Approximating Posterior Cramér–Rao Bounds for Nonlinear Filtering Problems Using Gaussian Mixture Models

SHUO ZHANG
DEFENG CHEN
TUO FU
HUAWEI CAO
School of Information and Electronics, Beijing Institute of Technology Beijing 100081, China

The posterior Cramér–Rao bound (PCRB) is a fundamental tool to assess the accuracy limit of the Bayesian estimation problem. In this article, we propose a novel framework to compute the PCRB for the general nonlinear filtering problem with additive white Gaussian noise. It uses the Gaussian mixture model to represent and propagate the uncertainty contained in the state vector and uses the Gauss–Hermite quadrature rule to compute mathematical expectations of vector-valued nonlinear functions of the state variable. The detailed pseudocodes for both the small and large component covariance cases are also presented. Three numerical experiments are conducted. All of the results show that the proposed method has high accuracy and it is more efficient than the plain Monte Carlo integration approach in the small component covariance case.

Manuscript received April 9, 2020; revised August 16, 2020 and October 15, 2020; released for publication October 17, 2020. Date of publication November 3, 2020; date of current version April 10, 2021.

DOI. No. 10.1109/TAES.2020.3035426

Refereeing of this contribution was handled by K. T. Wong.

This work was supported by the National Natural Science Foundation of China under Grant 61421001.

Authors' address: S. Zhang, D. Chen, T. Fu, and H. Cao are with the School of Information and Electronics, Beijing Institute of Technology, Beijing 100081, China, E-mail: (zsmathbit@gmail.com; defengchenbit@gmail.com; futuobit@gmail.com; hwei.cao@gmail.com). *(Corresponding author: Defeng Chen.)*

I. INTRODUCTION

Nonlinear state estimation or filtering is ubiquitous in engineering fields, such as target tracking, space vehicle navigation, and robotics. For a nonlinear filtering problem, its complete solution requires precise knowledge of the state posterior probability density function (pdf) conditioned on all of the available measurements, which generally cannot be described using a finite number of parameters [1], [2]. A special case is the linear model with additive white Gaussian noise (AWGN), in which the state posterior pdf is Gaussian. The optimal solution for this linear case is known as the Kalman filter. Although finding the optimal nonlinear state estimator is usually intractable under limited computational resources, many effective suboptimal filtering algorithms have been proposed, including the extended Kalman filter [3], unscented Kalman filter [4], [5], and particle filter (PF) [6]. All of these algorithms are based on either an approximation of the filter model or a preassumed special form of the posterior pdf. A practical concern when using a suboptimal filter is its accuracy compared with that of the optimal solution. However, quantitatively assessing the highest achievable accuracy is a nontrivial problem.

The Cramér–Rao bound (CRB) in the parametric estimation framework finds the lower bound on the minimum covariance when estimating the fixed-value parameter using noise-corrupted measurements. It has become a standard tool to assess the performance of a concrete estimator. However, the sequential state estimation is situated in the Bayesian setting, where the variable being estimated is random and described by a known prior pdf. Therefore, the CRB cannot be directly applied. The CRB counterpart in the Bayesian framework is the posterior CRB (PCRB), which was proposed by Van Trees in [7]. It finds the lower bound on the minimum mean square error (mse) when estimating the random parameter. The PCRB can be used for the following.

1) Assessing the effect of approximation for a suboptimal filter [8]–[10].
2) Verifying whether the imposed design requirement is realistic.
3) Selecting the best tracking device in a multisensor network [11]–[13].

It should be noted that when using PCRB in a highly nonlinear non-Gaussian context, there is some danger, the bound is not as tight as desired, especially in low signal-to-noise scenarios [14].

Due to its great value in both theoretical and practical aspects, the PCRB has been extensively studied since its emergence. A breakthrough was achieved in the work of Tichavský *et al.* [15], in which the authors presented a recursive formula for calculating the PCRB. This work sets a foundation for applying the PCRB to the nonlinear filtering realm and boosts numerous research works, including some important modifications. Perceiving Tichavský *et al.* [15] implicitly assumes that the detection probability $P_d = 1$ and the false alarm probability $P_{\text{fa}} = 0$, Farina *et al.* extended the recursive formula to target tracking applications with

$P_d < 1$ and $P_{\text{fa}} = 0$ by first computing the PCRB conditioned on a specific detection/miss sequence and then averaging over all possible detection/miss sequences [16]. Hernandez *et al.* [16] subsequently extended the result to $P_{\text{fa}} \neq 0$ situations [17]. All of the aforementioned works are based on a single target scenario. Hue *et al.* further generalized the original recursive PCRB to accommodate multitarget tracking applications [18].

