Analytical Performance Bounds for Full and Reduced-order Distributed Bayesian Estimation

Arash Mohammadi, Member, IEEE and Amir Asif, Senior Member, IEEE

Abstract—Motivated by the resource management problem in nonlinear multi-sensor tracking networks, the paper derives online, distributed estimation algorithms for computing the posterior Cramér-Rao lower bound (PCRLB) for full-order and reduced-order distributed Bayesian estimators without requiring a fusion centre and with nodal communications limited to local neighborhoods. For both cases, Riccati-type recursions are derived that sequentially determine the global Fisher information matrix (FIM) from localized FIMs of the distributed estimators. We use particle filter realizations for these bounds and quantify their performance for data fusion problems through Monte-Carlo simulations.

IEEE Keywords: Cramér-Rao bounds, Data fusion, Distributed estimation, dPCRLB, Multi-sensor tracking, Nonlinear systems, Particle filters, and PCRLB.

I. INTRODUCTION

Sequential Bayesian estimation arises in several signal processing applications including state prediction, target tracking, adaptive control, and analysis or prediction of non-stationary time series. Conventional estimation approaches use a centralized architecture with the participating nodes communicating their raw observations (either directly or indirectly via a multipath relay) to a central processing unit, referred to as the fusion centre. Lately, there has been a growing interest in the decentralized and distributed architectures [1], where: (i) there is no fusion centre; (ii) the sensor nodes do not require global knowledge of the network topology, and; (iii) each node determines its own estimate locally and exchanges data only within its neighborhood. Consensus approaches are then typically used to achieve estimation consistency across the network. Such architectures offer several advantages including scalability, flexibility, and higher immunity to network failure over their centralized counterparts.

The paper derives distributed algorithms for online computation of the optimal posterior Cramér-Rao lower bound (PCRLB) [2] for decentralized sensor networks. The motivation for this work comes from sensor selection decisions [3]-[13] especially in geographically dispersed networks deploying an unrestrictedly large number of sensor nodes. Limitations in power, frequency, and bandwidth restrict the maximum number of active sensors that can simultaneously participate in the decentralized estimation process. The problem of sensor selection is to determine the optimal way of dynamically selecting a subset of sensors over time that provides the best estimation performance. Among the criteria proposed for sensor resource management, the PCRLB [10]-[16] provides a predictive measure of the achievable optimal performance. More importantly, this bound is independent of the estimation mechanism. In the past, sensor management algorithms based on the PCRLB have only been presented for the centralized networks with a fusion centre. No such work has been pursued for distributed estimation networks primarily because of the difficulty in computing the PCRLB distributively. The paper addresses this gap and derives optimal recursive PCRLB expressions, referred to as the distributed PCRLB (dPCRLB), for sensor networks configured using decentralized architectures1.

The seminal work of Tichavsky et al. [2] provides a recursive formula to update the Fisher information matrix (FIM), i.e., the inverse of the PCRLB, iteratively for a general multidimensional, discrete time, nonlinear, estimation problem in the centralized architecture while keeping the dimensions of the FIM constant. Based on [2], there has been a surge of interest in extending the PCRLB to more practical scenarios, e.g., to include measurement origin uncertainty [17], [18], to consider issues related to the quantization of sensor data, to compute approximated online PCRLB [19], and to derive online conditional PCRLB [20]. Subsequently, the PCRLB theory has been extended to several applications, e.g., for adaptive resource management [14], dynamic sensor selection [15], bearing-only tracking [21] and multiple target tracking [22].

As stated earlier, previous derivations of the PCRLB are limited to the centralized [17]-[14] and hierarchical estimation architectures [15] and only recently a suboptimal PCRLB expression [16] has been derived for the decentralized architectures. In this paper, optimal dPCRLB algorithms are derived for two different estimation methodologies: (i) full-order distributed approaches [23]-[34], where the entire state vector is estimated locally at each observation node without resorting to a fusion centre; (ii) reduced-order distributed approaches [35]-[40] for large-scale dynamical systems [41]-[47], where the overall system is partitioned [44], [46] into several lower dimensional subsystems with only a subset of state variables estimated at each observation node. The reduced-order subsystems are coupled and connectivity between neighboring subsystems [35] is maintained. Another motivation for computing the dPCRLB in reduced-order estimation is the unavailability of the global estimates of the full state vector at a single processing node. In our full-order and reduced-order dPCRLB computations, average consensus algorithms are used to distributively compute the summation terms involving local statistics such as the local FIMs. For distributed computation of the dPCRLB, we assume a connected network with at least one path traversing the complete network. Observability over the entire network is assumed though local observability is not 1We note that the centralized computation of the PCRLB cannot be realized [16] for dynamic resource allocation in decentralized networks due to the absence of the fusion centre and the only alternative is real-time, recursive computation of the dPCRLB in a distributed fashion.
required. Extension of this work to quantized observations in full order estimation has lately been reported in [45].

To summarize, the paper makes the following three important contributions to the literature of nonlinear, distributed signal processing. First, we derive the optimal expression for computing the dPCRLB for full-order distributed architectures. We provide a Riccati-type recursion that sequentially determines the optimal decentralized FIM from localized FIMs of the distributed estimators, which is used to compute the full-order dPCRLB (FO/dPCRLB). Second, we extend the dPCRLB algorithms to the reduced-order architectures where computing the FO/dPCRLB is not feasible as the global estimate of the entire state vector is not available separately at individual processing nodes. These performance bounds are valid for both linear and nonlinear, reduced-order estimation approaches including the Kalman and particle filters. A third contribution of the paper is the derivation of sequential Monte Carlo methods and other near-optimal practical approximations introduced for efficient computation of the dPCRLBs. The proposed dPCRLBs and their practical implementations are compared for a variety of full-order and reduced-order systems using Monte Carlo simulations. Our results indicate that the proposed dPCRLB algorithms provide an optimal bound and overlap the PCRLB plot derived from the centralized architecture.

The rest of paper is organized as follows. Section II introduces notation and reviews the centralized PCRLB. Section III derives an expression for computing the dPCRLB for a full-order distributed architecture. Section IV derives an expression for computing the dPCRLB for a reduced-order distributed architecture. Section V derives the expression for computing the dPCRLB for a full-order distributed architecture. Section VI concludes the paper. Appendices A to D includes proofs for theorems and results presented in the paper. Appendix E reviews the consensus algorithms, while Appendix F reviews the distributed iterate-collapse inversion algorithm (DICI-RO).

II. BACKGROUND

A network comprising of $N$ nodes observing a set of $n_x$ state variables $x = [X_1, X_2, \ldots, X_n_x]^T$ is considered. The overall nonlinear state model is given by

$$x(k) = f(x(k-1), \xi(k)),$$

with the detailed state representation

$$
\begin{bmatrix}
X_1(k) \\
\vdots \\
X_n_x(k)
\end{bmatrix} =
\begin{bmatrix}
f_1(x(k-1), \xi_1(k)) \\
\vdots \\
f_n_x(x(k-1), \xi_n_x(k))
\end{bmatrix}.
$$

The observation model may also be potentially nonlinear as

$$z(k) = g(x(k), \zeta(k)),$$

where $z(k)$ denotes the composite of observations made at all $N$ nodes. The expanded observation model is

$$
\begin{bmatrix}
z^{(1)}(k) \\
\vdots \\
z^{(N)}(k)
\end{bmatrix} =
\begin{bmatrix}
g^{(1)}(x(k), \zeta^{(1)}(k)) \\
\vdots \\
g^{(N)}(x(k), \zeta^{(N)}(k))
\end{bmatrix},
$$

where node $l$, $1 \leq l \leq N$, makes vector measurements $z^{(l)}(k)$ with dimension $n_z^{(l)}$ at discrete time instants $k, k \geq 1$. The state and observation functions $\{f(\cdot), g(\cdot)\}$ may possibly be nonlinear, and the global uncertainties $\{\xi(\cdot), \zeta(\cdot)\}$ in the process and observation dynamics are not necessarily limited to the Gaussian models. We denote the states and observations up to time $k$ as $x(0:k)$ and $z(1:k)$, respectively. The joint density function of the state variables and observations is given by

$$P(x(0:k), z(1:k))$$

and the posterior density function by

$$P(x(0:k)|z(1:k)).$$

In this paper, we use different factorizations of the overall posterior distribution. One commonly used factorization [52] is

$$P(x(0:k+1)|z(1:k+1)) =
\frac{P(z(k+1)|x(k+1))P(x(k+1)|x(k))P(x(0:k)z(1:k))}{P(z(k+1)|z(1:k))}.$$  

(5)

For $k \geq 1$, we define $\hat{x}(k)$ to be the estimate (i.e., the expected value) of the state vector $x(k)$ at time step $k$ based on observations taken up to $k$, and $\hat{P}(k)$ to be the mean squared error (covariance) matrix associated with estimate $\hat{x}(k)$, i.e.,

$$
\hat{x}(k) \triangleq \mathbb{E}[x(k)|z(1:k)]
$$

and

$$
\hat{P}(k) \triangleq \mathbb{E}\{(x(k) - \hat{x}(k))(x(k) - \hat{x}(k))^T\},
$$

(6)

where $\mathbb{E}$ is the expectation operator. Similarly, the predicted value of the state vector and its associated error covariance are

$$\hat{x}(k+1) \triangleq \mathbb{E}[x(k+1)|z(1:k)]$$

(7)

and

$$\hat{P}(k+1) \triangleq \mathbb{E}\{(x(k+1) - \hat{x}(k+1))(x(k+1) - \hat{x}(k+1))^T\}. $$

(8)

In a distributed estimation setup, each node establishes its local estimates based on its own observations. Such a local estimate at node $l$, $(1 \leq l \leq N)$, is defined as

$$\hat{x}^{(l)}(k) \triangleq \mathbb{E}[x^{(l)}(k)|z^{(l)}(1:k)]$$

(9)

and

$$\hat{P}^{(l)}(k) \triangleq \mathbb{E}\{(x^{(l)}(k) - \hat{x}^{(l)}(k))(x^{(l)}(k) - \hat{x}^{(l)}(k))^T\}. $$

(10)

Likewise, the locally predicted state estimate at node $l$ is

$$\hat{x}^{(l)}(k+1) \triangleq \mathbb{E}[x^{(l)}(k+1)|z^{(l)}(1:k)]$$

(11)

and

$$\hat{P}^{(l)}(k+1) \triangleq \mathbb{E}\{(x^{(l)}(k+1) - \hat{x}^{(l)}(k+1))(x^{(l)}(k+1) - \hat{x}^{(l)}(k+1))^T\}. $$

(12)

A. PCRLB for Centralized Architecture

The PCRLB inequality [2] states that the mean square error (MSE) associated with the estimate $\hat{x}(0:k)$ of the state vector $x(0:k)$ is lower bounded by

$$\mathbb{E}\{(x(0:k) - \hat{x}(0:k))(x(0:k) - \hat{x}(0:k))^T\} \geq \mathbb{E}[J(x(0:k))]^{-1}$$

Matrix $J(x(0:k))$ is referred to as the Fisher information matrix (FIM) [2], i.e., the inverse of the PCRLB, derived from the joint probability density $P(x(0:k), z(1:k))$. Let $\nabla$ and $\Delta$ denote, respectively, the operators for the first and second order partial derivatives as $\nabla_x(k) = \left[\frac{\partial}{\partial x_1(k)}, \ldots, \frac{\partial}{\partial x_{n_x}(k)}\right]^T$ and $\Delta_x(k) = \nabla_x(k-1)\nabla_x(k)^T$. A common form [2] of the FIM is defined as

$$J(x(0:k)) = \mathbb{E}\{-\Delta_x(0:k) \log P(x(0:k), z(1:k))\},$$

(14)
where the expectation is with respect to the joint distribution of the states and observations, i.e., \( P(\mathbf{x}(0: k), \mathbf{z}(1: k)) \). An alternative expression for the FIM is derived by expressing 

\[
P(\mathbf{x}(0: k), \mathbf{z}(1: k)) = P(\mathbf{x}(0: k)|\mathbf{z}(1: k))P(\mathbf{z}(1: k)).
\]

(15)

Since \( P(\mathbf{z}(1: k)) \) is assumed independent of the state, Eq. (15) leads to the following definition for the FIM.

**Definition 1.** The FIM for the state vector \( \mathbf{x}(0: k) \) from time 0 to \( k \) is given by 

\[
J(\mathbf{x}(0: k)) = \mathbb{E}\{ -\Delta_{\mathbf{x}(0: k)} \log P(\mathbf{x}(0: k)|\mathbf{z}(1: k)) \} 
\]

(16)

where the expectation is taken with respect to \( P(\mathbf{x}(0: k), \mathbf{z}(1: k)) \) and the integration is multidimensional depending on the state dimensions.

