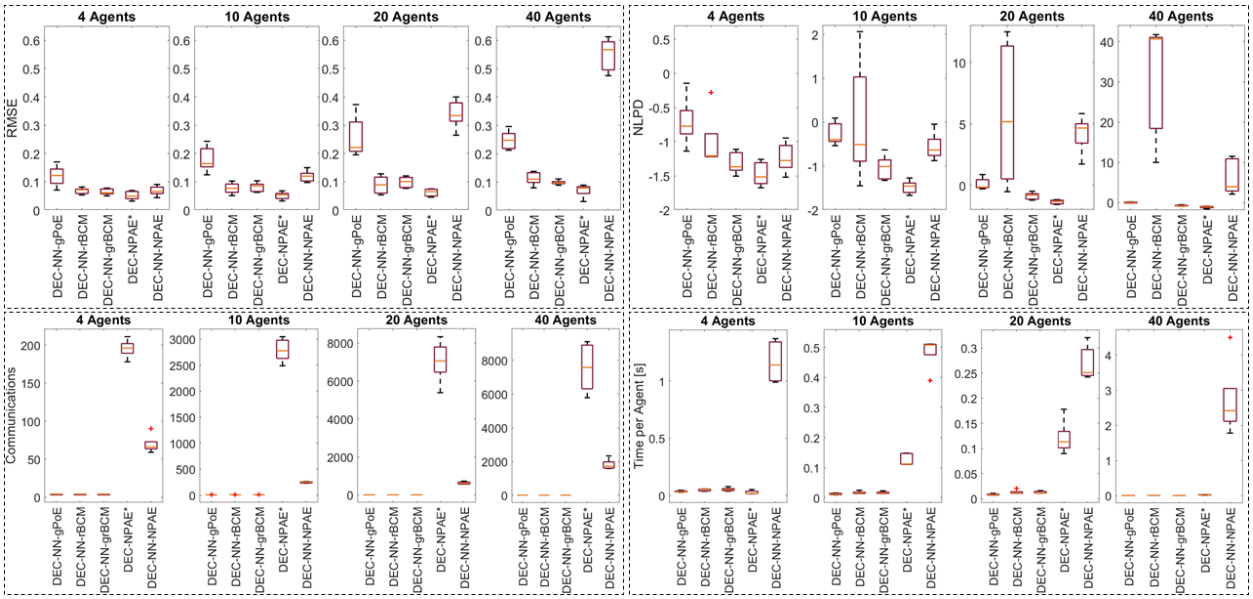


Figure 4.15: Comparison of accuracy, uncertainty quantification, communication rounds s^{end}, and computation time per agent for four fleet sizes and 15 replications on decentralized GP predictions at $N_{\text{t}} = 100$ unknown locations using $N = 20,000$ observations in a path graph network topology. Lower RMSE and NLPD values indicate better accuracy and better uncertainty quantification respectively. The comparison includes the five best decentralized GP prediction methods out of the 13 proposed methods.

gPoE, DEC-NN-rBCM, DEC-NN-grBCM, DEC-NPAE*, and DEC-NN-NPAE for all fleet sizes in a path graph topology. Accuracy is evaluated with RMSE, uncertainty quantification with NLPD, communication complexity with communication rounds, and scalability with computation time per agent. In terms of accuracy all methods perform well by producing low RMSE values for small fleet sizes. However, as we increase the fleet size only DEC-NN-rBCM, DEC-NN-grBCM, and DEC-NPAE* recover good accuracy with the later being the most accurate. Similarly for uncertainty quantification, all methods quantify satisfactorily the uncertainty by reporting low NLPD values for small fleet size (4 agents). Yet, as the fleet sizes increases only DEC-NN-gPoE, DEC-NN-grBCM, and DEC-NPAE* maintain good level of uncertainty quantification with DEC-NPAE* being the best. Inter-agent communication favors DEC-NN-gPoE, DEC-NN-rBCM, and DEC-NN-grBCM for all fleet

Table 4.8: Qualitative Assessment of Decentralized GP Methods

Method	RMSE Accuracy	NLPD UQ	Comms	Scalable
DEC-NN-gPoE	Moderate	Moderate	Excellent	Excellent
DEC-NN-rBCM	Excellent	Bad	Excellent	Excellent
DEC-NN-grBCM	Excellent	Excellent	Excellent	Excellent
DEC-NPAE*	Excellent	Excellent	Bad	Moderate
DEC-NN-NPAE	Bad	Moderate	Moderate	Bad

sizes. Notably the most accurate method both in terms of RMSE and NLPD (DEC-NPAE*) requires signification information exchange to converge. In terms of scalability, DEC-NN-gPoE, DEC-NN-rBCM, and DEC-NN-grBCM are executed very fast, while DEC-NPAE* requires reasonable computations.

A qualitative assessment of the comparison in Figure 4.15 for all four aspects is presented on Table 4.8. The results reveal that DEC-NN-grBCM is overall the best decentralized GP prediction method; DEC-NPAE* is an accurate method, quantifies the uncertianty well, and entails reasonable computations, but requires signification communication; and DEC-NN-rBCM is accurate, scalable, requires little information exchange, but quantifies the uncertainty poorly.

4.6 Conclusion

This chapter proposes decentralized methods that cover a broad spectrum of multi-agent learning applications as they can be employed both for decentralized GP training and decentralized GP prediction on various fleet sizes with different computation and communication capabilities of local agents. We utilize distributed optimization methods of ADMM to perform accurate and scalable GP training in networks. More specifically, a closed-form solution of the decentralize ADMM is derived for the case of GP hyper-parameter training with

maximum likelihood estimation. DEC-apx-GP is shown to achieve competitive accuracy in hyper-parameter estimates for small and medium fleet sizes, while DEC-gapx-GP produces accurate hyper-parameter estimates for all fleet sizes with reasonable computations of local entities. Additionally, we propose a centralized GP training method, the gapx-GP, that improves the accuracy of hyper-parameter estimates for medium and large fleet sizes, entails reasonable computations, and requires little information exchange. Next, we use iterative and consensus protocols to decentralize the implementation of various aggregation of GP experts methods. We propose 13 techniques that can be used in various applications depending on the fleet size, the computational resources of local agents, and the communication capabilities. Moreover, we introduce a nearest neighbor selection method, namely CBNN, that excludes agents with no statistical correlation from the GP prediction. Although the CBNN achieves on average 42.5% agent reduction, it does not sacrifice prediction accuracy, and leads to significant computation and communication reduction. Most of the proposed decentralized GP prediction methods converge to the optimal values without reporting approximation error for all fleet sizes. The decentralized NPAE-based methods converge with approximation error, yet for DEC-NPAE* the error is insignificant. DEC-NPAE* and DEC-NN-grBCM are the most competitive methods for all fleet sizes both in terms of accuracy and uncertainty quantification, yet DEC-NN-grBCM is also scalable with low communication overhead.