A practical difficulty related to the PCRB is how to efficiently compute it. The recursive formula presented in [15] and its modifications in [16]–[18] all require mathematical expectations of some nonlinear functions of the state vector. Such expectations involve nontrivial high-dimensional integrations of these functions multiplied by the state pdf. These integrations are conventionally approximated by Monte Carlo simulations [9], [10], which may consume too much computation time in real-world applications. Bréhard and Cadre derived the closed-form PCRB for the bearings-only tracking problem by introducing the logarithmic polar coordinate system [19]. A main deficiency of that research is that the authors assumed the target follows a nearly constant-velocity model in the Cartesian frame, which is not always adequate for practical applications, such as satellite tracking. Lei *et al.* developed an approach for approximating the PCRB using the online estimated state [20]. A main assumption in Lei's paper is that the state and measurement can be modeled as Gaussian distributed vectors. However, such an assumption may be inappropriate when the system is highly nonlinear or the state pdf becomes multimodal. More recently, Tulsyan *et al.* proposed a sequential Monte Carlo (SMC) based method to approximate the PCRB [21]. The authors first computed the expectations with respect to the measurement-conditioned state pdf by using the SMC approach and then with respect to the measurement pdf by the standard Monte Carlo method. This method is suitable for general nonlinear, non-Gaussian filtering problems. However, it still requires Monte Carlo simulation to generate multiple groups of measurements, which could be computationally inefficient. To our knowledge, a non-Monte Carlo based, general-purpose framework to accurately compute the PCRB remains unresolved.

In this article, a novel Gaussian mixture model (GMM) based approach to accurately compute the PCRB for general nonlinear filtering problems with the AWGN is proposed. This method uses the GMM to represent and propagate the uncertainty contained in the state vector and uses the Gauss–Hermite quadrature rule to compute the mathematical expectation of a vector-valued nonlinear function of the state variable. The main contributions are as follows.

1) Integrated frameworks with detailed pseudocodes for both the small and large component covariance cases are presented. Because the only model assumption is the AWGN, the proposed framework is suitable for any nonlinear system or measurement equation. Furthermore, this framework does not rely on the stochastic sampling technique, so its computational efficiency is superior to the plain Monte Carlo integration for computing the PCRB, especially in the small component covariance case.
2) When the system is highly nonlinear or the process noise is too large, covariances of the GMM components can be unduly enlarged in the uncertainty propagation process, resulting in degradation of accuracy. This is referred to as the covariance expansion. We propose a refine-coarsen step for propagating the state uncertainty, which successfully eliminates the covariance expansion phenomenon and overcomes the exponentially growing memory problem.
3) Numerical simulations, including a physically based satellite tracking problem, are performed. All of the results show that the accuracy of the proposed framework is high. In particular, from the satellite tracking simulation, we find that including the high-accuracy radial velocity measurement can effectively improve the orbit accuracy. Thus, a radar system working in coherent integration mode can not only improve its detection ability for small targets but also generate high-accuracy radial velocity measurements to better reduce the target uncertainty. This is meaningful for space situational awareness.

The rest of this article is organized as follows. Section II presents a concise review on the definition of the PCRB. Section III introduces the GMM, including the split of a general Gaussian pdf and the uncertainty mapping under a nonlinear transform. Section IV shows the detailed algorithms for computing PCRBs in both the small and large component covariance cases. Section V demonstrates the simulation results for three numerical examples. Finally, Section VI concludes this article.