The global FIM \( J(\mathbf{x}(0: k)) \) is factorized as follows [2]

\[
J(\mathbf{x}(0: k)) \triangleq \begin{bmatrix} A(k) & B(k) \\ B^T(k) & C(k) \end{bmatrix} = 
\begin{bmatrix} \mathbb{E}\{ -\Delta_{\mathbf{x}(0: k-1)} \log P_c(k) \} & \mathbb{E}\{ -\Delta_{\mathbf{x}(0: k-1)} \log P_c(k) \} \\ \mathbb{E}\{ -\Delta_{\mathbf{x}(0: k-1)} \log P_c(k) \} & \mathbb{E}\{ -\Delta_{\mathbf{x}(0: k-1)} \log P_c(k) \} \end{bmatrix}.
\]

(17)

where \( P_c(k) = P(\mathbf{x}(0: k)|\mathbf{z}(1: k)) \). The FIM \( J(\mathbf{x}(k)) \) associated with the estimate \( \hat{x}(k) \) is obtained by taking the inverse of \( (n_x \times n_x) \) right-lower square block of \( J(\mathbf{x}(0: k)) \) using the following Lemma [20].

**Lemma 1.** Matrix inversion Lemma:

\[
\begin{bmatrix} A & B \\ B^T & C \end{bmatrix}^{-1} = \begin{bmatrix} \Omega^{-1} & -A^{-1}B\Phi^{-1} \\ -\Phi^{-1}B^T \Omega^{-1} & \Phi^{-1} \end{bmatrix},
\]

(18)

where subblocks \( \{A, B, C\} \) have conformable dimensions, \( \Omega = A - B\Phi^{-1}B^T \), and \( \Phi = C - B^T \Omega^{-1}B \).

Based on Lemma 1, the FIM \( J(\mathbf{x}(k)) \) is given by

\[
J(\mathbf{x}(k)) = C(k) - B(k)^T \text{Anderson}(k)^{-1} B(k),
\]

(19)

Corollary 1 (derived in [2]), presents the centralized sequential formulation of the FIM \( J(\mathbf{x}(k)) \) that requires a central fusion centre but without the need of computing the inverse of \( J(\mathbf{x}(0: k)) \) or the inverse of other large matrices, e.g., \( A(k) \).

**Corollary 1.** The centralized FIM \( \{J(\mathbf{x}(k))\} \) associated with the filtered estimate \( \hat{x}(k) \) follows the recursion

\[
J(\mathbf{x}(k+1)) = D^{22}(k) - D^{21}(k)J(\mathbf{x}(k)) + D^{11}(k) - D^{12}(k),
\]

(20)

where

\[
D^{11}(k) = \mathbb{E}\{ -\Delta_{\mathbf{x}(k)} \log P(\mathbf{x}(k+1)|\mathbf{x}(k)) \},
\]

(21)

\[
D^{12}(k) = D^{21}(k)^T = \mathbb{E}\{ -\Delta_{\mathbf{x}(k+1)} \log P(\mathbf{x}(k+1)|\mathbf{x}(k)) \},
\]

(22)

\[
D^{22}(k) = \mathbb{E}\{ -\Delta_{\mathbf{x}(k+1)} \log P(\mathbf{z}(k+1)|\mathbf{x}(k+1)) \} + \mathbb{E}\{ -\Delta_{\mathbf{z}(k+1)} \log P(\mathbf{z}(k+1)|\mathbf{x}(k+1)) \}.
\]

(23)

The initial condition is

\[
J(\mathbf{x}(0)) = \mathbb{E}\{-\Delta_{\mathbf{x}(0)} \log P(\mathbf{x}(0)) \}.
\]

(24)

In the following discussion, we derive the bound on the state prediction estimate \( \hat{x}(k+1|k) \) as defined below.

**Definition 2.** Term \( J(\mathbf{x}(0: k + 1| k)) \) denotes the FIM corresponding to the predicted estimate of \( \mathbf{x}(0: k + 1) \) derived from the prediction density \( P(\mathbf{x}(0: k + 1)|\mathbf{z}(1: k)) \).

As for \( J(\mathbf{x}(k)) \), the FIM \( J(\mathbf{x}(k + 1| k)) \) associated with the predicted estimate \( \hat{x}(k+1|k) \) can be computed by taking the inverse of the \((n_x \times n_x)\) right-lower block of \( J(\mathbf{x}(0: k + 1| k)) \). This procedure is computationally intense. Instead, we derive an alternative expression for computing \( J(\mathbf{x}(k + 1| k)) \) from \( J(\mathbf{x}(k)) \).

**Corollary 2.** The centralized FIM \( \{J(\mathbf{x}(k + 1| k))\} \) for the predicted estimate \( \hat{x}(k+1|k) \) follows the recursion

\[
J(\mathbf{x}(k+1|k)) = B^{22}(k) - D^{21}(k)J(\mathbf{x}(k)) + D^{11}(k) - D^{12}(k),
\]

(25)

where \( J(\mathbf{x}(k)) \) is derived from Corollary 1. Terms \( D^{11}(k), D^{12}(k), \) and \( D^{21}(k) \) are given by Eqs. (21)-(22) and the additional term

\[
B^{22}(k) = \mathbb{E}\{ -\Delta_{\mathbf{x}(k+1)} \log P(\mathbf{x}(k+1)|\mathbf{x}(k)) \}.
\]

The proof of Corollary 2 is included in Appendix A. In centralized estimation, where all raw observations are forwarded to the central processing unit (fusion centre) for processing, Corollary 1 provides a recursive procedure for updating \( J(\mathbf{x}(k)) \) without the need for computing \( J(\mathbf{x}(0: k)) \). The predicted FIM \( J(\mathbf{x}(0: k + 1| k)) \), when needed, can be obtained from \( J(\mathbf{x}(k)) \) using Corollary 2. A second configuration that uses a centralized fusion centre is the hierarchical architecture where each node communicates its local estimates or other statistics based on its local observations to the fusion centre. The latter forms the global estimate and updates the global posterior density \( P(\mathbf{x}(0: k)|\mathbf{z}(1: k)) \). Reference [15] shows that the PCRLB equations for the centralized architecture are also valid for the hierarchical architecture. Therefore, Corollaries 1 and 2 can be used for both centralized and hierarchical architectures.

The focus of this paper is on distributed estimation, where a fusion centre is not implemented and all processing is performed locally at the nodes constituting the network. We are interested in the PCRLB based resource management techniques to dynamically select a subset of candidate sensor nodes participating in distributed state estimation. Due to the absence of the fusion centre, such sensor selection approaches necessitate the PCRLB to be computed online in a distributed fashion as is discussed next.

### III. dPCRLB for Full-order Distributed Estimation

We categorize distributed state estimation in two classes: Full order distributed estimation that estimates the entire state vector at each node and reduced-order distributed estimation that estimates a subset of the state vector at a node. The distributed PCRLB (dPCRLB) algorithms for the two classes are considered below.
A. Distributed Full-Order Configuration

In the full-order distributed estimation, there are \( N \) processing nodes observing the dynamical system and estimating all the state variables. In other words, there are \( N \) local estimators. The distributed full-order model at node \( l \) is

\[
x(k) = f(x(k-1), \xi(k)) \quad (26)
\]

and

\[
z^{(l)}(k) = g^{(l)}(x(k), \zeta^{(l)}(k)), \quad (27)
\]

for \( 1 \leq l \leq N \). Eqs. (26) and (27) represent a full-order distributed estimator where the entire state vector \( x(k) \) is estimated at each node. In such scenarios, estimates at each node are computed based only on the local observations without a fusion centre. Further, each estimator only communicates with its neighboring nodes. Such a decentralized estimation architecture initially computes the state estimates and then fuses them in a distributed fashion to form the global estimate. Two different scenarios are considered for the distributed full-order systems.

1) Scenario 1. (Estimation based only on local measurements): Node \( l \), \( 1 \leq l \leq N \), updates its local estimates based on its individual measurements \( z^{(l)}(1:k) \). Local filtering distributions \( P(x(k)|z^{(l)}(1:k)) \) are then fused into the global posterior \( P(x(0:k)|z^{(1):k}) \) in a decentralized fashion using, for example, a gossip type algorithm.

2) Scenario 2. (Estimation based on local measurements and previous global estimate): Same as Scenario 1 except local estimates are based on both the local measurements as well as the previous global state estimates (which themselves are based on the collective observations made previously across the network). This leads to local filtering distributions \( P(x(k)|z^{(1:k-1), z^{(l)}(k)}) \) being computed at node \( l \). As in Scenario 1, the local filtering estimates \( P(x(k)|z^{(1:k-1), z^{(l)}(k)}) \) are then fused into the global posterior \( P(x(0:k)|z^{(1):k}) \) distributively.

Scenario 1 is useful for networks with intermittent connectivity where consensus on the local FIMs is not reached between two consecutive observations. In such cases, two filters are implemented for state estimation. The local filter updates the local states and local FIMs, while the global filter derives the overall state estimate and its associated FIM from their local counterparts. The local filters continue to assimilate local observations independently of the global filter. Once the global filter has converged, it incorporates the recent local state estimates and local FIMs to form the global state estimates.

Scenario 2 is useful in applications where communication is relatively inexpensive as compared to sensing, e.g., in rendezvous control or coordination of mobile sensors [58]. Consensus on the state estimates can be reached between two consecutive observations. With the availability of the global state estimate, local state estimates are discarded and the next iteration is continued based on the global estimates. Unlike Scenario 1, where the local FIM at iteration \( k+1 \) is computed using the local FIM at iteration \( k \), Scenario 2 updates the local FIM at iteration \( k+1 \) from the global FIM.

The problem we want to solve is to compute the theoretical lower bound, i.e., PCRLB, on the error in the global state estimate. Below, we explain our dPCRLB computation algorithm in terms of Scenario 1. Later, we show that the equations used to compute the global FIM as a function of the local FIMs in Scenario 1 are similar in nature to those for Scenario 2 with some modifications.

B. Full-order dPCRLB (FO/dPCRLB)

In this section, we derive the recursive expression for computing the full-order dPCRLB, i.e., express the global information sub-matrix, denoted by \( J_{\text{FO}}(x(k+1)) \), as a function of its value \( J_{\text{FO}}(x(k)) \) for the previous iteration, local FIMs \( J_{\text{FO}}^{(l)}(x(k+1)) \), and the local prediction FIMs \( J_{\text{FO}}^{(l)}(x(k+1)|k) \), \( 1 \leq l \leq N \). In Definitions 3 and 4, we define the local FIMs.

**Definition 3.** Term \( J_{\text{FO}}^{(l)}(x(0:k)) \), for \( 1 \leq l \leq N \), denotes the local FIM corresponding to the local state estimate of \( x(0:k) \) derived from the local posterior density \( P(x(0:k)|z^{(l)}(1:k)) \) for a full order local estimator defined as

\[
J_{\text{FO}}^{(l)}(x(0:k)) = \mathbb{E}\{-\Delta x^{(l)(0:k)}|\log P(x(0:k)|z^{(l)}(1:k))\}, \quad (28)
\]

where the expectation is with respect to \( P(x(0:k), z^{(l)}(1:k)) \).

**Definition 4.** Term \( J_{\text{FO}}^{(l)}(x(0:k+1)|k) \) denotes the local FIM corresponding to the local prediction estimate of \( x(0:k+1) \) derived from the local prediction density \( P(x(0:k+1)|z^{(l)}(1:k)) \) for a full order local estimator defined as

\[
J_{\text{FO}}^{(l)}(x(0:k+1)|k) = \mathbb{E}\{-\Delta x^{(l)(0:k+1)}|\log P(x(0:k+1)|z^{(l)}(1:k))\}, \quad (29)
\]

where the expectation is with respect to \( P(x(0:k), z^{(l)}(1:k)) \).