Chapter 5

Conclusion and Future Work

This dissertation illustrates deficiencies in kriging for generating communication performance predictions, arising mainly from the structure of the assumptions. Moreover, our work shows that using range as a secondary variable in a cokriging formulation of the problem, yields lower absolute errors and performs better in long-term estimates. More specifically, we compare the proposed methodology with ordinary kriging and we show that the proposed framework provides better communication performance estimates with lower absolute errors in all simulation scenarios. Only in short-term estimates and in certain cases the ordinary kriging computes lower absolute errors. However, at distant locations of interest from the acquired measurements the proposed methodology provides better results. The simulations reveal that for realistic applications the assumption of stationary global mean of both techniques is rather conservative and develops unacceptable absolute errors. To address these problems we propose a model-based, data-driven learning technique for prediction of UWA communication performance in autonomous underwater vehicles beyond the observation area. The training with the proposed iterative technique, advocates to a bias-free and robust approach. We show that the proposed model-based learning yields accurate predictions, outperforming even three orders of magnitude other kriging methods in simulations. Moreover, the nested semivariogram function improves drastically the uncertainty quantification. In addition, experimental results reveal profoundly better predictions with our method for low ambient noise environments. In unpredictable and high ambient noise environments, there is

no method that provides accurate predictions. Yet, in such cases the proposed methodology identifies the ambient noise by reporting realistic uncertainty bounds.

To make the prediction of communication performance compatible for decentralized networks we propose methods to decentralize GPs. The proposed methods cover a broad spectrum of multi-agent learning applications as they can be employed both for decentralized GP training and decentralized GP prediction on various fleet sizes with different computation and communication capabilities of local agents. We utilize distributed optimization methods of ADMM to perform accurate and scalable GP training in networks. More specifically, a closed-form solution of the decentralize ADMM is derived for the case of GP hyper-parameter training with maximum likelihood estimation. DEC-apx-GP is shown to achieve competitive accuracy in hyper-parameter estimates for small and medium fleet sizes, while DEC-gapx-GP produces accurate hyper-parameter estimates for all fleet sizes with reasonable computations of local entities. Additionally, we propose a centralized GP training method, the gapx-GP, that improves the accuracy of hyper-parameter estimates for medium and large fleet sizes, entails reasonable computations, and requires little information exchange. Next, we use iterative and consensus protocols to decentralize the implementation of various aggregation of GP experts methods. We propose 13 techniques that can be used in various applications depending on the fleet size, the computational resources of local agents, and the communication capabilities. Moreover, we introduce a nearest neighbor selection method, namely CBNN, that excludes agents with no statistical correlation from the GP prediction. Although the CBNN achieves on average 42.5% agent reduction, it does not sacrifice prediction accuracy, and leads to significant computation and communication reduction. Most of the proposed decentralized GP prediction methods converge to the optimal values without reporting approximation error for all fleet sizes. The decentralized NPAE-based methods converge with approximation error, yet for DEC-NPAE* the error is insignificant. DEC-NPAE* and DEC-

NN-grBCM are the most competitive methods for all fleet sizes both in terms of accuracy and uncertainty quantification, yet DEC-NN-grBCM is also scalable with low communication overhead.

Ongoing work is focusing on decentralized active learning techniques to design multi-agent motion planning strategies that aim to reduce the uncertainty of areas in search missions. In addition, Bayesian model calibration will be used to calibrate variables of a realistic underwater acoustic propagation model using limited field data.

Appendices

Appendix A

Gradients

A.1 Partial derivative of SE covariance function

The partial derivative of the covariance function (4.5) is computed with respect to each hyperparameter as $\partial \boldsymbol{C}_\theta / \partial \boldsymbol{\theta} = (\partial \boldsymbol{C}_\theta / \partial l_1, \partial \boldsymbol{C}_\theta / \partial l_2, \ldots, \partial \boldsymbol{C}_\theta / \partial l_D, \partial \boldsymbol{C}_\theta / \partial \sigma_f, \partial \boldsymbol{C}_\theta / \partial \sigma_\epsilon)^\intercal$. In particular, for each length-scale l_d we obtain,

$$\left[\frac{\partial \boldsymbol{C}_\theta}{\partial l_d}\right]_{ij} = \sigma_f^2 \left[\exp\left\{-\frac{\|x_{id} - x_{jd}\|^2}{l_d}\right\} \frac{\|x_{id} - x_{jd}\|^2}{l_d^2}\right]_{ij},$$

where $\partial \boldsymbol{C}_\theta / \partial l_d \in \mathbb{R}^{N \times N}$. For the signal variance we get,

$$\left[\frac{\partial \boldsymbol{C}_\theta}{\partial \sigma_f}\right]_{ij} = 2\sigma_f \left[\exp\left\{-\sum_{d=1}^{D} \frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2}{l_d}\right\}\right]_{ij},$$

where $\partial \boldsymbol{C}_\theta / \partial \sigma_f \in \mathbb{R}^{N \times N}$. Lastly, for the measurement noise variance $\partial \boldsymbol{C}_\theta / \partial \sigma_\epsilon = 2\sigma_\epsilon I_N$.

A.2 Gradient for nested problem of DEC-c-ADMM-GP

Let the objective for the nested optimization problem (4.31b) of the DEC-c-ADMM-GP to be $\mathcal{K} = \mathcal{L}_i(\boldsymbol{\theta}_i) + \boldsymbol{\theta}_i^\intercal \boldsymbol{p}_i^{(s+1)} + \rho \sum_{j \in \mathcal{N}_i} \left\| \boldsymbol{\theta}_i - (\boldsymbol{\theta}_i^{(s)} + \boldsymbol{\theta}_j^{(s)})/2 \right\|_2^2$, then its gradient yields,

$$\frac{\partial \mathcal{K}}{\partial \boldsymbol{\theta}_i} = \nabla_{\boldsymbol{\theta}_i} \mathcal{L}_i(\boldsymbol{\theta}_i) + \boldsymbol{p}_i^{(s+1)} + 2\rho \sum_{j \in \mathcal{N}_i} \boldsymbol{\theta}_i - \frac{\boldsymbol{\theta}_i^{(s)} + \boldsymbol{\theta}_j^{(s)}}{2}.$$

Note that $\nabla_{\boldsymbol{\theta}_i} \mathcal{L}_i$ can be computed as in Appendix A.1.