II. REVIEW OF PCRBS

Let $\mathbf{x}_n$ be an L-dimensional state vector at time n that evolves according to

$$\mathbf{x}_{n+1} = \mathbf{f}_n(\mathbf{x}_n) + \mathbf{w}_n \tag{1}$$

where $\mathbf{f}_n(\cdot)$ is a general nonlinear function, and $\mathbf{w}_n$ is the zero-mean white Gaussian process noise with a nonsingular covariance matrix of $\mathbf{Q}_n$. The measurement $\mathbf{z}_{n+1}$ is linked to the state vector $\mathbf{x}_{n+1}$ by

$$\mathbf{z}_{n+1} = \mathbf{h}_{n+1}\left(\mathbf{x}_{n+1}\right) + \mathbf{v}_{n+1} \tag{2}$$

where $\mathbf{h}_{n+1}(\cdot)$ is also a general nonlinear function, and $\mathbf{v}_{n+1}$ is the zero-mean white Gaussian measurement noise with a nonsingular covariance matrix of $\mathbf{R}_{n+1}$. The initial state $\mathbf{x}_0$ is available as the *a priori* information, and $\mathbf{x}_0$, $\mathbf{w}_n$, and $\mathbf{v}_n$ are mutually independent.

The PCRB states that the mse matrix $\mathbf{M}_n$ for any unbiased estimation of $\mathbf{x}_n$ must satisfy

$$\mathbf{M}_n - \mathbf{J}_n^{-1} \geq \mathbf{0} \tag{3}$$

where $\mathbf{J}_n$ is the Fisher information matrix (FIM). The symbol "$\geq$" means that $\mathbf{M}_n - \mathbf{J}_n^{-1}$ is always positive semidefinite. Tichavský *et al.* [15] proved that the FIM $\mathbf{J}_n$ can be

expressed recursively as

$$\mathbf{J}_{n+1} = \mathbf{D}_n^{22} - \mathbf{D}_n^{21}\left(\mathbf{J}_n + \mathbf{D}_n^{11}\right)^{-1}\mathbf{D}_n^{12} \tag{4}$$

where

$$\mathbf{D}_n^{11} = \mathbf{E}\left\{\frac{\partial \mathbf{f}_n^T(\mathbf{x}_n)}{\partial \mathbf{x}_n}\mathbf{Q}_n^{-1}\frac{\partial \mathbf{f}_n(\mathbf{x}_n)}{\partial \mathbf{x}_n^T}\right\} \tag{5}$$

$$\mathbf{D}_n^{12} = -\mathbf{E}\left\{\frac{\partial \mathbf{f}_n^T(\mathbf{x}_n)}{\partial \mathbf{x}_n}\right\}\mathbf{Q}_n^{-1} \tag{6}$$

$$\mathbf{D}_n^{21} = -\mathbf{Q}_n^{-1}\mathbf{E}\left\{\frac{\partial \mathbf{f}_n(\mathbf{x}_n)}{\partial \mathbf{x}_n^T}\right\} = \left[\mathbf{D}_n^{12}\right]^T \tag{7}$$

$$\mathbf{D}_n^{22} = \mathbf{Q}_n^{-1} + \mathbf{E}\left\{\frac{\partial \mathbf{h}_{n+1}^T\left(\mathbf{x}_{n+1}\right)}{\partial \mathbf{x}_{n+1}}\mathbf{R}_{n+1}^{-1}\frac{\partial \mathbf{h}_{n+1}\left(\mathbf{x}_{n+1}\right)}{\partial \mathbf{x}_{n+1}^T}\right\}. \tag{8}$$

The initial FIM $\mathbf{J}_0$ will be $\mathbf{\Sigma}_0^{-1}$ if the initial state $\mathbf{x}_0$ is distributed as $N(\boldsymbol{\mu}_0, \mathbf{\Sigma}_0)$.

It is noted that the PCRB derived in [15] is for more general scenarios, and this article only presents a special case with AWGN. A prominent feature of the PCRB is that it is determined only by the initial state distribution and the system and measurement models. Therefore, the PCRB can be computed completely offline, before any real measurement datum is received. However, there are three main difficulties that prevent the efficient computation of PCRBs for general nonlinear filtering problems, which are as follows.

1) The state transition function $\mathbf{f}_n(\cdot)$ may not have an explicit expression, especially in situations where the evolution of the state is described by a group of ordinary differential equations.
2) The state pdf $p(\mathbf{x}_n)$ evolves according to the Chapman–Kolmogorov equation, which generally does not have an analytic solution.
3) Even if $\mathbf{f}_n(\cdot)$ and $p(\mathbf{x}_n)$ can be expressed analytically, the high-dimensional integrations involved in (5)–(8) when acquiring the expectations also have no simple expressions in general.