Note that the inverse of the local filtering FIM, i.e., \( [J_{\text{FO}}^{(l)}(x(k))]^{-1} \), is equal to the \((n_x \times n_x)\) right-lower block of \( [J_{\text{FO}}(x(0:k))]^{-1} \) as explained previously for \( J(x(k)) \) based on Lemma 1. The expressions for recursively computing \( J_{\text{FO}}^{(l)}(x(k)) \) are similar in nature to Eqs. (20)-(23) except that the likelihood function \( P(z(k+1)|x(k+1)) \) originally used in Eq. (23) is replaced by its corresponding local likelihood \( P(z^{(l)}(k+1)|x(k+1)) \), i.e.,

\[
J^{(l)}(z^{(l)}(k+1)) = \mathbb{E}\{-\Delta x^{(l)(k+1)}|\log P(z^{(l)}(k+1)|x(k+1))\}, \quad (30)
\]

where the expectation is with respect to \( P(x(0:k), z^{(l)}(1:k)) \). Similarly, computation of \( J_{\text{FO}}^{(l)}(x(k+1)|k) \) is also based on Proposition 1 except \( J(x(k)) \) gets replaced by \( J_{\text{FO}}^{(l)}(x(k)) \).

In deriving the optimal recursive expressions for computing the dPCRLB, we encounter yet another form of the local FIM (denoted by \( J_{\text{FO}}^{(l)}(x(k)) \)) associated with the local state estimate derived from the local filtering distribution \( P(x(k)|z^{(l)}(1:k)) \), i.e.,

\[
J_{\text{FO}}^{(l)}(x(k)) = \mathbb{E}\{-\Delta x^{(k)(l)}|\log P(x(k)|z^{(l)}(1:k))\}, \quad (31)
\]

where the expectation is with respect to \( P(x(0:k), z^{(l)}(1:k)) \) as before. Similarly, the prediction FIM \( J_{\text{FO}}^{(l)}(x(k+1)|k) \) associated with the local prediction estimate is given by

\[
J_{\text{FO}}^{(l)}(x(k+1)|k) = \mathbb{E}\{-\Delta x^{(k)(l)}|\log P(x(k+1)|z^{(l)}(1:k))\}, \quad (32)
\]
where the expectation is with respect to $P(x(0:k), z^{(l)}(1:k))$.

**Difference between $J_{FO}^{(l)}(x(k))$ and $\tilde{J}_{FO}^{(l)}(x(k))$:** For linear systems with Gaussian excitation, it has been shown in [20] that the two FIMs \{ $J_{FO}^{(l)}(x(k))$, $\tilde{J}_{FO}^{(l)}(x(k))$ \} are the same. For nonlinear systems, the two FIMs are generally different. An analytical comparison of $J_{FO}^{(l)}(x(k))$ with $\tilde{J}_{FO}^{(l)}(x(k))$ is difficult due to complicated integral terms. Note, however, that the localized FIM $J_{FO}^{(l)}(x(k))$ is obtained by inverting the $(n_x \times n_x)$ right lower square block of $[J_{FO}^{(l)}(x(0:k))]^{-1}$ using Eqs. (17)-(19) directly or its recursive implementation using Eq. (20). On the other hand, its counterpart $\tilde{J}_{FO}^{(l)}(x(k))$ is derived directly from Eq. (30) by taking the expectation and Laplacian of the local conditional posterior. A way of obtaining term $\tilde{J}_{FO}^{(l)}(x(k))$ is by re-initializing (rewriting) the system prior probability density function (PDF) at time $k$ with the posterior PDF, i.e., $P_0(x(k)) = F(x(k)|z^{(l)}(k))$. While $J_{FO}^{(l)}(x(k))$ can be computed recursively, determining $\tilde{J}_{FO}^{(l)}(x(k))$ is not generally straightforward [14]. Further explanation on the differences between $J_{FO}^{(l)}(x(k))$ and $\tilde{J}_{FO}^{(l)}(x(k))$ is presented in [14], [20]. A similar difference exists between the localized predictive FIMs $J_{FO}^{(l)}(x(k+1|k))$ derived from $J_{FO}^{(l)}(x(0:k))$ using Eq. (24) and $\tilde{J}_{FO}^{(l)}(x(k+1|k))$ obtained from Eq. (31).

**Scenario 1** (Estimation based only on local measurements): Theorem 1 presented below provides the optimal recursive formula for computing the distributed FIM corresponding to the global estimation from the local FIMs $J_{FO}^{(l)}(x(k))$ and local prediction FIMs $J_{FO}^{(l)}(x(k+1|k))$ for Scenario 1.

**Theorem 1.** The sequence \{ $J_{FO}^{(l)}(x(k))$ \} of information submatrices for the global estimates follows the recursion

\[
J_{FO}^{(l)}(x(k+1)) = C_{FO}^{(2)}(k) - C_{FO}^{(12)}(k)(J_{FO}^{(l)}(x(k)) + C_{FO}^{(11)}(k)^{-1}C_{FO}^{(12)}(k))
\]

where terms $C_{FO}^{(11)}(k)$, $C_{FO}^{(2)}(k)$, $C_{FO}^{(12)}(k)$ and $C_{FO}^{(22)}(k)$ are

\[
C_{FO}^{(11)}(k) = \mathbb{E}\{ -\Delta x^{(k)}_z \log P(x(k+1)|x(k)) \},
\]

\[
C_{FO}^{(12)}(k) = C_{FO}^{(2)}(k)^T = \mathbb{E}\{ -\Delta x^{(k+1)}_z \log P(x(k+1)|x(k)) \},
\]

\[
C_{FO}^{(22)}(k) = \sum_{l=1}^{N} J_{FO}^{(l)}(x(k+1)) - \sum_{l=1}^{N} J_{FO}^{(l)}(x(k+1|k))
\]

+ $\mathbb{E}\{ -\Delta x^{(k+1)}_z \log P(x(k+1)|x(k)) \}.
\]

In order to compute dPCRLB and specifically to compute $C_{FO}^{(22)}(k)$, we propose to replace $J_{FO}^{(l)}(x(k))$ with $\tilde{J}_{FO}^{(l)}(x(k))$ (and similarly $J_{FO}^{(l)}(x(k+1|k))$ with $\tilde{J}_{FO}^{(l)}(x(k+1|k))$) in Eq. (36), i.e.,

\[
C_{FO}^{(22)}(k) = \sum_{l=1}^{N} J_{FO}^{(l)}(x(k+1)) - \sum_{l=1}^{N} J_{FO}^{(l)}(x(k+1|k))
\]

+ $\mathbb{E}\{ -\Delta x^{(k+1)}_z \log P(x(k+1)|x(k)) \}.
\]

Note that Eq. (37) is an approximation given that $J_{FO}^{(l)}(x(k))$ may be different from $\tilde{J}_{FO}^{(l)}(x(k))$ for nonlinear systems. In our simulations for a nonlinear/Gaussian system, we illustrate through Monte Carlo simulations that Eq. (37) provides reasonably accurate results. The proof of Theorem 1 for Scenario 1 is included in Appendix B.

**Scenario 2** (Estimation based on local measurements and previous global estimate): We extend Theorem 1 to compute the global FIM as a function of the local FIMs for Scenario 2 where the local estimator at node $l$, for $1 \leq l \leq N$, is still restricted to local observations but additionally uses the previous estimated global state.

**Corollary 3.** Theorem 1 provides the optimal expression for Scenario 2 except for (36) involving $C_{FO}^{(22)}(k)$, which changes to

\[
C_{FO}^{(22)}(k) = \sum_{l=1}^{N} \mathbb{E}\{ -\Delta x^{(k)}_z \log P(x(k+1)|x(k)) \}
\]

where the first term on the right hand side (RHS) of Eq. (38) associated with the local state estimate is derived from the local filtering distribution $P(x(k)|z(1:k), z^{(l)}(1:k))$.

The proof of Corollary 3 is provided in Appendix C. In Eq. (38), we use the local FIM $J_{FO}^{(l)}(x(k+1))$ instead of the first term on the RHS of (38). Because Scenario 2 is based on the collective observations made previously across the network, the prediction FIM $J_{FO}^{(l)}(x(k+1|k))$ will be the same across the network, and the summation in the second term on the RHS of Eq. (38) reduces to $N$ times $J_{FO}^{(l)}(x(k+1|k))$. Eq. (38), therefore, simplifies to

\[
C_{FO}^{(22)}(k) = \sum_{l=1}^{N} J_{FO}^{(l)}(x(k+1|k)) - N J_{FO}^{(l)}(x(k+1|k))
\]

+ $\mathbb{E}\{ -\Delta x^{(k+1)}_z \log P(x(k+1)|x(k)) \},
\]

where we use the local FIM $J_{FO}^{(l)}(x(k+1))$ instead of the first term on the RHS of (38).

We make the following observations from Theorem 1.

- For updating $J_{FO}^{(l)}(x(k+1))$, reference [16] derives the following approximate expression

\[
J_{FO}^{(l)}(x(k+1)) = \sum_{l=1}^{N} J_{FO}^{(l)}(x(k+1|k)) - J_{FO}^{(l)}(x(k+1|k))
\]

+ $\mathbb{E}\{ -\Delta x^{(k+1)}_z \log P(x(k+1)|x(k)) \}
\]

(40)

There is one notable difference between Eq. (40) and Theorem 1. The third term on the RHS of (40) is based on the previous local FIM $J_{FO}^{(l)}(x(k))$ at node $l$ thus making it node-dependent. The corresponding term in Eq. (33) is based on the overall FIM from the previous iteration. When the PCRLB is computed in a distributed manner, Eq. (40) differs from one node to another. Theorem 1 is an exact result.

- Theorem 1 is optimal but computationally more intense than the approximated Eq. (40), which is the price paid for increased optimality.

- In additive Gaussian state-space models, the forcing term $\xi(k)$ and observation noise $\zeta^{(l)}(k)$ in Eqs. (26)-(27) are assumed to be uncorrelated and normally distributed.
Theorem 1 computes the FO/dPCRLB with communication restored. Assume that the global FIM needs to fuse local observations and can be computed as soon as the local FIMs are available. This typically happens in networks with intermittent communications. The local FIM includes no communication and can be computed as soon as the local observation is made. The global FIM needs to fuse local FIMs, which in this case will be possible only when communication is restored. Assume that the global FIM $J_{\text{FO}}(x(j))$ is available for iteration $j = (k+1-m)$ and the next fusion occurs at iteration $k+1$. For such a scenario, Theorem 1 is expressed in the form of the following corollary.

**Corollary 4.**

$$J_{\text{FO}}(x(k+1)) = C_{\text{FO}}^{22}(k)$$

$$-C_{\text{FO}}^{11}(k)J_{\text{FO}}(x(k+j)) + C_{\text{FO}}^{21}(k)$$

where $C_{\text{FO}}^{11}(k)$ and $C_{\text{FO}}^{21}(k) = [C_{\text{FO}}^{21}(k)]^T$ are given by Eqs. (34) and (35), and

$$C_{\text{FO}}^{22}(k) = \sum_{i=1}^{N} \left[ J_{\text{FO}}^{(i)}(x(k+1)) - J_{\text{FO}}^{(i)}(x(k+1)) \right]$$

$$+ E\left\{ -\Delta x_{x(k+1)} \log P(x(k+1)|x(k)) \right\}.$$ (45)

Term $J_{\text{FO}}(x(k+j))$ is the global $m$-step-ahead predictive FIM. Similarly, $J_{\text{FO}}(x(k+1))$ is the local predictive FIM. For more details on predictive FIMs refer to [54].

- Lack of invertibility of the local FIM $J_{\text{FO}}^{(i)}(x(k))$ indicates that the states are locally unobservable. This happens if the condition number $\kappa(J_{\text{FO}}^{(i)}(x(k)))$, i.e., the common logarithm of the ratio of its largest eigenvalue $\lambda_{\text{max}}^{(i)}$ to its smallest eigenvalue $\lambda_{\text{min}}^{(i)}$, is a large number. When the local FIM at node $l$ is singular, the node can not track the target on the basis of only its local observations. Therefore, it can not update its local FIM. In cases when the local FIM at node $l$ is not invertible, the dPCRLB algorithm drops node $l$ from the consensus step. Consensus is achieved using the remaining nodes. The local FIM $J_{\text{FO}}^{(l)}(x(k))$ at node $l$ is then updated using the global FIM obtained from the consensus step.

- Finally, we investigate the communication overhead for the FO/dPCRLB. When average consensus is used to distributively compute the summation terms in Eq. (33), the communication overhead is of $O(n^2 |N_c| |N|)$ at each node, where $n_c$ is number of states, $|N|$ the number of nodes in the neighborhood of node $l$ with $|\cdot|$ denoting the cardinality operator, and $N_c$ is the number of consensus iterations. The communication overhead for the approximate expression ((40)) is the same.