Appendix B

Proofs

B.1 Proof of Proposition 4.10

The prediction mean value of any agent i using PoE yields,

$$\begin{aligned}\mu_{\text{PoE}}(\boldsymbol{x}_*) &= \sigma^2_{\text{PoE}}(\boldsymbol{x}_*) \sum_{i=1}^{M} \beta_i \sigma_i^{-2}(\boldsymbol{x}_*) \mu_i(\boldsymbol{x}_*) \\ &= \left(\sum_{i=1}^{M} \beta_i \sigma_i^{-2}(\boldsymbol{x}_*) \right)^{-1} \sum_{i=1}^{M} \beta_i \sigma_i^{-2}(\boldsymbol{x}_*) \mu_i(\boldsymbol{x}_*) \\ &= \sum_{i=1}^{M} \sigma_i^2(\boldsymbol{x}_*) \sum_{i=1}^{M} \sigma_i^{-2}(\boldsymbol{x}_*) \mu_i(\boldsymbol{x}_*). \end{aligned} \tag{B.1}$$

The prediction mean of the i-th agent using gPoE yields,

$$\begin{aligned}
\mu_{\text{gPoE}}(\boldsymbol{x}_*) &= \sigma^2_{\text{gPoE}}(\boldsymbol{x}_*) \sum_{i=1}^{M} \beta_i \sigma_i^{-2}(\boldsymbol{x}_*) \mu_i(\boldsymbol{x}_*) \\
&= \left(\sum_{i=1}^{M} \beta_i \sigma_i^{-2}(\boldsymbol{x}_*) \right)^{-1} \sum_{i=1}^{M} \beta_i \sigma_i^{-2}(\boldsymbol{x}_*) \mu_i(\boldsymbol{x}_*) \\
&= \left(\sum_{i=1}^{M} \frac{1}{M} \sigma_i^{-2}(\boldsymbol{x}_*) \right)^{-1} \sum_{i=1}^{M} \frac{1}{M} \sigma_i^{-2}(\boldsymbol{x}_*) \mu_i(\boldsymbol{x}_*) \\
&= \left(\frac{1}{M} \right)^{-1} \frac{1}{M} \sum_{i=1}^{M} \sigma_i^2(\boldsymbol{x}_*) \sum_{i=1}^{M} \sigma_i^{-2}(\boldsymbol{x}_*) \mu_i(\boldsymbol{x}_*) \\
&= \sum_{i=1}^{M} \sigma_i^2(\boldsymbol{x}_*) \sum_{i=1}^{M} \sigma_i^{-2}(\boldsymbol{x}_*) \mu_i(\boldsymbol{x}_*). \qquad \text{(B.2)}
\end{aligned}$$

Hence, from (B.1), (B.2) $\mu_{\text{PoE}}(\boldsymbol{x}_*) = \mu_{\text{gPoE}}(\boldsymbol{x}_*)$ for all $i \in \mathcal{V}$.

B.2 Proof of Theorem 4.21

Let us employ the local objective of (4.33b) as,

$$\mathcal{Q}_i = \nabla_{\boldsymbol{\theta}}^\intercal \mathcal{L}_i(\boldsymbol{\theta}_i^{(s)})(\boldsymbol{\theta}_i - \boldsymbol{\theta}_i^{(s)}) + \frac{\kappa_i}{2} \left\| \boldsymbol{\theta}_i - \boldsymbol{\theta}_i^{(s)} \right\|_2^2 + \boldsymbol{\theta}_i^\intercal \boldsymbol{p}_i^{(s+1)} + \rho \sum_{j \in \mathcal{N}_i} \left\| \boldsymbol{\theta}_i - \frac{\boldsymbol{\theta}_i^{(s)} + \boldsymbol{\theta}_j^{(s)}}{2} \right\|_2^2,$$

where $\mathcal{Q}_i : \mathbb{R}^{D+2} \to \mathbb{R}$. Next, factor out the optimizing parameter $\boldsymbol{\theta}_i$ to obtain,

$$\begin{aligned}
\mathcal{Q}_i &= \nabla_{\boldsymbol{\theta}}^\intercal \mathcal{L}_i(\boldsymbol{\theta}_i^{(s)}) \boldsymbol{\theta}_i - c_1 + \frac{\kappa_i}{2} \left(\boldsymbol{\theta}_i^\intercal \boldsymbol{\theta}_i - 2 \boldsymbol{\theta}_i^\intercal \boldsymbol{\theta}_i^{(s)} + c_2 \right) + \boldsymbol{\theta}_i^\intercal \boldsymbol{p}_i^{(s+1)} + \mathcal{T}_i \\
&= \boldsymbol{\theta}_i^\intercal \left[\nabla_{\boldsymbol{\theta}} \mathcal{L}_i(\boldsymbol{\theta}_i^{(s)}) - \kappa_i \boldsymbol{\theta}_i^{(s)} + \boldsymbol{p}_i^{(s+1)} \right] + \frac{\kappa_i}{2} \boldsymbol{\theta}_i^\intercal \boldsymbol{\theta}_i + \mathcal{T}_i, \qquad \text{(B.3)}
\end{aligned}$$

where $\mathcal{T}_i = \rho \sum_{j \in \mathcal{N}_i} \| \boldsymbol{\theta}_i - (\boldsymbol{\theta}_i^{(s)} + \boldsymbol{\theta}_j^{(s)})/2 \|_2^2$, $c_1 = -\nabla_{\boldsymbol{\theta}}^\intercal \mathcal{L}_i(\boldsymbol{\theta}_i^{(s)}) \boldsymbol{\theta}_i^{(s)}$, and $c_2 = \boldsymbol{\theta}_i^{\intercal(s)} \boldsymbol{\theta}_i^{(s)}$. Note that c_1, c_2 are constants with respect to the optimizing parameter $\boldsymbol{\theta}_i$ and thus irrelevant to

the problem. For any strongly connected graph topology, the term $\mathcal{T}_i$ can be expressed as,

$$\begin{aligned}
\mathcal{T}_i &= \rho \sum_{j\in\mathcal{N}_i} \left\| \boldsymbol{\theta}_i - \frac{\boldsymbol{\theta}_i^{(s)} + \boldsymbol{\theta}_j^{(s)}}{2} \right\|_2^2 \\
&= \rho \sum_{j\in\mathcal{N}_i} \boldsymbol{\theta}_i^\intercal \boldsymbol{\theta}_i - \boldsymbol{\theta}_i^\intercal (\boldsymbol{\theta}_i^{(s)} + \boldsymbol{\theta}_j^{(s)}) + c_3 \\
&= \rho \mathrm{card}(\mathcal{N}_i) \boldsymbol{\theta}_i^\intercal \boldsymbol{\theta}_i - \rho \sum_{j\in\mathcal{N}_i} \boldsymbol{\theta}_i^\intercal \boldsymbol{\theta}_i^{(s)} + \boldsymbol{\theta}_i^\intercal \boldsymbol{\theta}_j^{(s)} \\
&= \rho \mathrm{card}(\mathcal{N}_i) \boldsymbol{\theta}_i^\intercal \boldsymbol{\theta}_i - \rho \mathrm{card}(\mathcal{N}_i) \boldsymbol{\theta}_i^\intercal \boldsymbol{\theta}_i^{(s)} - \rho \boldsymbol{\theta}_i^\intercal \sum_{j\in\mathcal{N}_i} \boldsymbol{\theta}_j^{(s)} \\
&= \mathrm{card}(\mathcal{N}_i) \rho \boldsymbol{\theta}_i^\intercal \boldsymbol{\theta}_i - \rho \boldsymbol{\theta}_i^\intercal \Big[\mathrm{card}(\mathcal{N}_i) \boldsymbol{\theta}_i^{(s)} + \sum_{j\in\mathcal{N}_i} \boldsymbol{\theta}_j^{(s)} \Big],
\end{aligned} \tag{B.4}$$