For the aforementioned reasons, the PCRB is usually obtained via Monte Carlo integrations, which could be computationally heavy. The lack of an efficient yet high-accuracy algorithm for evaluating the PCRB still remains unresolved.

A careful look at (5)–(8) reveals that the only random vector (RV) in evaluating $\mathbf{D}_n^{11}$, $\mathbf{D}_n^{12}$, and $\mathbf{D}_n^{21}$ is $\mathbf{x}_n$, whereas that in evaluating $\mathbf{D}_n^{22}$ is $\mathbf{x}_{n+1}$. Therefore, if we can find a more effective method to represent $p(\mathbf{x}_n)$ and $p(\mathbf{x}_{n+1})$ recursively, we can develop a more computationally efficient algorithm to obtain the PCRB.

III. INTRODUCTION TO GMMS

A. Gaussian Mixture Refinement

The GMM is a powerful tool to approximate any true pdf of an RV by a group of weighted Gaussian pdfs. Suppose that $\mathbf{x}$ is a general RV with a pdf of $p(\mathbf{x})$. $p(\mathbf{x})$ can then be approximately represented as

$$p(\mathbf{x}) \approx \sum_{i=1}^{\alpha} w_i N\left(\mathbf{x};\, \boldsymbol{\mu}_i,\, \mathbf{\Sigma}_i\right) \tag{9}$$

where α is the component number, and w_i, $\boldsymbol{\mu}_i$, and $\mathbf{\Sigma}_i$ are the weight, mean, and covariance of the ith Gaussian component, respectively. In the rest of this article, the notation $N(\cdot)$ with three parameters represents the pdf, whereas $N(\cdot)$ with two parameters refers to the distribution itself. To retain a valid pdf, w_i must satisfy the following constraints:

$$0 \le w_i \le 1, i = 1, 2, \ldots, \alpha \tag{10}$$

$$\sum_{i=1}^{\alpha} w_i = 1. \tag{11}$$

It has been proved that the GMM can converge to the true pdf uniformly when the component covariance tends to zero and the component number tends to infinity [22]. Due to its high accuracy and mathematical simplicity, the GMM has been widely used in the fields of uncertainty propagation, nonlinear filtering, etc.

In the context of nonlinear filtering, once the state pdf is approximated by a GMM, there may be a requirement for increasing the number of components of the GMM while decreasing the component covariances, which is referred to as the GMM refinement. This can be done by refining each Gaussian component separately. The foundation of refining a general Gaussian pdf $N(\mathbf{x};\, \boldsymbol{\mu},\, \mathbf{\Sigma})$ into a GMM is the refinement of the standard univariate normal pdf $N(x;\, 0,\, 1)$

$$N(x;\, 0,\, 1) \approx \sum_{i=1}^{M} w_i N\left(x;\, \mu_i,\, \sigma_i^2\right). \tag{12}$$

Such a GMM is also referred to as the 1-D splitting library. It can be computed and stored in advance. The parameters to be determined are the component number M and the triplet (w_i, μ_i, σ_i) for each component. The solution for these parameters usually comes down to a complex nonlinear optimization process with the following objective function:

$$E = \int_{-\infty}^{+\infty}\left[N(x;\, 0,\, 1) - \sum_{i=1}^{M} w_i N\left(x;\, \mu_i,\, \sigma_i^2\right)\right]^2 \mathrm{d}x \tag{13}$$

where E is the L_2 distance between $N(x;\, 0,\, 1)$ and the splitting library.

To reduce the complexity, Horwood *et al.* proposed two suboptimal solutions to the aforementioned problem [23], [24]. One is based on the Gauss–Hermite quadrature rule, in which w_i is set to the ith quadrature weight and $\mu_i = k\varepsilon_i$, where ε_i is the ith quadrature node and k is a dispersion parameter; $\sigma_i^2 = \sigma^2 = 1 - k^2$. One salient feature of such a splitting library is that its higher order moments are matched with $N(x;\ 0,\, 1)$. However, the library places too many components at the tail of $N(x;\, 0,\, 1)$, especially when the component number M is large. The second suboptimal solution is to set M, μ_i, and σ_i according to

$$\sigma_i = \sigma < 1 \tag{14}$$

$$\mu_i = -m + \sigma\,(i-1) \tag{15}$$

$$M = \lceil 1 + 2m/\sigma \rceil \tag{16}$$

where $m = 6$ if $\sigma < 1/2$, else $m = 4$; $\lceil\cdot\rceil$ is the ceiling function. Only the weights then must be optimized. Under these extra constraints, the optimization of (13) reduces to a quadratic optimization problem, which is much simpler to solve.