### C. Distributed Computation of the Full-order dPCRLB

Assume submatrices $J_{\text{FO}}^{(i)}(x(k))$, $J_{\text{FO}}^{(i)}(x(k+1))$, and $J_{\text{FO}}(x(k))$ are available from iteration $k$ of the dPCRLB update (or via initialization). Below, we explain iteration $k+1$ for updating the dPCRLB.

**Step 1:** Node $l$, for $1 \leq l \leq N$, computes terms $C_{\text{FO}}^{11}(k)$, $C_{\text{FO}}^{21}(k)$, and $C_{\text{FO}}^{22}(k)$ using Eqs. (34)-(35). Since these terms are based on the global state mode (Eq. (26)), they can be computed locally at each node without requiring any communication with the neighboring nodes.

**Step 2:** Compute term $C_{\text{FO}}^{22}(k)$ using (37). This involves the local FIMs $J_{\text{FO}}^{(i)}(x(k+1))$ and $J_{\text{FO}}^{(i)}(x(k+1))$ representing the bound on the local estimator at node $l$. Term $J_{\text{FO}}^{(l)}(x(k+1))$, for example, is computed by extending Corollary 1 to the distributed estimation model as

$$J_{\text{FO}}^{(l)}(x(k+1)) = [D_{\text{FO}}^{22}(k)]^{(l)} -$$

$$[D_{\text{FO}}^{21}(k)]^{(l)}J_{\text{FO}}^{(l)}(x(k))+[D_{\text{FO}}^{11}(k)]^{(l)})^{-1}[D_{\text{FO}}^{21}(k)]^{(l)},$$ (46)

$$[D_{\text{FO}}^{11}(k)]^{(l)} = E\left\{ -\Delta x_{x(k)} \log P(x(k+1)|x(k)) \right\}.$$ (47)

$$[D_{\text{FO}}^{12}(k)]^{(l)} = \left( [D_{\text{FO}}^{21}(k)]^{(l)} \right)^T =$$

$$E\left\{ -\Delta x_{x(k)} \log P(x(k+1)|x(k)) \right\}.$$ (48)

$$[D_{\text{FO}}^{22}(k)]^{(l)} = E\left\{ -\Delta x_{x(k)} \log P(x(k+1)|x(k)) \right\} +$$

$$E\left\{ -\Delta x_{x(k+1)} \log P(x(k+1)|x(k+1)) \right\}.$$ (49)

Scenario 2 replaces Eq. (46) with

$$J_{\text{FO}}^{(l)}(x(k+1)) = [D_{\text{FO}}^{22}(k)]^{(l)} -$$

$$[D_{\text{FO}}^{21}(k)]^{(l)}J_{\text{FO}}^{(l)}(x(k))+[D_{\text{FO}}^{11}(k)]^{(l)})^{-1}[D_{\text{FO}}^{21}(k)]^{(l)},$$ (50)

with the local FIM at iteration $k$ on the RHS of Eq. (46) replaced by the global FIM at iteration $k$. Note that Eqs. (46)-(49) only require information available locally at each node. The expression for computing $J_{\text{FO}}^{(l)}(x(k+1))$ is based on Corollary 2 expanded as follows

$$J_{\text{FO}}^{(l)}(x(k+1)) = [B_{\text{FO}}^{22}(k)]^{(l)} -$$

$$[B_{\text{FO}}^{21}(k)]^{(l)}J_{\text{FO}}^{(l)}(x(k))+[B_{\text{FO}}^{11}(k)]^{(l)})^{-1}[B_{\text{FO}}^{21}(k)]^{(l)}(51)$$

where $[B_{\text{FO}}^{22}(k)]^{(l)} = E\left\{ -\Delta x_{x(k+1)} \log P(x(k+1)|x(k)) \right\}. (52)$
Across the network, \( J_{\text{FO}}^{(i)}(x(k+1)) \) and \( J_{\text{FO}}^{(i)}(x(k+1|k)) \) will have different values. Having computed \( J_{\text{FO}}^{(i)}(x(k+1)) \) and \( J_{\text{FO}}^{(i)}(x(k+1|k)) \), the summation term involving \( x_{c1}^{(i)}(0) \) (43) is computed using average consensus [23] in a distributed fashion. Consensus algorithms are described in Appendix E.

The derivation of a summation term using average consensus algorithm requires information on the total number \( N \) of active nodes. Since the prime motivation for computing the dPCRLB is sensor selection, therefore, the number of active nodes should be known beforehand. We note that when \( N \) is unknown, an additional average consensus step with the value of one node set to 1 and others to 0 can instead be used to determine the number of nodes in the network. Average consensus will converge to \( 1/N \) and its reciprocal will provide the value of \( N \) at each local node. Please refer to [56] for a review of techniques based on this method. Node \( l \), for \( 1 \leq l \leq N \), initializes its consensus state as

\[
x_{c1}^{(i)}(0) = J_{\text{FO}}^{(i)}(x(k+1)) - J_{\text{FO}}^{(i)}(x(k+1|k))
\]

and continues to iterate

\[
x_{c1}^{(i)}(t+1) = x_{c1}^{(i)}(t) + \epsilon \sum_{j \in \mathcal{N}(i)} (x_{c1}^{(j)}(t) - x_{c1}^{(i)}(t))
\]

till convergence to

\[
x_{c1}(\infty) = \frac{1}{N} \sum_{i=1}^{N} \left( J_{\text{FO}}^{(i)}(x(k+1)) - J_{\text{FO}}^{(i)}(x(k+1|k)) \right)
\]

is achieved. In Eq. (54), \( \epsilon \) is a small value satisfying \( 0 < \epsilon \leq \frac{1}{2 \Delta \mathcal{G}_f} \) and \( \Delta \mathcal{G}_f = \max_l D^{(i)} \) is the maximum degree for fusion graph \( \mathcal{G}_f \) and \( D^{(i)} \) is the number of neighboring nodes for fusion node \( l \). Once the consensus converges, each fusion node substitutes the result of Eq. (55) in Eq. (37) to compute \( C_{\text{ Parl}}^{(2)}(k) \). Note that the consensus approach in Eq. (54) is a distributed algorithm where each node communicates only with its neighboring nodes. The final expectation term in (37) depends only on the state model and can be derived locally.

**Step 3:** Theorem 1 is now used to compute the dPCRLB, which is the same at all nodes.

Note that only Step 2 requires cooperation among the neighboring nodes achieved using a consensus algorithm across the network, while Steps 1 and 3 can be computed locally at each processing node. Finally, we note that when the dPCRLB is computed using average consensus algorithms with (i) the network being connected; (ii) fast connectivity allowing for consensus to be achieved between two consecutive observations, the proposed dPCRLB coincides with its centralized value. This is in fact exploited by the dPCRLB algorithm. We note that, assumptions (i) and (ii) are commonly used in the consensus-based literature related to distributed implementation of the particle filter and the Kalman filter [23], [24]. Such assumptions are reasonable in applications where compared to sensing communication is relatively inexpensive, e.g., in rendezvous control or coordination of mobile sensors.

### D. Particle Filter Realization for Full-order dPCRLB

In nonlinear dynamical systems, direct computation of \( \{ C_{\text{ Parl}}^{(1)}(k), C_{\text{ Parl}}^{(2)}(k), C_{\text{ Parl}}^{(21)}(k), C_{\text{ Parl}}^{(22)}(k) \} \) as well as localized terms \( \{ [D_{\text{ Parl}}^{(1)}(k)]^{(i)}, [D_{\text{ Parl}}^{(2)}(k)]^{(i)}, [D_{\text{ Parl}}^{(21)}(k)]^{(i)}, [D_{\text{ Parl}}^{(22)}(k)]^{(i)} \} \) is difficult due to the involvement of nonlinear terms within the expectation operator [51]. Sequential Monte Carlo methods (such as the particle filter [52], [55]) are usually used to compute these terms. For completeness, the following section explains how the expectation terms in the FO/dPCRLB are computed using particle filters. The particle filter uses a set of \( N_p \) ‘vector particles’ \( x_i(k) \) and associated weights \( W_i(k) \), for \( 1 \leq i \leq N_p \), to estimate the state vector \( x(k) \). The overall posterior of the state vector at time iteration \( k \) is expressed in terms of the particles and their associated weights as

\[
P(x(k)|z(1:k)) \approx \sum_{i=1}^{N_p} W_i^{(i)}(k) \delta(x(k) - x_i^{(i)}(k)). \tag{56}
\]

where \( \delta(\cdot) \) denotes the delta function. Recently, a number of distributed implementations of the particle filter have been proposed [24]-[34], where the particle filter is partitioned into several localized filters. Each local filter (associated with a single or a small combination of nodes) incorporates its local measurements. In general, an additional higher order particle filter (referred to as the global/fusion/consensus filter) is introduced that assimilates the local statistics from these local filters into global statistics2. In the sequel, \( \{ x_i^{(l,GF)}(k), W_i^{(l,GF)} \} \) refers to the global particle set computed at node \( l \), for \( 1 \leq l \leq N_x \), using the higher order global filter. In a general case, the global particle set and associated weights can be used to implement Steps 1-3 of the full order dPCRLB computational algorithm described in Section III-C. For the sake of completeness, we summarize Eqs. (34)-(36) in terms of the global particle set \( \{ x_i^{(l,GF)}(k), W_i^{(l,GF)} \} \) of the distributed particle filter followed by their equivalent representation for the case where the forcing terms are additive Gaussian. Superscript GF used in \( \{ x_i^{(l,GF)}(k), W_i^{(l,GF)} \} \) associates the weighted particles to the global filter that assimilates local statistics into global statistics. Representing (34) in terms of the global particle filter set, we get

\[
\hat{C}_{\text{ Parl}}^{(11)}(k) \approx - \sum_{i=1}^{N_p} W_i^{(l,GF)}(k) \times \left( \Delta \mathcal{X}_i^{(k)} \log P(x(k+1)|x(k))) \right) \bigg|_{x(k)=x_i^{(l,GF)}(k)}.
\]

For the additive Gaussian forcing terms, Eq. (57) simplifies to

\[
\hat{C}_{\text{ Parl}}^{(11)}(k) \approx \sum_{i=1}^{N_p} W_i^{(l,GF)}(k) \times \left( \left( \nabla_{x(k)} f^T(x(k)) \right) Q^{-1}(k) \left( \nabla_{x(k)} f^T(x(k)) \right)^T \right) \bigg|_{x(k)=x_i^{(l,GF)}(k)}.
\]

2 Other distributed implementations of the particle filter [26]-[34] do not maintain separate local and global particle sets. Only one set of particles is maintained. Information or statistics from local particle sets is then fused in a distributed way to update the particle set to better represent the global posterior. The proposed distributed computation of dPCRLB is also applicable in such cases as long as the global particle sets are available at each node.
Similarly, Eq. (35) in terms of the global particle set is
\[ \hat{C}_{F/O}^{12}(k) = \left[ \hat{C}_{F/O}^{22}(k) \right]^T \approx \]
\[ - \sum_{i=1}^{N_p} W_i^{(GF)}(k) \left( \Delta x_i^{(k)} \log P(x(k+1)|x(k)) \right) \bigg|_{x_i(x) = x_i^{(GF)}(k)} \]
which for the additive Gaussian forcing terms simplifies to
\[ \hat{C}_{F/O}^{12}(k) = \left[ \hat{C}_{F/O}^{22}(k) \right]^T \approx \]
\[ - \sum_{i=1}^{N_p} W_i^{(GF)}(k) \times \left( \left[ \nabla_{x_i} f^T(x_i(k)) \right] Q^{-1}(k) \right) \bigg|_{x_i(x) = x_i^{(GF)}(k)} \]
Term \( C_{F/O}^{22}(k) \) in Eq. (37) requires participation of all the local fusion nodes to compute the submatrices \( J_{F/O}^{(l)}(x_i(k+1)) \) and \( J_{F/O}^{(l)}(x_i(k+1)) \) of the local FIM. Submatrix \( J_{F/O}^{(l)}(x_i(k+1)) \) is computed based on Eq. (46) with terms \( D_{F/O}^{11}(k) \), \( D_{F/O}^{12}(k) \), and \( D_{F/O}^{22}(k) \) having particle filter representations similar to the ones expressed for Eqs. (34)-(35), which are omitted here to save on space. Below, we write these terms for the Gaussian case
\[ \hat{D}_{F/O}^{11}(k)^{(l)} \approx \sum_{i=1}^{N_p} W_i^{(GF)}(k) \]
\[ \times \left( \left[ \nabla_{x_i} f^T(x_i(k)) \right] Q^{-1}(k) \left[ \nabla_{x_i} f^T(x_i(k)) \right]^T \right) \bigg|_{x_i(x) = x_i^{(GF)}(k)} \]
\[ \hat{D}_{F/O}^{12}(k)^{(l)} \approx \left[ \hat{D}_{F/O}^{22}(k)^{(l)} \right]^T \approx \]
\[ \sum_{i=1}^{N_p} W_i^{(GF)}(k) \left( \left[ \nabla_{x_i} f^T(x_i(k)) \right] Q^{-1}(k) \right) \bigg|_{x_i(x) = x_i^{(GF)}(k)} \]
\[ \hat{D}_{F/O}^{22}(k)^{(l)} \approx Q^{-1}(k) + \sum_{i=1}^{N_p} W_i^{(GF)}(k) \]
\[ \left( \nabla_{x_i(k+1)} g_i^{(l)}(k+1) \right) R^{-1}(k+1) \left( \nabla_{x_i(k+1)} g_i^{(l)}(k+1) \right)^T \bigg|_{x_i(x) = x_i^{(GF)}(k+1)} \]
where particles \( x_i^{(GF)}(k+1) \) are computed by propagating particles \( x_i^{(GF)}(k) \) through the transitional density \( P(x(k+1)|x(k)) \) obtained from the state equation (Eq. (26)). Note that the required terms in Eqs. (61)-(63) are computed based on the available particles for iteration \( k \). Eqs. (51)-(52) are then used to compute \( J_{F/O}^{(l)}(x_i(k+1)) \).