where $c_3 = (1/4)(\boldsymbol{\theta}_i^{(s)} + \boldsymbol{\theta}_j^{(s)})^\intercal (\boldsymbol{\theta}_i^{(s)} + \boldsymbol{\theta}_j^{(s)})$ is a constant for the optimizing parameter $\boldsymbol{\theta}_i$ and thus ignored. The local objective $\mathcal{Q}_i$ by combining (B.3), (B.4) results in,

$$\begin{aligned}
\mathcal{Q}_i =& \boldsymbol{\theta}_i^\intercal \Big[\nabla_{\boldsymbol{\theta}} \mathcal{L}_i(\boldsymbol{\theta}_i^{(s)}) - \kappa_i \boldsymbol{\theta}_i^{(s)} + \boldsymbol{p}_i^{(s+1)} \Big] + \frac{\kappa_i}{2} \boldsymbol{\theta}_i^\intercal \boldsymbol{\theta}_i \\
&+ \mathrm{card}(\mathcal{N}_i) \rho \boldsymbol{\theta}_i^\intercal \boldsymbol{\theta}_i - \rho \boldsymbol{\theta}_i^\intercal \Big[\mathrm{card}(\mathcal{N}_i) \boldsymbol{\theta}_i^{(s)} + \sum_{j\in\mathcal{N}_i} \boldsymbol{\theta}_j^{(s)} \Big] \\
=& \boldsymbol{\theta}_i^\intercal \Big[\nabla_{\boldsymbol{\theta}} \mathcal{L}_i(\boldsymbol{\theta}_i^{(s)}) - \Big(\kappa_i + \mathrm{card}(\mathcal{N}_i) \rho \Big) \boldsymbol{\theta}_i^{(s)} + \boldsymbol{p}_i^{(s+1)} \\
&- \rho \sum_{j\in\mathcal{N}_i} \boldsymbol{\theta}_j^{(s)} \Big] + \Big(\frac{\kappa_i}{2} + \mathrm{card}(\mathcal{N}_i) \rho \Big) \boldsymbol{\theta}_i^\intercal \boldsymbol{\theta}_i.
\end{aligned} \tag{B.5}$$

Next, we show that the local objective $\mathcal{Q}_i$ (B.5) is a convex function in a quadratic form [11] by computing its Hessian,

$$\begin{aligned}
\mathcal{H}_{\mathcal{Q}_i} &= \frac{\partial^2 \mathcal{Q}_i}{\partial \boldsymbol{\theta}_i^2} \\
&= \left(\kappa_i + 2\mathrm{card}(\mathcal{N}_i) \rho \right) I_{D+2} \succ 0.
\end{aligned}$$

Since the local objective $\mathcal{Q}_i$ is convex and quadratic, we can obtain a closed-form solution by computing the first derivative,

$$\frac{\partial \mathcal{Q}}{\partial \boldsymbol{\theta}_i} = \nabla_{\boldsymbol{\theta}} \mathcal{L}_i(\boldsymbol{\theta}_i^{(s)}) - \Big(\kappa_i + \text{card}(\mathcal{N}_i)\rho\Big)\boldsymbol{\theta}_i^{(s)} + \boldsymbol{p}_i^{(s+1)} - \rho \sum_{j \in \mathcal{N}_i} \boldsymbol{\theta}_j^{(s)} + 2\Big(\frac{\kappa_i}{2} + \text{card}(\mathcal{N}_i)\rho\Big)\boldsymbol{\theta}_i,$$

and then set $\partial \mathcal{Q}/\partial \boldsymbol{\theta}_i = \mathbf{0}$, which yields,

$$\boldsymbol{\theta}_i = \frac{1}{\kappa_i + 2\text{card}(\mathcal{N}_i)\rho}\Big[\rho \sum_{j \in \mathcal{N}_i} \boldsymbol{\theta}_j^{(s)} - \nabla_{\boldsymbol{\theta}} \mathcal{L}_i(\boldsymbol{\theta}_i^{(s)}) + \Big(\kappa_i + \text{card}(\mathcal{N}_i)\rho\Big)\boldsymbol{\theta}_i^{(s)} - \boldsymbol{p}_i^{(s+1)}\Big].$$

The rest proof is a direct consequence of [19, Theorem 1].

B.3 Proof of Lemma 4.42

The separable squared exponential kernel (4.2) is a monotonically decreasing function. Note that the rate of decrease depends on the signal variance σ_f^2 and the length-scales l_d for all $d = 1, \ldots, D$. In particular, the CBNN is described by a sub-graph $\mathcal{G}_{\text{NN}} = (\mathcal{V}_{\text{NN}}, \mathcal{E}_{\text{NN}}(t))$, where $\mathcal{V}_{\text{NN}} = \{i \in \mathcal{V} : \|\boldsymbol{x}_i - \boldsymbol{x}_*\| \leq r_{\text{NN}}\}$ for all $\boldsymbol{x}_*$ with r_{NN} the covariance-based radius and $\mathcal{E}_{\text{NN}}(t) \subseteq \mathcal{V}_{\text{NN}} \times \mathcal{V}_{\text{NN}}$. Hence, the CBNN maintains strong connectivity.

Bibliography

[1] Robert J Adler and Jonathan E Taylor. *Random fields and geometry*. New York, NY, USA: Springer, 1 edition, 2007.

[2] Ian F Akyildiz, Dario Pompili, and Tommaso Melodia. Underwater acoustic sensor networks: Research challenges. *Ad Hoc Networks*, 3(3):257–279, 2005.

[3] Rakshit Allamraju and Girish Chowdhary. Communication efficient decentralized Gaussian process fusion for multi-UAS path planning. In *American Control Conference*, pages 4442–4447, 2017.

[4] Ricardo Augusto and Cristiano Panazio. On geostatistical methods for radio environment maps generation under location uncertainty. *Journal of Communication and Information Systems*, 33(1), 2018.

[5] François Bachoc, Nicolas Durrande, Didier Rullière, and Clément Chevalier. Some properties of nested Kriging predictors. *arXiv preprint arXiv:1707.05708*, 2017.

[6] Tucker Balch and Ronald C Arkin. Communication in reactive multiagent robotic systems. *Autonomous Robots*, 1(1):27–52, 1994.

[7] Sudipto Banerjee, Bradley P Carlin, and Alan E Gelfand. *Hierarchical modeling and analysis for spatial data*. Boca Raton, FL, USA: CRC Press, 2 edition, 2014.

[8] Subhasish Basak, Sébastien Petit, Julien Bect, and Emmanuel Vazquez. Numerical issues in maximum likelihood parameter estimation for Gaussian process interpolation. In *7th International Conference on Machine Learning, Optimization and Data science (LOD 2021)*, 2021.