For the multivariate standard Gaussian pdf $N(\mathbf{x}; \mathbf{0}, \mathbf{I})$, the splitting library must be applied along a specific dimension. Assuming the dth dimension is to be split, we have

$$N(\mathbf{x}; \mathbf{0}, \mathbf{I}) \approx \sum_{i=1}^{M} w_i N\left(\mathbf{x}; \boldsymbol{\mu}_i^d, \boldsymbol{\Sigma}_i^d\right) \tag{17}$$

where

$$\boldsymbol{\mu}_i^d = \left[0, \ldots, 0_{d-1}, \mu_i, 0_{d+1}, \ldots, 0\right]^T \tag{18}$$

$$\boldsymbol{\Sigma}_i^d = \operatorname{diag}\left\{1, \ldots, 1_{d-1}, \sigma^2, 1_{d+1}, \ldots, 1\right\}. \tag{19}$$

In (19), diag$\{\cdot\}$ denotes a diagonal matrix.

For a general multivariate Gaussian pdf $N(\mathbf{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$, the affine transform $\mathbf{x} = \mathbf{L}\mathbf{y} + \boldsymbol{\mu}$ maps $N(\mathbf{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$ into $N(\mathbf{x}; \mathbf{0}, \mathbf{I})$, where $\boldsymbol{\Sigma} = \mathbf{L}\mathbf{L}^T$. Therefore, the unidirectional GMM for $N(\mathbf{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$ is

$$N(\mathbf{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) \approx \sum_{i=1}^{M} w_i N\left(\mathbf{x}; \boldsymbol{\mu} + \mathbf{L}\boldsymbol{\mu}_i^d, \mathbf{L}\boldsymbol{\Sigma}_i^d\mathbf{L}^T\right). \tag{20}$$

Note that in (20), each component is a Gaussian pdf, so it can be furthered refined by splitting. By repeatedly applying the splitting library, we finally obtain the multidirectional GMM for a general Gaussian pdf.

B. Uncertainty Mapping

In the rest of the article, let $p(\cdot)$ denotes a pdf. Assume $\mathbf{y} = \mathbf{g}(\mathbf{x})$ is a nonlinear function of $\mathbf{x}$. The problem of finding $p(\mathbf{y})$ from the known $p(\mathbf{x})$ is referred to as the uncertainty mapping, which is nontrivial to be rigorously solved. In the context of nonlinear filtering, finding $p(\mathbf{x}_{n+1})$ from $p(\mathbf{x}_n)$ recursively is also known as the uncertainty propagation problem.

If $p(\mathbf{x})$ has been represented as a GMM, then $p(\mathbf{y})$ can also be approximated by a GMM. There are two widely used approaches to perform the uncertainty mapping between the two GMMs: the linearization method and the unscented transform (UT). Suppose that $p(\mathbf{x})$ is approximated by

$$p(\mathbf{x}) \approx \sum_{i=1}^{\alpha} w_i^{\mathbf{x}} N\left(\mathbf{x}; \boldsymbol{\mu}_i^{\mathbf{x}}, \boldsymbol{\Sigma}_i^{\mathbf{x}}\right) \tag{21}$$

$p(\mathbf{y})$ then can be approximated as

$$p(\mathbf{y}) \approx \sum_{i=1}^{\beta} w_i^{\mathbf{y}} N\left(\mathbf{y}; \boldsymbol{\mu}_i^{\mathbf{y}}, \boldsymbol{\Sigma}_i^{\mathbf{y}}\right). \tag{22}$$