The aforementioned procedure using particles and weights associated to the distributed particle filter can readily be extended to non-Gaussian forcing terms.

IV. dPCRLB FOR REDUCED-ORDER DISTRIBUTED ESTIMATION

Section IV extends the dPCRLB computational approaches in Section III to distributed, reduced order systems. We start our discussion by introducing the reduced order systems.

A. Distributed Reduced-Order Configuration

In large-scale physical systems arising, for example, in meteorology, physical oceanography, or resulting from discretization of partial differential equations, the discretized dynamical models are sparse and localized. The observation \( z_i^{(l)}(k) \) made at node \( l \), for \( 1 \leq l \leq N \), is also localized such that a subset of state variables \( x_i^{(l)}(k) \subset x_i(k) \) (referred to as the local state vector) is observed at each node [35]. For such reduced-order systems \( S_i^{(l)} \), the observation model (Eq. (3)) for node \( l \) reduces to
\[ S_i^{(l)}: \quad z_i^{(l)}(k) = g_i^{(l)} \left( x_i^{(l)}(k) \right) + c_i^{(l)}(k). \]

The local state vectors in the above equation may have shared states, i.e., \( |x_i^{(l)}(k) \cap x_j^{(l)}(k)| \geq 0, 1 \leq l, j \leq N \). The reduced-order state-space model is obtained by spatially decomposing the overall system based on the observable states at each node. Other states, if present, are treated as forcing terms. The reduced-order state model at node \( l \) is then given by
\[ S_i^{(l)}: \quad x_i^{(l)}(k) = f_i^{(l)}(x_i^{(l)}(k-1), d_i^{(l)}(k-1)) + f_i^{(l)}(k). \]

where \( d_i^{(l)}(k) \) is the coupling state vector. When the overall system is partitioned into subsystems\(^1\), the dynamical model for a subsystem may contain states that are directly observed by the subsystem and additional states that are not observed but are part of the global state model. The coupling state vector \( d_i^{(l)}(k) \) includes such states which are not directly observed but are part of the subsystem’s model. Let \( n_{x/l} \) denote the number of states in the local state vector \( x_i^{(l)}(k) \). The relationship between the local state vector \( x_i^{(l)}(k) \) and global vector \( x_i(k) \) can be expressed as
\[ x_i^{(l)}(k) = T^{(l)}(k)x_i(k), \]
with \( T^{(l)}(k) \) denoting the \( (n_{x/l} \times n_{x_i}) \) nodal transformation matrix [40]. The local process functions are constructed using a similar nodal transformation, i.e., \( f_i^{(l)}(x_i^{(l)}(k), d_i^{(l)}(k)) = T^{(l)}(k)f_i(x_i(k)) \). The local state estimate at node \( l \) has the same relation to the global state estimate, i.e., \( \hat{x}_i^{(l)}(k) = T^{(l)}(k)\hat{x}_i(k) \). Further, the relationship between the global covariance \( P(k) \) and local covariance matrix \( P_i^{(l)}(k) \) is
\[ P_i^{(l)}(k) = T^{(l)}(k)P(k)T^{(l)}(k)^T. \]
To arrange node \( l \)’s information \( P_i^{(l)}(k) \) in the global state-space, we use the covariance transformation
\[ \hat{P}_i^{(l)}(k) = \left[ T^{(l)}(k) \right]^+ \hat{P}_i^{(l)}(k) \left[ T^{(l)}(k) \right]^+ \]
where \( \left[ T^{(l)}(k) \right]^+ \) refers to the Moore-Penrose generalized inverse (or the right pseudo inverse) of \( T^{(l)}(k) \), i.e., \( \left[ T^{(l)}(k) \right]^+ = T^{(l)}(k) \left[ T^{(l)}(k)T^{(l)}(k)^T \right]^{-1} \). Subsystems \( S_i^{(l)} \) and \( S_i^{(l)} \) may have shared states. The shared state transformation matrix \( T^{(l,j)}(k) \) is a \( (n_{x/l} \times n_{x_j}) \) matrix where \( x_i^{(l)} \cup x_j^{(l)} \) is the number of shared states between subsystems \( S_i^{(l)} \) and \( S_i^{(l)} \). Each row of \( T^{(l,j)}(k) \) has only one non-zero entry at the location of the shared states. The shared state transformation matrix \( T^{(l,j)}(k) \) is used to extract the covariance block
\[ \hat{P}^{(l,j)}(k) = \left[ T^{(l,j)}(k)T^{(l,j)}(k)^T \right] \hat{P}_i^{(l)}(k) \left[ T^{(l,j)}(k)T^{(l,j)}(k)^T \right]^T. \]

\(^1\)In this paper, each node is classified as a subsystem. Without any loss in generality, a combination of nodes can also be grouped together to form a subsystem. The results presented here are applicable to both cases.
corresponding to the shared states. To arrange the covariance block \( \hat{P}^{(l,j)}(k) \) corresponding for the shared states in the global state space \( \hat{P}_{G}^{(l,j)}(k) \), the following covariance transformation (similar to (68)) is used

\[
\hat{P}^{(l,j)}(k) = \left[ T^{(l,j)}(k)^{T} \right] \hat{P}^{(l,j)}(k) \left[ T^{(l,j)}(k) \right]^{T}.
\]  

(70)

To recap, the process model (65) and observation model (64) collectively provide the nonlinear, localized reduced-order representation for the dynamical system.

For an example of a reduced-order system, please refer to [36] where transitional matrices are derived for a simplified reduced-order system. As a final note, we observe that the above result extends the linear partitioning approach of Moura et al. [35] to nonlinear systems.

B. Reduced-order dPCRLB (RO/dPCRLB)

In this section, we derive the recursive expression for computing the dPCRLB for reduced-order configured systems. The problem we wish to solve is to express the global information sub-matrix, denoted by \( \hat{J}_{RO}(x(k+1)) \), in terms of its previous iterate \( \tilde{J}_{RO}(x(k)) \), local FIMs \( J_{RO}(x(k)) \), and local prediction FIMs \( J_{RO}(x(k+1)) \), for \( 1 \leq l \leq N \). The reduced order local FIMs are defined below.

Definition 5. Term \( J_{RO}(x(0 : k), \{z_l \}) \), for \( 1 \leq l \leq N \), denotes the local FIM corresponding to the local estimate of \( x(0 : k) \) derived from the local posterior density \( P(x(0 : k) \mid z(1 : k)) \). We define \( J_{RO}(x(0 : k)) \) as the FIM submatrix for estimating \( x(0 : k) \) given \( z(1 : k) \).

Definition 6. Term \( J_{RO}(x(0 : k+1|k)) \) denotes the local FIM corresponding to the local prediction estimate of \( x(0 : k+1) \) derived from the local posterior density \( P(x(0 : k+1) \mid z(1 : k)) \). Term \( J_{RO}(x(k+1|k)) \) is similarly defined as the FIM submatrix for estimating \( x(k+1) \) given \( z(1 : k) \).

As for the full-order system, the inverse of the local filtering FIM, i.e., \( [J_{RO}(x(0 : k))]^{-1} \), is equal to the \( n_{x} \times n_{x} \) right-lower block of \( [J_{RO}(x(0 : k))]^{-1} \). In deriving the recursive expression for computing the reduced-order dPCRLB, we encounter a second form of the local reduced-order FIM (denoted by \( \tilde{J}_{RO}(x(0 : k)) \)) as the bound on the local filtering distribution \( P(x(0 : k) \mid z(1 : k)) \), i.e.,

\[
\tilde{J}_{RO}(x(0 : k)) = \mathbb{E}\{-\Delta_{x(0 : k)}^{2}(k) \log P(x(0 : k) \mid z(1 : k)) \}.
\]  

(71)

The inverse of the prediction FIM \( J_{RO}(x(k+1|k)) \) is given by the inverse of the \( n_{x} \times n_{x} \) right-lower block of \( J_{RO}(x(0 : k+1|k))^{-1} \). The bound on the local prediction is

\[
\tilde{J}_{RO}(x(k+1|k)) = \mathbb{E}\{-\Delta_{x(k+1)}^{2}(k) \log P(x(k+1) \mid z(0 : k)) \}.
\]  

(72)

Next, we present Theorem 2 that forms the basis of the optimal recursive algorithm for updating \( J_{RO}(x(k)) \).

Theorem 2. The reduced-order FIM \( \{J_{RO}(x(k))\} \) for the filtering estimate \( \hat{x}(k) \) follows the recursion

\[
\hat{J}_{RO}(x(k+1|k)) = C_{RO}^{22}(k)[J_{RO}(x(k))] + C_{RO}^{11}(k)^{-1} J_{RO}^{12}(k)
\]  

(73)

where

\[
C_{RO}^{11}(k) = \mathbb{E}\{-\Delta_{x(k)}^{2}(k) \log P(x(k+1) \mid x(k)) \},
\]  

(74)

\[
C_{RO}^{12}(k) = [C_{RO}^{(1)}(k)]^{T} \mathbb{E}\{-\Delta_{x(k)}^{2}(k) \log P(x(k)+1) \mid x(k)) \},
\]  

(75)

\[
C_{RO}^{22}(k) = \mathbb{E}\{-\Delta_{x(k)}^{2}(k) \log P(x(k+1) \mid x(k)) \} + \sum_{l=1}^{N} \left[ (T(l))^{T} \tilde{J}_{RO}(x(l) \mid k+1)) - \tilde{J}_{RO}(x(l) \mid k+1)) \right] [T(l)^{T} \tilde{J}_{RO}(x(l) \mid k+1)]]^{T}.
\]  

(76)

Derived for reduced-order estimation, Theorem 2 is similar in nature to Theorem 1 for the full-order dPCRLB (Eqs. (34)-(35)) except for \( C_{RO}^{12}(k) \) which involves local reduced-order FIMs \( \tilde{J}_{RO}(x(l)) \) and \( \tilde{J}_{RO}(x(l+1)) \). Terms \( C_{RO}^{11}(k) \), \( C_{RO}^{12}(k) \) and \( C_{RO}^{22}(k) \) are the same as their counterparts and still based on the overall state model. As for full-order systems, terms \( \tilde{J}_{RO}(x(l+1)) \) and \( \tilde{J}_{RO}(x(l+1)) \) are approximated by their counterparts \( J_{RO}(x(l+1)) \) and \( J_{RO}(x(l+1)) \). Later in this section, we investigate how to compute these terms locally within each reduced-order subsystem. The proof of Theorem 2 is included in Appendix D.

C. Reduced-order Computation of RO/dPCRLB

In order to compute the RO/dPCRLB, one approach is to follow the steps listed for the full-order scenario in Section III-C. This will result in the global FIM at each node. In a reduced-order system, the processing nodes do not have access to the global model nor estimates for all states, therefore, such an approach is impractical. Instead, we propose computation of a block of FIM that corresponds to the states local at a node. In our approach, subsystem \( l \) computes the diagonal block \( J_{RO}^{G}(x(l+1)) \) of the FIM \( J_{RO}(x(l+1)) \) corresponding to its local states \( x(l) \). The FIM block for \( x(l) \) is

\[
J_{RO}^{G}(x(l+1)) = T(l,k) J_{RO}(x(l+1)) T(l,k)^{T}, \]  

(77)

where \( T(l,k) \) denotes the \( (n_{x(l)} \times n_{x}) \) transformation matrix. Exploiting the block banded structure of the global FIM, the dPCRLB for the local states is then computed from the local FIM block and the adjacent blocks obtained from the neighboring nodes. This is explained later in Step 3.