[9] Christian Berg, Jens Peter Reus Christensen, and Paul Ressel. *Harmonic analysis on semigroups: Theory of positive definite and related functions*, volume 100. Springer, 1984.

[10] Dimitri P Bertsekas and John N Tsitsiklis. *Parallel and distributed computation: Numerical methods*. Athena Scientific, 2003.

[11] Stephen Boyd, Stephen P Boyd, and Lieven Vandenberghe. *Convex optimization*. Cambridge University Press, 2004.

[12] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. *Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers*, volume 3. Foundations and Trends in Machine Learning, 2011.

[13] Leonid Maksimovich Brekhovskikh and Yu P Lysanov. *Fundamentals of ocean acoustics*. New York, NY, USA: Springer-Verlag, 3 edition, 2003.

[14] Francesco Bullo, Jorge Cortes, and Sonia Martinez. *Distributed control of robotic networks: A mathematical approach to motion coordination algorithms*, volume 27. Princeton University Press, 2009.

[15] Wolfram Burgard, Mark Moors, Cyrill Stachniss, and Frank E Schneider. Coordinated multi-robot exploration. *IEEE Transactions on Robotics*, 21(3):376–386, 2005. doi: 10.1109/TRO.2004.839232.

[16] Richard H Byrd, Peihuang Lu, Jorge Nocedal, and Ciyou Zhu. A limited memory algorithm for bound constrained optimization. *SIAM Journal on Scientific Computing*, 16(5):1190–1208, 1995.

[17] Yanshuai Cao and David J Fleet. Generalized product of experts for automatic and

principled fusion of Gaussian process predictions. *arXiv preprint arXiv:1410.7827*, 2014.

[18] Paulo Cardieri and Theodore S Rappaport. Statistics of the sum of lognormal variables in wireless communications. In *IEEE Vehicular Technology Conference*, pages 1823–1827, 2000.

[19] Tsung-Hui Chang, Mingyi Hong, and Xiangfeng Wang. Multi-agent distributed optimization via inexact consensus admm. *IEEE Transactions on Signal Processing*, 63 (2):482–497, 2014.

[20] Jie Chen, Kian Hsiang Low, Yujian Yao, and Patrick Jaillet. Gaussian process decentralized data fusion and active sensing for spatiotemporal traffic modeling and prediction in mobility-on-demand systems. *IEEE Transactions on Automation Science and Engineering*, 12(3):901–921, 2015.

[21] Zexun Chen and Bo Wang. How priors of initial hyperparameters affect Gaussian process regression models. *Neurocomputing*, 275:1702–1710, 2018.

[22] Toshio Michael Chin, Jorge Vazquez-Cuervo, and Edward M Armstrong. A multi-scale high-resolution analysis of global sea surface temperature. *Remote sensing of environment*, 200:154–169, 2017.

[23] Sungjoon Choi, Mahdi Jadaliha, Jongeun Choi, and Songhwai Oh. Distributed Gaussian process regression under localization uncertainty. *Journal of Dynamic Systems, Measurement, and Control*, 137(3), 2015.

[24] Siddharth Choudhary, Luca Carlone, Carlos Nieto, John Rogers, Henrik I Christensen, and Frank Dellaert. Distributed mapping with privacy and communication constraints:

Lightweight algorithms and object-based models. *International Journal of Robotics Research*, 36(12):1286–1311, 2017.

[25] Jorge Cortés. Distributed Kriged Kalman filter for spatial estimation. *IEEE Transactions on Automatic Control*, 54(12):2816–2827, 2009.

[26] Noel Cressie. Fitting variogram models by weighted least squares. *Journal of the International Association for Mathematical Geology*, 17(5):563–586, 1985.

[27] Noel Cressie. *Statistics for spatial data*. Wiley, 2 edition, 1993.

[28] Noel Cressie and Douglas M Hawkins. Robust estimation of the variogram: I. *Journal of the International Association for Mathematical Geology*, 12(2):115–125, 1980.

[29] Abhirup Datta, Sudipto Banerjee, Andrew O Finley, and Alan E Gelfand. Hierarchical nearest-neighbor Gaussian process models for large geostatistical datasets. *Journal of the American Statistical Association*, 111(514):800–812, 2016.

[30] Marc Deisenroth and Jun Wei Ng. Distributed Gaussian processes. In *International Conference on Machine Learning*, pages 1481–1490, 2015.

[31] Peter J Diggle and Paulo J Ribeiro. *Model-based geostatistics*. New York, NY, USA: Springer, 1 edition, 2007.

[32] David Duvenaud, James Robert Lloyd, Roger Grosse, Joshua B Tenenbaum, and Zoubin Ghahramani. Structure discovery in nonparametric regression through compositional kernel search. In *International Conference on Machine Learning*, 2013.

[33] Paul C Etter. *Underwater acoustic modeling and simulation*. CRC press, 2018.

[34] Brian Ferris, Dieter Fox, and Neil D Lawrence. WiFi-SLAM using Gaussian process latent variable models. In *International Joint Conferences on Artificial Intelligence*, pages 2480–2485, 2007.

[35] Jonathan Fink, Alejandro Ribeiro, and Vijay Kumar. Robust control of mobility and communications in autonomous robot teams. *IEEE Access*, 1:290–309, 2013. doi: 10.1109/access.2013.2262013.

[36] Lee Freitag and Sandipa Singh. Performance of micro-modem PSK signaling under variable conditions during the 2008 RACE and SPACE experiments. In *IEEE OCEANS*, pages 1–8, 2009.

[37] Lee Freitag and Sandipa Singh. Performance of micro-modem PSK signaling with a mobile transmitter during the 2010 MACE experiment. In *IEEE OCEANS-Genova*, pages 1–7, 2015.

[38] Wayne A Fuller. *Introduction to statistical time series*. John Wiley & Sons, 2 edition, 1996.

[39] Eric Gallimore, Jim Partan, Ian Vaughn, Sandipa Singh, Jon Shusta, and Lee Freitag. The WHOI micromodem-2: A scalable system for acoustic communications and networking. In *MTS/IEEE OCEANS*, 2010.

[40] Gene H Golub and Charles F Van Loan. *Matrix computations*. Johns Hopkins University Press, 4 edition, 2013.

[41] Ramón González, Paramsothy Jayakumar, and Karl Iagnemma. Stochastic mobility prediction of ground vehicles over large spatial regions: a geostatistical approach. *Autonomous Robots*, 41(2):311–331, 2017.

[42] Rishi Graham and Jorge Cortés. Spatial statistics and distributed estimation by robotic sensor networks. In *American Control Conference*, pages 2422–2427, 2010.

[43] Robert B. Gramacy. *Surrogates: Gaussian Process Modeling, Design and Optimization for the Applied Sciences*. Chapman Hall/CRC, Boca Raton, Florida, 2020. http://bobby.gramacy.com/surrogates/.

[44] Dongbing Gu and Huosheng Hu. Spatial Gaussian process regression with mobile sensor networks. *IEEE Transactions on Neural Networks and Learning Systems*, 23 (8):1279–1290, 2012.