For the sake of simplicity, the component number and weights are assumed to stay unchanged before and after mapping, although there are studies concerning updating the weights to improve the accuracy [25]. Therefore

$$\beta = \alpha \tag{23}$$

$$w_i^{\mathbf{y}} = w_i^{\mathbf{x}}. \tag{24}$$

For the linearization method, the mapped mean and covariance are

$$\boldsymbol{\mu}_i^{\mathbf{y}} = \mathbf{g}\left(\boldsymbol{\mu}_i^{\mathbf{x}}\right) \tag{25}$$

$$\boldsymbol{\Sigma}_i^{\mathbf{y}} = \tilde{\mathbf{G}}(\boldsymbol{\mu}_i^{\mathbf{x}})\boldsymbol{\Sigma}_i^{\mathbf{x}}\tilde{\mathbf{G}}^T(\boldsymbol{\mu}_i^{\mathbf{x}}) \tag{26}$$

where $\tilde{\mathbf{G}} = \partial\mathbf{g}(\mathbf{x})/\partial\mathbf{x}^T$ supposing the first-order partial derivative of $\mathbf{g}(\mathbf{x})$ exists. Although the linearization method is efficient, it could become inaccurate when $\mathbf{g}(\mathbf{x})$ is highly nonlinear. Furthermore, the need for the first-order partial derivative can also become a restriction when $\mathbf{g}(\mathbf{x})$ is nondifferentiable at some points. To overcome the deficiencies of linearization, UT is proposed [4], [5]. It uses a set of deterministically chosen, weighted sigma points to approximate the mean and covariance for each component in (22).

Here, we use the scaled UT (SUT) to map the uncertainty, which is a variant of the original UT approach [26], [27]. For the ith component in (21), we first generate the sigma points and weights based on $N(\mathbf{x}; \boldsymbol{\mu}_i^{\mathbf{x}}, \boldsymbol{\Sigma}_i^{\mathbf{x}})$ as follows:

$$\boldsymbol{\xi}_0 = \boldsymbol{\mu}_i^{\mathbf{x}}, \quad W_0^m = \frac{\lambda}{L+\lambda}, \quad W_0^c = \frac{\lambda}{L+\lambda} + \left(1 - \varepsilon^2 + \beta\right) \tag{27}$$

$$\boldsymbol{\xi}_l = \boldsymbol{\mu}_i^{\mathbf{x}} + \left(\sqrt{(L+\lambda)\,\boldsymbol{\Sigma}_i^{\mathbf{x}}}\right)_l, \quad W_l^m = W_l^c = \frac{1}{2\,(L+\lambda)} \quad l = 1, 2, \ldots, L \tag{28}$$

$$\boldsymbol{\xi}_{l+L} = \boldsymbol{\mu}_i^{\mathbf{x}} - \left(\sqrt{(L+\lambda)\,\boldsymbol{\Sigma}_i^{\mathbf{x}}}\right)_l, \quad W_{l+L}^m = W_{l+L}^c = \frac{1}{2\,(L+\lambda)} \quad l = 1, 2, \ldots, L \tag{29}$$

where $\{\boldsymbol{\xi}_l\}_{l=0}^{2L}$ is the sigma point set, $(\cdot)_l$ denote the lth column of a matrix; $\{W_l^m\}_{l=0}^{2L}$ and $\{W_l^c\}_{l=0}^{2L}$ are the weights for computing the mean and covariance, respectively; L is the dimension of the state; $\lambda = \varepsilon^2(L+\kappa) - L$. There are three free parameters in this sigma-point set, namely, ε, β, and κ. The parameter ε controls the dispersion of the sigma-point set, its value should satisfy $0 \le \varepsilon \le 1$. For Gaussian distributions, $\beta = 2$ is optimal; to make sure that the weights are all positive numbers, κ should satisfy $\kappa \ge 0$ [28]. The mean and covariance of the ith component in (22) are then calculated as follows:

$$\boldsymbol{\eta}_l = \mathbf{g}\left(\boldsymbol{\xi}_l\right) \tag{30}$$

$$\boldsymbol{\mu}_i^{\mathbf{y}} = \sum_{l=0}^{2L} W_l^m \boldsymbol{\eta}_l \tag{31}$$

$$\boldsymbol{\Sigma}_i^{\mathbf{y}} = \sum_{l=0}^{2L} W_l^c \left(\boldsymbol{\eta}_l - \boldsymbol{\mu}_i^{\mathbf{y}}\right)\left(\boldsymbol{\eta}_l - \boldsymbol{\mu}_i^{\mathbf{y}}\right)^T. \tag{32}$$

The SUT approach has better accuracy than the linearization method and is suitable for situations where $\mathbf{g}(\mathbf{x})$ does not have explicit expressions or is nondifferentiable.