We first outline the procedure for updating FIM block \( \hat{J}_{RO}(x(k+1)) \) at node \( l \). Using Theorem 2, Eq. (77) is expanded as follows

\[
\hat{J}_{RO}(x(k+1)) = [C_{RO}^{22}(k)]^{T} - T(l,k) C_{RO}^{21}(k) [J_{RO}^{G}(x(l)) + C_{RO}^{11}(k)]^{-1} T(l,k) C_{RO}^{12}(k)
\]  

(78)

where \( [C_{RO}^{22}(k)]^{T} = T(l,k) C_{RO}^{22}(k) T(l,k)^{T} \). Next we describe the steps required to compute Eq. (78) in a distributed reduced-order fashion.

Step 1: In order to compute \( [C_{RO}^{22}(k)]^{T} \), node \( l \), for \( 1 \leq l \leq N \), needs to compute local FIM blocks \( J_{RO}(x(l+1)) \)
and $J_{RO}(x^{(l)}(k+1))$. Based on Corollary 1 (following the procedure for derivation of Eq. (46)), we get

$$J_{RO}(x^{(l)}(k+1)) = [D_{RO}^{22}(k)]^{(l)} - \left[D_{RO}^{12}(k)]^{(l)} (J_{RO}(x^{(l)}(k)) + [D_{RO}^{12}(k)]^{(l)})^{-1} [D_{RO}^{12}(k)]^{(l)} \right.$$

with

$$[D_{RO}^{11}(k)]^{(l)} = -\mathbb{E}_{x^{(l)}} \left[ -\Delta \log P(x^{(l)}(k+1)|x^{(l)}(k), d^{(l)}(k)) \right],$$

$$[D_{RO}^{12}(k)]^{(l)} = -\mathbb{E}_{x^{(l)}} \left[ -\Delta \log P(x^{(l)}(k+1)|x^{(l)}(k), d^{(l)}(k)) \right],$$

$$[D_{RO}^{22}(k)]^{(l)} = -\mathbb{E}_{x^{(l)}} \left[ -\Delta \log P(x^{(l)}(k+1)|x^{(l)}(k), d^{(l)}(k)) \right].$$

The local predictive FIM is similarly derived from Eq. (52) and is given by

$$J_{RO}(x^{(l)}(k+1)) = [D_{RO}^{22}(k)]^{(l)} - \left[D_{RO}^{12}(k)]^{(l)} (J_{RO}(x^{(l)}(k)) + [D_{RO}^{12}(k)]^{(l)})^{-1} [D_{RO}^{12}(k)]^{(l)} \right.$$

where

$$[B_{RO}^{22}(k)]^{(l)} = -\mathbb{E}_{x^{(l)}} \left[ -\Delta \log P(x^{(l)}(k+1)|x^{(l)}(k), d^{(l)}(k)) \right].$$

Note that terms $[D_{RO}^{11}(k)]^{(l)}$, $[D_{RO}^{12}(k)]^{(l)}$, $[D_{RO}^{22}(k)]^{(l)}$, and $[B_{RO}^{22}(k)]^{(l)}$ are based on reduced-order models and can be computed locally.

**Step 2:** Having computed the local FIMs $J_{RO}(x^{(l)}(k+1))$ and $J_{RO}(x^{(l)}(k+1))$, node $l$ computes $C_{RO}^{22}(k)$ with a modified version of Eq. (76) where the summation is limited to neighboring nodes of node $l$ with which it has shared states. Due to the sparse and localized nature of the process model, only the neighboring nodes of subsystem $l$ have shared states with node $l$. Therefore, the communication and computational overheads for the distributed computation of $C_{RO}^{22}(k)$ is limited to its local neighborhoods.

**Step 3:** The next step is to compute the second term on the RHS of Eq. (78). First, note that because the local state model at node $l$ only includes a subset of state variables, $x^{(l)}(\cdot)$ and $d^{(l)}(\cdot)$, derivations with respect to $x(\cdot)$ will result in a block of zero terms corresponding to the states not present in the local state model. Therefore, $T^{(l)}(k)C_{RO}^{22}(k)$ is partitioned as

$$\begin{bmatrix} C_{RO}^{22}(k) \end{bmatrix}^{(l)} = \begin{bmatrix} [C_{RO}^{22}(k)]^{(l,l)}; [C_{RO}^{22}(k)]^{(l,d)}; [C_{RO}^{22}(k)]^{(d,l)}; [C_{RO}^{22}(k)]^{(d,d)} \end{bmatrix},$$

with

$$[C_{RO}^{22}(k)]^{(l,l)} = T^{(l)}(k)C_{RO}^{22}(k)T^{(l)}(k)^T$$

and

$$[C_{RO}^{22}(k)]^{(l,d)} = T^{(l)}(k)C_{RO}^{22}(k)T^{(d,l)}(k)^T$$

$$[C_{RO}^{22}(k)]^{(d,l)} = T^{(l)}(k)C_{RO}^{22}(k)T^{(l,d)}(k)^T$$

$$[C_{RO}^{22}(k)]^{(d,d)} = T^{(l)}(k)C_{RO}^{22}(k)T^{(d,d)}(k)^T$$

Matrix $T^{(l,d)}(k)$ denotes the $n_{d}(l) \times n_{x}(l)$ nodal transformation matrix corresponding to the $n_{d}(l)$ required forcing terms $d^{(l)}(k)$ at node $l$. Second, based on the above partitioning, a subdivision of matrix $S(k)$ is constructed as follows

$$S(k) = \begin{bmatrix} S^{(l,l)}(k) & S^{(l,d)}(k) \\ S^{(d,l)}(k)^T & S^{(d,d)}(k) \end{bmatrix}$$

**Step 4:** Finally, Eq. (78) is used to update $J_{RO}^{G}(x^{(l)}(k+1))$ at node $l$, for $1 \leq l \leq N$. The convergence of the proposed computational algorithm for estimating a sub-block of the global FIM corresponding to the local state subset is guaranteed by the convergence properties of the DICI-OR algorithm. See [35] for details.

**D. Computing the RO/dPCRLB from localized FIM**

The inversion algorithm for block banded matrices can be used to compute the RO/dPCRLB (i.e., to compute inverse of the FIM). One such approach, referred to as the DICI-RO, is explained in Appendix F. Note that the FIM is a full matrix and the RO/dPCRLB algorithm suggested in Section IV-C updates only its diagonal block entries. This may result in some variation in the RO/dPCRLB as compared to the approach suggested in Section III. The accuracy of the block-banded FIM approach can be improved by computing the off-diagonal blocks, which will add additional computation overhead. In this paper, we limit ourselves to obtaining the RO/dPCRLB from the diagonal blocks of the FIM.

**E. Particle Filter Realization for reduced-order dPCRLB**

The particle-based computation of the dPCRLB equations for the reduced-order systems is similar to the full-order scenario (Section III.1) except for the following differences.
At subsystem $l$, derivations in Eq. (57)-(63) are now based on the local state vector $x^{(l)}(\cdot)$. A reduced-order distributed implementation of the particle filter is employed to compute the required particle set $\{X_i^{(l,GP)}(k), W_i^{(l,GP)}\}$. For example, Eq. (80) can be represented in terms of the reduced-order particle sets as

$$[\hat{D}_{RO}^{(l)}(k)]^{(l)} \approx - \sum_{i=1}^{N_p} W_i^{(l,GF)}(k) \times \left( \frac{\Delta x_k^{(l)}(k) \log P(x_k^{(l)}(k+1)|x_k^{(l)}(k))}{|x_k^{(l)}(k) = X_i^{(l,GP)}|} \right),$$  

(89)

Although not included here, Eqs. (81)-(83) can similarly be represented in terms of the reduced-order global particle sets.

As a final note to our dPCRLB implementations, we note the differences between Theorem 1 (the dPCRLB algorithm for full-order systems) and Theorem 2 (the dPCRLB algorithm for reduced-order systems). Theorem 1 is applicable when the estimates of the entire state vector is available locally at each node. In reduced-order estimation, a different subset of the state vector is estimated at the local nodes. Eq. (35) included in Theorem 1 cannot be implemented in the reduced-order systems and is replaced by Eq. (75) which allows for reduced-order FIMs corresponding to different subsets of the state vector to be fused to determine the overall FIM. In the reduced-order format, Theorem 2 includes Eqs. (73)-(74) which are similar to Eqs. (33)-(34). In reality, reduced-order systems can not compute Eqs. (73)-(74) which are similar to Eqs. (33)-(34). In reality, reduced-order systems can not compute Eqs. (73)-(74) directly which requires the entire state vector to be known at each node. In Section IV.C, we discussed how Eqs. (73)-(74) in Theorem 2 are computed in a reduced-order fashion.

V. SIMULATIONS

In this section, we run Monte Carlo simulations to determine the accuracy of the proposed dPCRLB expressions for full-order (Theorem 1, Section III-B) and reduced-order (Theorem 2, Section IV-B) systems by comparing them with the results obtained using the centralized PCRLB (Corollary 1). A.

**Full-order Systems** A distributed bearing-only target tracking (BOT) application is used to demonstrate the accuracy of the proposed full-order dPCRLB. The state vector is given by $x(k) = [X(k), Y(k), \dot{X}(k), \dot{Y}(k)]$ where $\{X(k), Y(k)\}$ is the coordinates of the target at iteration $k$ and $\{\dot{X}(k), \dot{Y}(k)\}$ are the velocity components of the target. The nonlinear state model is

$$x(k+1) = f(x(k))x(k) + \Delta \xi(k+1)$$  

(90)

where the target’s motion model $f(x(k))$ [53] is given by

$$f(x(k)) = \begin{bmatrix} 1 & 0 & \sin(\Omega(k)\Delta T) & -1 - \cos(\Omega(k)\Delta T) \\ 0 & 1 & 1 - \cos(\Omega(k)\Delta T) & \sin(\Omega(k)\Delta T) \\ 0 & 0 & \cos(\Omega(k)\Delta T) & -\sin(\Omega(k)\Delta T) \\ 0 & 0 & \sin(\Omega(k)\Delta T) & \cos(\Omega(k)\Delta T) \end{bmatrix}$$  

(91)

with

$$\Omega(k) = \frac{A_{mz}}{\sqrt{(\dot{X}(k))^2 + (\dot{Y}(k))^2}},$$

and

$$\mathbf{A} = \begin{bmatrix} 0.5\Delta T^2 & 0 & \Delta T & 0 \\ 0 & 0.5\Delta T^2 & 0 & \Delta T \\ \Delta T & 0 & 0 & 0 \end{bmatrix}.$$  

The discretization time interval $\Delta T$ equals 1s and the maneuver acceleration parameter $A_m$ is set to $1.08 \times 10^{-3}$m/s$^2$ [53]. The area under surveillance is $(16 \times 16)$ m$^2$. Within the sensor network each sensor communicates only within a connectivity radius of $\sqrt{2 \log(N)/N}$ meters. In addition, the network is assumed to be connected with each node linked to at least one other node in the network. The measurement $z^{(l)}(k)$ available at node $l$ is the target’s bearings with respect to the node’s platform referenced (clockwise positive) to the $y$-axis, i.e.,

$$z^{(l)}(k) = \arctan\left( \frac{X(k) - X^{(l)}}{Y(k) - Y^{(l)}} \right) + \zeta^{(l)}(k),$$  

(92)

where $(X^{(l)}, Y^{(l)})$ are the coordinates of node $l$. The process and observation noises are normally distributed, i.e.,

$$\xi(k) \sim N(0, Q) \quad \text{and} \quad \zeta(k) = \{\zeta^{(l)}(k)\}_{l=1}^{N} \sim N(0, R).$$