[45] Jun Guo, Yan-Zhao Xie, and Ai-Ci Qiu. JOR iterative method for the modeling of MTLs excited by EMP. *IEEE Antennas and Wireless Propagation Letters*, 15:536–539, 2015.

[46] Brian Ferris Dirk Hähnel and Dieter Fox. Gaussian processes for signal strength-based location estimation. In *Robotics: Science and Systems*, 2006.

[47] Trevor Halsted, Ola Shorinwa, Javier Yu, and Mac Schwager. A survey of distributed optimization methods for multi-robot systems. *arXiv preprint arXiv:2103.12840*, 2021.

[48] David A Harville. Bayesian inference for variance components using only error contrasts. *Biometrika*, 61(2):383–385, 1974.

[49] Tomislav Hengl, Gerard BM Heuvelink, and Alfred Stein. A generic framework for spatial prediction of soil variables based on regression-kriging. *Geoderma*, 120(1-2): 75–93, 2004.

[50] James Hensman, Nicolo Fusi, and Neil D Lawrence. Gaussian processes for big data. In *Conference on Uncertainty in Artificial Intelligence*, page 282, 2013.

[51] Geoffrey E Hinton. Training products of experts by minimizing contrastive divergence. *Neural computation*, 14(8):1771–1800, 2002.

[52] Trong Nghia Hoang, Quang Minh Hoang, Kian Hsiang Low, and Jonathan How. Collective online learning of Gaussian processes in massive multi-agent systems. In *AAAI Conference on Artificial Intelligence*, volume 33, pages 7850–7857, 2019.

[53] Mingyi Hong, Zhi-Quan Luo, and Meisam Razaviyayn. Convergence analysis of alternating direction method of multipliers for a family of nonconvex problems. *SIAM Journal on Optimization*, 26(1):337–364, 2016.

[54] Douglas Horner and Geoffrey Xie. Data-driven acoustic communication modeling for undersea collaborative navigation. In *IEEE OCEANS*, 2013.

[55] Eric Horvitz and Deirdre Mulligan. Data, privacy, and the greater good. *Science*, 349 (6245):253–255, 2015.

[56] Dohyun Jang, Jaehyun Yoo, Clark Youngdong Son, Dabin Kim, and H Jin Kim. Multi-robot active sensing and environmental model learning with distributed Gaussian process. *IEEE Robotics and Automation Letters*, 5(4):5905–5912, 2020.

[57] Finn B Jensen, William A Kuperman, Michael B Porter, and Henrik Schmidt. *Computational ocean acoustics*. Springer Science & Business Media, 2011.

[58] Andre G Journel. Markov models for cross-covariances. *Mathematical Geology*, 31(8): 955–964, 1999.

[59] JPL MUR MEaSUREs Project. GHRSST Level 4 MUR Global Foundation Sea Surface Temperature Analysis. Ver. 4.1. PO.DAAC, CA, USA; https://doi.org/10.5067/GHGMR-4FJ04, 2015. Online; accessed October, 20 2020.

[60] Yiannis Kantaros and Michael M Zavlanos. Distributed intermittent connectivity control of mobile robot networks. *IEEE Transactions on Automatic Control*, 62(7):3109–3121, 2016. doi: 10.1109/tac.2016.2626400.

[61] Michael E Kepler and Daniel J Stilwell. An approach to reduce communication for multi-agent mapping applications. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4814–4820.

[62] Koresh Khateri, Mahdi Pourgholi, Mohsen Montazeri, and Lorenzo Sabattini. A connectivity preserving node permutation local method in limited range robotic networks. *Robotics and Autonomous Systems*, page 103540, 2020. doi: 10.1016/j.robot.2020.103540.

[63] Reza Khodayi-mehr, Yiannis Kantaros, and Michael M Zavlanos. Distributed state estimation using intermittently connected robot networks. *IEEE Transactions on Robotics*, 35(3):709–724, 2019. doi: 10.1109/tro.2019.2897865.

[64] Jongyun Kim, Pawel Ladosz, and Hyondong Oh. Optimal communication relay positioning in mobile multi-node networks. *Robotics and Autonomous Systems*, 129:103517, 2020. doi: 10.1016/j.robot.2020.103517.

[65] Peter K Kitanidis. Parametric estimation of covariances of regionalized variables. *Journal of the American Water Resources Association*, 23(4):557–567, 1987.

[66] Peter K Kitanidis. Generalized covariance functions in estimation. *Mathematical Geology*, 25(5):525–540, 1993.

[67] Jakub Konečnỳ, H Brendan McMahan, Felix X Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492*, 2016.

[68] George P Kontoudis and Daniel J Stilwell. A comparison of kriging and cokriging for estimation of underwater acoustic communication performance. In *International Conference on Underwater Networks & Systems*, pages 1–8, 2019.

[69] George P Kontoudis and Daniel J Stilwell. Decentralized nested Gaussian processes for multi-robot systems. In *IEEE International Conference on Robotics and Automation*, pages 8881–8887, 2021.

[70] George P Kontoudis and Daniel J Stilwell. Prediction of acoustic communication performance in marine robots using model-based kriging. In *American Control Conference*, pages 3779–3786, 2021.

[71] George P Kontoudis, Stephen Krauss, and Daniel J Stilwell. Model-based learning of underwater acoustic communication performance for marine robots. *Robotics and Autonomous Systems*, 142:103811, 2021.

[72] Liu Lanbo, Zhou Shengli, and Cui Jun-Hong. Prospects and problems of wireless communication for underwater sensor networks. *Wireless Communications and Mobile Computing*, 8(8):977–994, 2008.

[73] Alberto Quattrini Li, Phani Krishna Penumarthi, Jacopo Banfi, Nicola Basilico, Jason M O'Kane, Ioannis Rekleitis, Srihari Nelakuditi, and Francesco Amigoni. Multi-robot online sensing strategies for the construction of communication maps. *Autonomous Robots*, pages 299–319, 2019.

[74] Andreas Lichtenstern. Kriging methods in spatial statistics. Thesis, Technical University of Munich, August 2013.

[75] Tony X Lin, Said Al-Abri, Samuel Coogan, and Fumin Zhang. A distributed scalar

field mapping strategy for mobile robots. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 11581–11586, 2020.

[76] Haitao Liu, Jianfei Cai, Yi Wang, and Yew Soon Ong. Generalized robust Bayesian committee machine for large-scale Gaussian process regression. In *International Conference on Machine Learning*, pages 3131–3140, 2018.

[77] Haitao Liu, Yew-Soon Ong, Xiaobo Shen, and Jianfei Cai. When Gaussian process meets big data: A review of scalable GPs. *IEEE Transactions on Neural Networks and Learning Systems*, 2020.

[78] Ji Liu, Shaoshuai Mou, and A Stephen Morse. Asynchronous distributed algorithms for solving linear algebraic equations. *IEEE Transactions on Automatic Control*, 63 (2):372–385, 2018.