Furthermore, the observation noise is considered as state dependent such that the bearing noise variance $\sigma_{\xi}^{2}$ at node $l$ depends on the distance $r^{(l)}(k)$ between the observer and target as follows [57]

$$\sigma_{\xi}^{2}(k) = B_{m}r^{(l)}(k)^2 + 0.1150r^{(l)}(k) + 0.7405,$$  

(93)

where different values for parameter $B_m$ are used to test various signal to noise ratios (SNR). Please refer to [57] for the choice of coefficients used in Eq. (93). In other words, $R(k)$ is a diagonal matrix with bearing noise variance $\sigma_{\xi}^{2}$ as its diagonal elements. In each run, the target starts its track from coordinates $[3, 6]$, with the initial course set at $-120^\circ$ with the standard deviation of the process noise $\sigma_{\xi} = 1.6 \times 10^{-2}$ meter. Matrix $Q$ is also a diagonal matrix given by $\sigma_{\xi}^2 I$ with $I$ denoting the $(2 \times 2)$ identity matrix. As stated in [53], the overall process noise $A\xi(k+1)$ in Eq. (90) corresponds to a piecewise constant white acceleration noise model. Our comparison for full-order systems is based on results from three implementations schemes: (i) The centralized PCRLB (Corollary 1); (ii) proposed FO/dPCRLB approach (Sections III-C and III-D); (iii) approximated expression for dPCRLB given in [16] (Eq. (40)). The following two scenarios
Scenario 1. The first simulation is based on a fixed target trajectory (i.e., the same track is used in each Monte Carlo run) and the true value of the state variables (as in [53]). The proposed algorithm for full-order systems outlined in Section III-C is used to compute the dPCRLB. In Step 3, Theorem 1 based on Eqs. (41)-(43) is used. The expectation operators disappear with the required Jacobians computed using the true trajectories. This is a test case included to evaluate the correctness of the proposed dPCRLB and to see how close the proposed dPCRLB can potentially be to the centralized PCRLB. In reality, the exact state values are not available. Scenario 2 considers the latter case. Fig. 1(a) shows one realization of the sensor placement along with the target’s track. Though all 20 nodes shown in Fig. 1(a) are used by the dPCRLB algorithm, Eqs. (41)-(43), Fig. 1(b) depicts the trace of the local PCRLBs computed using Eqs. (46)-(49) at four randomly selected nodes highlighted as Nodes 1-4 in Fig. 1(a). The local performance of nodes varies due to the state dependent nature of the problem. The dPCRLB is then computed from all 20 local PCRLBs based on Theorem 1. Fig. 2 compares the proposed full-order dPCRLB, the centralized PCRLB based on a fusion centre (included here as the ground truth), and the suboptimal dPCRLB based on [16] over 200 Monte Carlo runs with the same sensor network configuration. For each dPCRLB, its trace is plotted. Due to the state-dependent observation noise variance, we note that the SNR is time-varying and differs from one sensor node to the other depending on the location of the target. Two different SNR cases (averaged across all nodes and time) are considered: (i) High SNR, where the SNRs at different nodes varies form 17 dB to 24 dB with a mean value of 20 dB, (ii) Low SNR, where the SNRs ranges from 0 dB to 11 dB across the network with a mean value of 6 dB. Fig. 2(a) is for the high SNR case, while Fig. 2(b) plots the bounds for the low SNR scenario. Since the proposed bound is exact without any approximation, it predicts the estimator’s performance more accurately than the approximated approach [16] as illustrated in Figs. 2(a) and 2(b) where the centralized and distributed PCRLBs virtually overlap. Finally, we note that for low SNR scenarios, the performance of the approximated full-order dPCRLB (Eq. (40)) degrades significantly from the true bound due to the localized nature of the previous FIM (third term on the RHS of Eq. (40)). As illustrated in the first bullet after Lemma 2, the approximated expression uses $J_{FO}(l) \bar{x}(k)$ instead of the global FIM $J_{FO}(l) \bar{x}(k)$ which results in inaccuracies as well as variances in the dPCRLB from one node to another.

Scenario 2: uses the BOT model specified in Scenario 1 with the following differences: (i) the target track is not fixed (i.e., unlike Scenario 1 with fixed track, the track varies from one iteration to another in a Monte Carlo simulation); (ii) the dPCRLB is based on the estimated state values obtained from the particle filter [52] (as opposed to the true state values utilized in Scenario 1) in both centralized and distributed computation of the PCRLBs; (iii) in each Monte Carlo run (Monte Carlo simulation of 200 runs is performed), a different sensor network configuration is considered, where $N = 20$ nodes are scattered in a square region of dimension $(16 \times 16)$ m$^2$ at random. Because of these differences, the baseline (centralized PCRLB) and the comparison results are different between Figs. 2 and 3.

The full-order dPCRLB algorithm explained in Section III-D is used to compute the dPCRLB with Step 3 (incorporating Theorem 1) based on Eqs. (60)-(63), which includes expectations. We use consensus/fusion based distributed implementation of the particle filter (CF/DPF) [25] to compute the expectation terms over possible realizations of the state and observation sequences. Note that it is not the purpose of the paper to develop the distributed tracking algorithm and evaluate its performance. Rather our focus is on the proposed dPCRLB algorithm and the state vector can be obtained using any distributed approach. For the BOT problem, the computation of the Jacobian terms $\nabla_{\bar{x}(k)} f^T(k)$ and $\nabla_{\bar{x}(k+1)} g^{T(l)}(k+1)$ used in Eqs. (60)-(63) and the initialization step are further described in [53]. Matrix $[D_{FO}^{(l)}(k)]^{(l)}$ in Eq. (63) is derived based on the particle based approximation given in [49]. We note that both the centralized PCRLB and dPCRLB use state estimates from the particle filters. The centralized PCRLB uses state estimates computed by the centralized particle filter, while the distributed PCRLB uses estimates from the distributed particle filter such as the CF/DPF [25]. Consequently, any drop in the accuracy of the state estimates (due to for example a reduction in the SNR) affects both bounds. As long as the distributed particle filter is an optimal implementation of the centralized particle filter, the centralized PCRLB and dPCRLB should result in similar bounds. Comparing Figs. 3(a) and (b), we do notice an increase in the value of the overall bound itself at low SNRs that is reflected in a similar fashion in the plots for both centralized and dPCRLB.

Fig. 3(a) is for the high SNR case, while Fig. 3(b) plots the bounds for the low SNR scenario. In both cases, the centralized PCRLB and proposed dPCRLB are close (almost overlapping), while the approximated dPCRLB [16] fluctuates from the true value. In Figs. 3(a) and 3(b) corresponding to Scenario 2, the PCRLBs are higher than Figs. 2(a) and 2(b) corresponding...
Fig. 3. Scenario 2 in Full-order System: Same as Fig. 2 except particles set \{X_i(k), W_i(k)\} is used to compute expectations in Eqs. (46)-(49).

Fig. 4. Scenario 1 in Full-order System without Consensus: (a) Same as Fig. 1(b) except the local PCRLBs are fused within their neighborhoods at each filtering iteration. No consensus run is employed. (b) Magnified version of (a) in the range, 10 ≤ k ≤ 30.

to Scenario 1 because estimated values for states are used instead of the actual values and the target track varies between different runs of the Monte Carlo simulation.

Having justified that the proposed dPCRLB is an accurate representation of its centralized counterpart, Fig. 4 considers an alternative option in which all nodes use Theorem 1 to fuse the local PCRLBs within their immediate neighborhoods without performing any consensus, i.e., to perform local fusion without any consensus. In cases where consensus overhead is not tolerable, this technique can be employed. Such a scenario does not require knowing the number N of active nodes. Due to the localized nature of the dPCRLB fusion in this scenario, some variations in the dPCRLBs are observed among different neighborhoods. Fig. 4 further suggests that some nodes are self-confined where global fusion is not needed while some nodes do not achieve reasonable performance on the basis of local information and extra communication may be needed for some filter iterations if a higher accuracy is desired. Fig. 4 shows that there is potential in computing the dPCRLB without a consensus step. Further investigation in reducing or replacing the consensus step with an alternative approach will be considered as future work.

B. Reduced-order Systems

A large scale system comprising of 100 states (i.e., |x(k)| = 100) and 10 observation nodes (i.e., |z(k)| = 10) leading to 10 subsystems is simulated. Simulations are based on a linear model \(x(k+1) = Fx(k) + \xi(k+1)\), where the state matrix \(F\) is square with dimensions of \((100 \times 100)\). The state dependencies in \(F\) are shown in Fig. 5(a), where symbol ‘•’ at coordinate \((X_i, X_j)\) with \(X_i, X_j \in x\) indicates that \(X_i\) depends on \(X_j\) in the state model. The observation model is also linear and is given by \(z(k) = Gx(k) + \zeta(k)\). Fig. 5(b) illustrates the state dependencies in the observation model \(G\) with the \(y\)-axis representing observations made in Subsystems 1 to 10 and the \(x\)-axis representing the 100 states. For further details on the structure of \(F\) and \(G\), please refer to [35].

Our comparison for reduced-order systems is based on the results from three implementation schemes: (i) The centralized PCRLB based on Corollary 1; (ii) the dPCRLB based on Theorem 2 following the steps listed in Section III, (as discussed in Section IV-C, this approach is not in a reduced-order fashion, however, it is included as a benchmark to evaluate the correctness of Theorem 2); (iii) distributed reduced-order computation of the RO/dPCRLB as outlined in Section IV-C. As explained in Section IV-C, in Scheme (iii), each subsystem only compute and maintain a diagonal block of the global FIM which corresponds to its local state variables \(x^{(i)}(k)\). This results in a block diagonal banded FIM matrix while in Scheme (ii), each subsystem computes the global FIM using the RO/dPCRLB expressions which results in a full matrix. Scheme (ii), therefore, results in the exact dPCRLB while scheme (iii) produces approximate results as it only uses the diagonal blocks of the FIM to compute the PCRLB. Due to sparse and localized nature of the state model, we observe that removing the off-diagonal blocks of the global FIM does not result in significant loss of accuracy in the computed RO/PCRLB as shown in our results. Another case is to consider decoupled subsystems, i.e., to remove the forcing terms \(d^{(i)}(k-1)\) from the local state model at subsystem \(l\) (Eq. (65)). The dPCRLB for the decoupled systems increases monotonically because of the dropping of the forcing terms.
For the reduced-order systems, the FIM blocks $J$ from the local state models. This results in large errors beyond the scale used in Fig. 6 and is not reported here.

Fig. 6 plots the trace of the reduced-order dPCRLB $J^{-1}_R (x(k))$ obtained from a Monte Carlo simulations of 200 runs where the state model is randomly selected at each run. For the reduced-order systems, the FIM blocks $J^G_R (x^{(l)}(k))$ are appropriately combined to compute the overall FIM. In Fig. 6(a), the observation noise variance varies with time and across the sensor network. Averaged over the sensor network, the maximum and minimum values of the SNR are 26.59dB and 0.54dB across iterations $k$. The mean SNR is 16.40dB. The state covariance matrix $Q^{(l)}$ for $x^{(l)}(k)$ is diagonal and assumed known. Fig. 6(b) considers a low SNR scenario. Averaged over the sensor network, the maximum and minimum values of the SNR are 12.36dB and −4.32dB across iterations $k$. The mean SNR is 3.86dB. Figs. 6 show that the RO/dPCRLB (Scheme (ii)), its reduced-order computation (Scheme (iii)) and the centralized PCRLB (Scheme (i)) follow the same trend. The RO/dPCRLB is exact and overlaps with its centralized counterpart and its reduced-order implementation (Scheme (iii)) provides reasonable results. Slight degradation in the performance of Scheme (iii) is due to maintaining only block diagonals of the global FIM at each subsystem as explained in Section IV.D. Further improvements can be achieved by using off-diagonal FIM blocks corresponding to the interactions between states observed at the neighboring subsystems.

VI. CONCLUSION

The paper derives the dPCRLB for distributed full-order and reduced order estimation architectures without the need of a central processing unit as in distributed sensor network topologies. The centralized PCRLB can not be computed for these networks. The paper proposes the distributed PCRLB (dPCRLB) algorithms for full-order (FO/dPCRLB) expressed in terms of Theorem 1 and reduced-order (RO/dPCRLB) distributed systems in Theorem 2 and verify their accuracy through Monte Carlo simulations. Theorem 1 is applicable when the estimates of the entire state vector is available locally at each node. In reduced-order estimation, a different subset of the state vector is estimated at the local nodes. Theorem 1 cannot be implemented in the reduced-order systems and is replaced by Theorem 2 which allows for reduced-order FIMs corresponding to different subsets of the state vector to be fused to determine the overall FIM. Motivated by resource management decisions in sensor networks, optimal and near-optimal expressions for recursively computing the FO/dPCRLB are derived. We further extended our dPCRLB framework to reduced-order distributed estimation where the underlying state models are localized, while the sensor observations are sparse limited to a few state variables at each node. The proposed bounds offer significant improvement over the existing approaches (such as expression [16] for full-order systems).

As future work, we will consider issues related to computational complexity, data origin-uncertainty, and those arising from bandwidth limitation of the channel including quantization of local observations prior to communication.

APPENDIX A

Proof of Corollary 2: The proof of Corollary 2 uses the Markovian property of the state variables and is based on the following factorization of the joint prediction distribution

$$P(x(0:k+1)|z(1:k)) = P(x(0:k)|z(1:k))P(x(k+1)|x(k)).$$

APPENDIX B

Proof of Theorem 1: The proof for Theorem 1 is based on the following nonlinear Bayesian fusion rule [48] (Lemma 2), which expresses the global posterior density as a function of local filtering and prediction densities.

Lemma 2. Assuming that the observations conditioned on the state variables made at node $i$ are independent of the observations made at a different node $j$, $(j \neq i)$, the global posterior for a $N$-sensor network is

$$P(x(0:k)|z(1:k)) = \frac{\prod_{i=1}^{N} P(x(k+1)|z^{(i)}(1:k+1))}{\prod_{i=1}^{N} P(x(k+1)|z^{(i)}(1:k))} P(x(0:k)|z(1:k)).$$

We first consider $J_{FO}(x(0:k))$. Decomposing $x(0:k) = [x^T(0:k-1), x^T(k)]^T$ in $J_{FO}(x(0:k))$, Eq. (16) from Definition 1 reduces to

$$J_{FO}(x(0:k)) = \mathbb{E} \left\{ \begin{bmatrix} \Delta \pi_{x(k)}(0:k-1) | \Delta \pi_{x(k)}(0:k-1) \\ \Delta \pi_{x(k)}(0:k) | \Delta \pi_{x(k)}(0:k) \\ \Delta \pi_{x(k)}(0:k) | \Delta \pi_{x(k)}(0:k) \end{bmatrix} \log P(x(0:k)|z(1:k)) \right\}$$

$$\triangleq \begin{bmatrix} A_{F0}^{11}(k) & A_{F0}^{12}(k) \\ A_{F0}^{21}(k) & A_{F0}^{22}(k) \end{bmatrix},$$

provided that the aforementioned expectations and derivatives exist. The bottom right block (denoted by $A_{F0}^{22}(k)$) on the right hand side (RHS) of Eq. (95) corresponds to a $(2 \times 2)$ block matrix, i.e., $A_{F0}^{22}(k) = \mathbb{E} \left\{ -\Delta \pi_{x(k)}(0:k) \log P(x(0:k)|z(1:k)) \right\}$, and similarly for the remaining $A_{F0}^{ij}$'s.
Following the aforementioned procedure used to derive Eq. (95) for $J_{FO}(x(0:k+1))$, we get

$$J_{FO}(x(0:k+1)) =$$

$$\exp\left\{-\begin{bmatrix}
\Delta_{x(0:k-1)}(k) & \Delta_x(k) & \Delta_{x(k)}(k) \\
\Delta_{x(0:k-1)}(k) & \Delta_x(k) & \Delta_{x(k)}(k) \\
\Delta_{x(0:k-1)}(k) & \Delta_x(k) & \Delta_{x(k)}(k)
\end{bmatrix}
\right\} \log P(x(0:k+1)|z(1:k+1))$$

$$= \Delta_{x(0:k-1)}(k) \log \frac{P(z(k+1)|x(k+1))P(x(k+1)|x(k))}{P(z(1:k)|x(1:k))} + \Delta_{x(0:k-1)}(k) \log P(x(0:k)|z(1:k)),$$

which leads to

$$E_{FO}^{11}(k) \triangleq \mathbb{E}[\Delta_{x(0:k-1)}(k) \log \frac{P(z(k+1)|x(k+1))P(x(k+1)|x(k))}{P(z(1:k)|x(1:k))} + \Delta_{x(0:k-1)}(k) \log P(x(0:k)|z(1:k))],$$

where $0$ stands for a block of all zeros with the appropriate dimension. To save on space, only prove the remaining entries can be proven following a similar procedure.

**Case 1** (Proof for $E_{FO}^{11}(k) = A_{FO}^{11}(k)$): Factoring the posterior distribution (Eq. (5)) for the top left block in Eq. (96),

$$\Delta_{x(0:k-1)}(k) \log P(x(0:k+1)|z(1:k+1))$$

$$= \Delta_{x(0:k-1)}(k) \log \frac{P(z(k+1)|x(k+1))P(x(k+1)|x(k))}{P(z(1:k)|x(1:k))} + \Delta_{x(0:k-1)}(k) \log P(x(0:k)|z(1:k)),$$

which leads to

$$E_{FO}^{11}(k) \triangleq \mathbb{E}[\Delta_{x(0:k-1)}(k) \log \frac{P(z(k+1)|x(k+1))P(x(k+1)|x(k))}{P(z(1:k)|x(1:k))} + \Delta_{x(0:k-1)}(k) \log P(x(0:k)|z(1:k))],$$

where $0$ stands for a block of all zeros with the appropriate dimension. To save on space, only prove the remaining entries can be proven following a similar procedure.

**Appendix C:**

**Proof of Corollary 3:** The proofs for Eqs. (38) and (39) (details not included here due to lack of space) are similar to that for Theorem 1 with the posterior factorization of $P(x(0:k+1)|z(1:k+1))$ defined in Lemma 3, [48], below.

**Lemma 3.** Assuming that the observations conditioned on the state variables made at node $l$ are independent of the observations made at a different node $j$, ($j \neq l$), the global posterior for a $N$-sensor network is

$$P(x(0:k+1)|z(1:k+1)) \propto \prod_{l=1}^{N} P(x(k+1)|z(l)(k+1), z(1:k)) \prod_{l=1}^{N} P(x(k+1)|z(l)(1:k)) \times P(x(0:k)|z(1:k)).$$

The change in $P(x(0:k+1)|z(1:k+1))$ is due to the setup used in Scenario 2, where both current local observation and previous global observations are used in the current state estimate.

**Appendix D:**

**Proof of Theorem 2:** To prove Theorem 2, we use a different factorization of the posterior, which expresses the global posterior distribution $P(x(0:k)|z(1:k))$ as a function of local reduced-order filtering distributions $P(x(0:k)|z(l)(1:k))$. Lemma 4 [50] describes the nonlinear fusion rule.
Lemma 4. Assuming that the observations conditioned on the state variables made at node \( l \) are independent of the observations made at node \( j \), \( j \neq l \), the global posterior for a reduced-order estimation model is given by

\[
P(x(0:k)|z(1:k)) \propto \prod_{i=1}^{N} P(x^{(i)}(k)|z^{(i)}(1:k))
\]

Due to limited space, we only highlight the main steps of the proof. The FIM \( J_{\text{RO}}(x(0:k+1)) \) and the associated notation \( E^{**}(k) \) for the reduced-order is similar in structure to Eq. (96) except the subscript 'FO' is replaced by 'RO'. Using factorization (107) in the first term on RHS of Eq. (96) for \( J_{\text{RO}}(x(0:k+1)) \) and simplifying

\[
J_{\text{RO}}(x(0:k+1)) = \begin{bmatrix}
A_{11}^{\text{RO}}(k) & A_{12}^{\text{RO}}(k) & 0 \\
A_{21}^{\text{RO}}(k) & A_{22}^{\text{RO}}(k) + C_{11}^{\text{RO}}(k) & C_{12}^{\text{RO}}(k) \\
0 & D_{22}^{\text{RO}}(k) & C_{22}^{\text{RO}}(k)
\end{bmatrix}
\]

where terms \( A_{11}^{\text{RO}}(k), A_{12}^{\text{RO}}(k), A_{21}^{\text{RO}}(k), A_{22}^{\text{RO}}(k), \) and \( C_{22}^{\text{RO}}(k) \) are the same as their full-order counterparts (i.e., \( A_{11}^{**}(k) = A_{22}^{**}(k) \)) as defined in Eq. (95) and \( C_{11}^{\text{RO}}(k), C_{12}^{\text{RO}}(k), C_{21}^{\text{RO}}(k), C_{22}^{\text{RO}}(k) \) are expressed in Eqs. (74)-(76). Note that the derivation of Eq. (108) is similar to the derivation of (97) included in the proof of Theorem 5. The information sub-matrix \( J_{\text{RO}}(x(0:k+1)) \) is calculated as the inverse of the right lower \((n_x \times n_x)\) sub-matrix of \( [J_{\text{RO}}(x(0:k+1))]^{-1} \) in Eq. (108) which is given by Eq. (73).

**APPENDIX E**

In the context of distributed estimation, consensus is the process of establishing a consistent value for some statistics of the state vector across the network by interchanging relevant information between the connected neighboring nodes. Consensus algorithms are generally iterative in nature, where each node begins with a set of local information, at each iteration, data is exchanged between a subset of nodes, which assimilates new information to update the local parameters. The average consensus algorithm [23] is considered in the manuscript. Consider a connected network with \( N \) nodes where each node \( l \), \( 1 \leq l \leq N \), possesses a scalar quantity \( X_c(0) \) and the goal is to distributively compute the average \( 1/N \sum_{i=1}^{N} X_c(0) \) such that the result is available at each node. At iteration \( t \), each node updates its consensus state \( X_c(t) \) as follows

\[
X_c^{(l)}(t+1) = U_{ll}(t)X_c^{(l)}(t) + \sum_{j \in N_L^{(l)}} U_{lj}(t)X_c^{(j)}(t), \quad (109)
\]

which can alternatively be expressed as \( x_c(t+1) = U(t)x_c(t) \) in the matrix-vector format, where \( U(t) \Delta \{U_{lj}\} \in \mathbb{R}^{N \times N} \) is referred to as the consensus matrix representing the network communication constraints, \( X_c^{(l)}(t) \) is the consensus state variable(s) at node \( l \), for \( 1 \leq l \leq N \), \( t \) is the consensus time index that is different from the filtering time index \( k \), and \( N_L^{(l)} \) represents the set of neighboring nodes for node \( l \). A recent review on the average consensus algorithms can be found in [23].

**APPENDIX F**

**Distributed Iterate-Collapse Inversion Algorithm (DICI-RO):** The DICI-RO is an iterative distributed algorithm used for computing the inverse of the symmetric positive definite banded matrix \( S(k) \) defined in Eq. (78), when its submatrices in the banded area are distributed among different local nodes. The DICI-RO is a 2-step algorithm with an *iterate step* and a *collapse step*. The iterate step is implemented to compute the corresponding (banded) elements of the inverse of \( S(k) \). A nonlinear collapse step is then employed to compute the non-banded elements of the inverse of \( S(k) \) from already computed banded elements of the inverse of \( S(k) \). Please refer to [35] for further details. In our problem, we need to compute the inverse of \( S(k) \) from diagonal blocks distributed across the network at local subsystems. Matrix \( S(k) = J_{\text{RO}}^G(x(k)) + C_{11}^{\text{RO}}(k) \) is assumed block-banded as only diagonal blocks corresponding to the local subsystems are computed in our algorithm. Matrix \( C_{11}^{\text{RO}}(k) \) is also banded because of the localized and sparse nature of the state model.

**REFERENCES**


Amir Asif (M’97 - SM’02) received the M.S. and Ph.D. degrees in electrical and computer engineering from Carnegie Mellon University (CMU), Pittsburgh, PA, in 1993 and 1996, respectively. Previously, he was on the faculty of CMU, where he was a Research Engineer from 1997 to 1999 and the Technical University of British Columbia, Vancouver, BC, Canada, where he was an Associate Professor from 1999 to 2002. Since 2002, he has been an Associate Professor / Professor of electrical engineering and computer science in Lassonde School of Engineering at York University, Toronto, ON, Canada. He works in the area of statistical signal processing and communications. His current projects include time reversal; distributed signal processing for sensor networks; genomic signal processing; and sparse, block-banded matrix technologies. He has authored about 100 technical contributions, including invited ones, published in international journals and conference proceedings, and the textbook Continuous and Discrete Time Signals and Systems published by the Cambridge University Press. Dr. Asif has been a Technical Associate Editor for the IEEE SIGNAL PROCESSING LETTERS from 2002 to 2006 and from 2009 to 2012. Currently, he is serving as the Associate Editor of IEEE TRANSACTIONS ON SIGNAL PROCESSING. He has organized three IEEE conferences on signal processing theory and applications and served on the technical committees of several others. He has received several distinguishing awards, including the York University Faculty of Graduate Studies Teaching Award in 2009 and York University Wide Teaching Award (Full-Time Senior Faculty Category) in 2008 from York University. He is a member of the Professional Engineering Society of Ontario.