[79] DJ C MACKAY. Introduction to Gaussian processes. *NATO ASI series. Series F: Computer and System Sciences*, pages 133–165, 1998.

[80] Ali Makhdoumi and Asuman Ozdaglar. Convergence rate of distributed ADMM over networks. *IEEE Transactions on Automatic Control*, 62(10):5082–5095, 2017.

[81] Mehrzad Malmirchegini and Yasamin Mostofi. On the spatial predictability of communication channels. *IEEE Transactions on Wireless Communications*, 11(3):964–978, 2012.

[82] Gonzalo Mateos, Juan Andrés Bazerque, and Georgios B Giannakis. Distributed sparse linear regression. *IEEE Transactions on Signal Processing*, 58(10):5262–5276, 2010.

[83] Bertil Matérn. *Spatial variation*. Berlin, Germany: Springer-Verlag, 2 edition, 1960; reprinted 1986.

[84] Georges Matheron. *Traité de géostatistique appliquée*, volume 1. Editions Technip, 1962.

[85] Georges Matheron. Principles of geostatistics. *Economic Geology*, 58(8):1246–1266, 1963.

[86] Brian McCarter, Stephen Portner, Wayne L Neu, Daniel J Stilwell, Dexter Malley, and Jason Minis. Design elements of a small AUV for bathymetric surveys. In *IEEE/OES Autonomous Underwater Vehicles*, pages 1–5, 2014.

[87] Marco Minelli, Jacopo Panerati, Marcel Kaufmann, Cinara Ghedini, Giovanni Beltrame, and Lorenzo Sabattini. Self-optimization of resilient topologies for fallible multi-robots. *Robotics and Autonomous Systems*, 124:103384, 2020. doi: 10.1016/j.robot.2019.103384.

[88] In Jae Myung. Tutorial on maximum likelihood estimation. *Journal of Mathematical Psychology*, 47(1):90–100, 2003.

[89] Andrew A Neath and Joseph E Cavanaugh. The Bayesian information criterion: Background, derivation, and applications. *Wiley Interdisciplinary Reviews: Computational Statistics*, 4(2):199–203, 2012.

[90] Jun Wei Ng and Marc Peter Deisenroth. Hierarchical mixture-of-experts model for large-scale gaussian process regression. *arXiv preprint arXiv:1412.3078*, 2014.

[91] Linh V Nguyen, Sarath Kodagoda, Ravindra Ranasinghe, and Gamini Dissanayake. Information-driven adaptive sampling strategy for mobile robotic wireless sensor network. *IEEE Transactions on Control Systems Technology*, 24(1):372–379, 2015.

[92] Ertug Olcay, Fabian Schuhmann, and Boris Lohmann. Collective navigation of a multi-

robot system in an unknown environment. *Robotics and Autonomous Systems*, 132: 103604, 2020. doi: 10.1016/j.robot.2020.103604.

[93] Reza Olfati-Saber, J Alex Fax, and Richard M Murray. Consensus and cooperation in networked multi-agent systems. *Proceedings of the IEEE*, 95(1):215–233, 2007.

[94] Alex Olshevsky and John N Tsitsiklis. Convergence speed in distributed consensus and averaging. *SIAM Journal on Control and Optimization*, 48(1):33–55, 2009.

[95] Derek A Paley and Artur Wolek. Mobile sensor networks and control: Adaptive sampling of spatiotemporal processes. *Annual Review of Control, Robotics, and Autonomous Systems*, 3, 2019.

[96] Fernando Pérez-Cruz, Steven Van Vaerenbergh, Juan José Murillo-Fuentes, Miguel Lázaro-Gredilla, and Ignacio Santamaria. Gaussian processes for nonlinear signal processing: An overview of recent advances. *IEEE Signal Processing Magazine*, 30(4): 40–50, 2013.

[97] Gianluigi Pillonetto, Luca Schenato, and Damiano Varagnolo. Distributed multi-agent Gaussian regression via finite-dimensional approximations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(9):2098–2111, 2018.

[98] Michael B Porter. The BELLHOP manual and user's guide: Preliminary draft, 2011.

[99] Parastoo Qarabaqi and Milica Stojanovic. Modeling the large scale transmission loss in underwater acoustic channels. In *IEEE Allerton Conference on Communication, Control, and Computing*, pages 445–452, 2011.

[100] Parastoo Qarabaqi and Milica Stojanovic. Statistical characterization and computationally efficient modeling of a class of underwater acoustic communication channels. *IEEE Journal of Oceanic Engineering*, 38(4):701–717, 2013.

[101] Joaquin Quiñonero-Candela and Carl Edward Rasmussen. A unifying view of sparse approximate Gaussian process regression. *Journal of Machine Learning Research*, 6 (Dec):1939–1959, 2005.

[102] Joaquin Quinonero-Candela, Carl Edward Rasmussen, Fabian Sinz, Olivier Bousquet, and Bernhard Schölkopf. Evaluating predictive uncertainty challenge. In *Machine Learning Challenges Workshop*, pages 1–27, 2005.

[103] Carl Edward Rasmussen and Hannes Nickisch. Gaussian processes for machine learning (gpml) toolbox. *Journal of Machine Learning Research*, 11:3011–3015, 2010.

[104] Carl Edward Rasmussen and Christopher KI Williams. *Gaussian Processes for Machine Learning*. Cambridge, MA, USA: MIT Press, 2 edition, 2006.

[105] Paulo J Ribeiro Jr and Peter J Diggle. The geoR package. https://cran.r-project.org/web/packages/geoR/index.html, 2007. version 1.7-5.2.1.

[106] Jacques Rivoirard. Which models for collocated cokriging? *Mathematical Geology*, 33 (2):117–131, 2001.

[107] Didier Rullière, Nicolas Durrande, François Bachoc, and Clément Chevalier. Nested Kriging predictions for datasets with a large number of observations. *Statistics and Computing*, 28(4):849–867, 2018.

[108] Lorenzo Sabattini, Nikhil Chopra, and Cristian Secchi. Decentralized connectivity maintenance for cooperative control of mobile robotic systems. *The International Journal of Robotics Research*, 32(12):1411–1423, 2013. doi: 10.1177/0278364913499085.

[109] Gideon Schwarz. Estimating the dimension of a model. *The Annals of Statistics*, 6(2): 461–464, 1978.

[110] Wei Shi, Qing Ling, Kun Yuan, Gang Wu, and Wotao Yin. On the linear convergence of the ADMM in decentralized consensus optimization. *IEEE Transactions on Signal Processing*, 62(7):1750–1761, 2014.

[111] Michael Shinego, Geoff Edelson, Francine Menas, Michael Richman, and Robert Nation. Underwater acoustic data communications for autonomous platform command, control and communications. Technical report, BAE Systems and Technology, 2001.

[112] Amarjeet Singh, Andreas Krause, Carlos Guestrin, and William J Kaiser. Efficient informative sensing using multiple robots. *Journal of Artificial Intelligence Research*, 34:707–755, 2009.

[113] Edward Snelson and Zoubin Ghahramani. Sparse Gaussian processes using pseudo-inputs. In *Advances in Neural Information Processing Systems*, pages 1257–1264, 2006.

[114] Balaji Vasan Srinivasan, Ramani Duraiswami, and Raghu Murtugudde. Efficient kriging for real-time spatio-temporal interpolation. In *Conference on Probability and Statistics in the Atmospheric Sciences*, 2010.

[115] Milica Stojanovic. On the relationship between capacity and distance in an underwater acoustic communication channel. *ACM SIGMOBILE Mobile Computation and Communication Review*, 11(4):34–43, 2007.

[116] Milica Stojanovic and James Preisig. Underwater acoustic communication channels: Propagation models and statistical characterization. *IEEE Communications Magazine*, 47(1):84–89, 2009.

[117] Jie Sun, Shijie Liu, Fumin Zhang, Aijun Song, Jiancheng Yu, and Aiqun Zhang. A kriged compressive sensing approach to reconstruct acoustic fields from measurements collected by underwater vehicles. *IEEE Journal of Oceanic Engineering*, 2020.

[118] Yoonchang Sung, Ashish Kumar Budhiraja, Ryan K Williams, and Pratap Tokekar. Distributed assignment with limited communication for multi-robot multi-target tracking. *Autonomous Robots*, 44(1):57–73, 2020. doi: 10.1007/s10514-019-09856-1.

[119] Varun Suryan and Pratap Tokekar. Learning a spatial field in minimum time with a team of robots. *IEEE Transactions on Robotics*, 36(5):1562–1576, 2020.

[120] Hiroyuki Takeda, Sina Farsiu, and Peyman Milanfar. Kernel regression for image processing and reconstruction. *IEEE Transactions on Image Processing*, 16(2):349–366, 2007.

[121] Hwee-Pink Tan, Roee Diamant, Winston KG Seah, and Marc Waldmeyer. A survey of techniques and challenges in underwater localization. *Ocean Engineering*, 38(14-15): 1663–1676, 2011.

[122] Qiuyang Tao, Yuehai Zhou, Feng Tong, Aijun Song, and Fumin Zhang. Evaluating acoustic communication performance of micro autonomous underwater vehicles in confined spaces. *Frontiers of Information Technology and Electronic Engineering*, 19(8): 1013–1023, 2018.

[123] Mostafa Tavassolipour, Seyed Abolfazl Motahari, and Mohammad Taghi Manzuri Shalmani. Learning of Gaussian processes in distributed and communication limited systems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(8):1928–1941, 2020.

[124] Beatrice Tomasi, James Preisig, and Michele Zorzi. On the predictability of underwater acoustic communications performance: The KAM11 data set as a case study. In *International Workshop on Underwater Networks*, pages 1–7, 2011.

[125] Donald M. Topkis. Concurrent broadcast for information dissemination. *IEEE Transactions on Software Engineering*, SE-11(10):1107–1112, 1985.

[126] Volker Tresp. A Bayesian committee machine. *Neural computation*, 12(11):2719–2741, 2000.

[127] Firdaus E Udwadia. Some convergence results related to the JOR iterative method for symmetric, positive-definite matrices. *Applied Mathematics and Computation*, 47(1): 37–45, 1992.

[128] Muhammad Umer, Lars Kulik, and Egemen Tanin. Spatial interpolation in wireless sensor networks: Localized algorithms for variogram modeling and kriging. *Geoinformatica*, 14(1):101, 2010.

[129] Hans Wackernagel. *Multivariate geostatistics: An introduction with applications*. New York, NY, USA: Springer-Verlag, 3 edition, 2003.

[130] Xuan Wang, Shaoshuai Mou, and Dengfeng Sun. Improvement of a distributed algorithm for solving linear equations. *IEEE Transactions on Industrial Electronics*, 64(4): 3113–3117, 2016.

[131] Richard Webster and Margaret A Oliver. *Geostatistics for environmental scientists*. New York, NY, USA: Wiley, 1 edition, 2007.

[132] Gordon M Wenz. Acoustic ambient noise in the ocean: Spectra and sources. *The Journal of the Acoustical Society of America*, 34(12):1936–1956, 1962.

[133] Ryan K Williams and Gaurav S Sukhatme. Constrained interaction and coordination in proximity-limited multiagent systems. *IEEE Transactions on Robotics*, 29(4):930–944, 2013. doi: 10.1109/tro.2013.2257578.

[134] Wencen Wu, Aijun Song, Paul Varnell, and Fumin Zhang. Cooperatively mapping of the underwater acoustic channel by robot swarms. In *ACM International Conference on Underwater Networks & Systems*, pages 1–8, 2014.

[135] Ang Xie, Feng Yin, Yue Xu, Bo Ai, Tianshi Chen, and Shuguang Cui. Distributed Gaussian processes hyperparameter optimization for big data using proximal ADMM. *IEEE Signal Processing Letters*, 26(8):1197–1201, 2019.

[136] Yue Xu, Feng Yin, Wenjun Xu, Jiaru Lin, and Shuguang Cui. Wireless traffic prediction with scalable Gaussian process: Framework, algorithms, and verification. *IEEE Journal on Selected Areas in Communications*, 37(6):1291–1306, 2019.

[137] Yunfei Xu, Jongeun Choi, and Songhwai Oh. Mobile sensor network navigation using Gaussian processes with truncated observations. *IEEE Transactions on Robotics*, 27 (6):1118–1131, 2011.

[138] Vikas Yadav and Murti V Salapaka. Distributed protocol for determining when averaging consensus is reached. In *Allerton Conference on Communication, Control, and Computing*, pages 715–720, 2007.

[139] Hongjun Yu, Cheng-Chew Lim, Robert Hunjet, and Peng Shi. Flocking and topology manipulation based on space partitioning. *Robotics and Autonomous Systems*, 124: 103328, 2020. doi: 10.1016/j.robot.2019.103328.

[140] Xi Yu and M Ani Hsieh. Synthesis of a time-varying communication network by robot teams with information propagation guarantees. *IEEE Robotics and Automation Letters*, 5(2):1413–1420, 2020. doi: 10.1109/lra.2020.2967704.

[141] Zhenyuan Yuan and Minghui Zhu. Communication-aware distributed Gaussian process

regression algorithms for real-time machine learning. In *American Control Conference*, pages 2197–2202, 2020.

[142] Zhenyuan Yuan and Minghui Zhu. Resource-aware distributed Gaussian process regression for real-time machine learning. *arXiv preprint arXiv:2105.04738*, 2021.

[143] Weiming Zhi, Lionel Ott, Ransalu Senanayake, and Fabio Ramos. Continuous occupancy map fusion with fast Bayesian Hilbert maps. In *IEEE International Conference on Robotics and Automation*, pages 4111–4117, 2019.

[144] Dale L Zimmerman. Computationally efficient restricted maximum likelihood estimation of generalized covariance functions. *Mathematical Geology*, 21(7):655–672, 1989.