The basic idea behind the GMM uncertainty mapping is that if the covariance of a Gaussian vector is small, the nonlinear mapping relationship can be well approximated by a first-order Taylor expansion around the mean of this Gaussian vector. Therefore, each Gaussian component in (21) is mapped into another Gaussian component in (22) while the whole pdf $p(\mathbf{y})$ can deviate from the shape of $p(\mathbf{x})$ dramatically.

C. Expectation of a Nonlinear Function

Now, consider the mathematical expectation of $\mathbf{y} = \mathbf{g}(\mathbf{x})$; namely

$$\mathbf{E}(\mathbf{y}) = \int \mathbf{g}(\mathbf{x})\, p(\mathbf{x})\, \mathrm{d}\mathbf{x}. \tag{33}$$

Such a high-dimensional integration usually does not have an analytic result, and the unknown of the precise expression for $p(\mathbf{x})$ intensifies this difficulty. However, if we replace $p(\mathbf{x})$ with (9), then (33) reduces to

$$\mathbf{E}(\mathbf{y}) \approx \sum_{i=1}^{\alpha} w_i \int \mathbf{g}(\mathbf{x})\, N\left(\mathbf{x};\, \boldsymbol{\mu}_i,\, \boldsymbol{\Sigma}_i\right) \mathrm{d}\mathbf{x}. \tag{34}$$

The core of (34) is the integration of the product of a general function and a Gaussian pdf, which has an efficient numerical algorithm: the Gauss–Hermite quadrature rule [29].

Let us first consider the 1-D case. Based on the Gauss–Hermite quadrature rule, the following equation can be established:

$$\int_{-\infty}^{+\infty} g(x) e^{-x^2} \mathrm{d}x \approx \sum_{i=1}^{n_q} \varpi_i g(\varepsilon_i) \tag{35}$$

where $\{\varpi_i\}_{i=1}^{n_q}$ and $\{\varepsilon_i\}_{i=1}^{n_q}$ are the quadrature weights and the zeros of the nth degree Hermite polynomial, respectively. By normalizing $\{\varpi_i\}_{i=1}^{n_q}$ and $\{\varepsilon_i\}_{i=1}^{n_q}$ as

$$\omega_i = \frac{1}{\sqrt{\pi}} \varpi_i, \quad z_i = \sqrt{2}\varepsilon_i, \quad i = 1, 2, \ldots, n_q \tag{36}$$

the integration $\int_{-\infty}^{+\infty} g(x) N(x;\, 0,\, 1) \mathrm{d}x$ can be expressed as

$$\int_{-\infty}^{+\infty} g(x) N(x;\, 0,\, 1)\, \mathrm{d}x \approx \sum_{i=1}^{n_q} \omega_i g(z_i). \tag{37}$$

In the L-dimensional multivariate function case, the 1-D quadrature weights $\{\omega_i\}_{i=1}^{n_q}$ and points $\{z_i\}_{i=1}^{n_q}$ are generated first, then the high-dimensional quadrature points $\{\mathbf{z}_i\}_{i=1}^{n_q^L}$ are constructed from the tensor product of 1-D points, and the high-dimensional quadrature weights $\{\omega_i\}_{i=1}^{n_q^L}$ are obtained by multiplying the corresponding 1-D weights together. For the ith integration in (34), supposing $\boldsymbol{\Sigma}_i$ can be decomposed as $\boldsymbol{\Sigma}_i = \mathbf{L}_i \mathbf{L}_i^T$, the affine transform $\mathbf{x} = \mathbf{L}_i \mathbf{z} + \boldsymbol{\mu}_i$ then maps $N(\boldsymbol{\mu}_i,\, \boldsymbol{\Sigma}_i)$ to $N(\mathbf{0},\, \mathbf{I})$. Now, the higher order Gauss–Hermite quadrature rule can be written as [30]

$$\int \mathbf{g}(\mathbf{x})\, N\left(\mathbf{x};\, \boldsymbol{\mu}_i,\, \boldsymbol{\Sigma}_i\right) \mathrm{d}\mathbf{x} \approx \sum_{j=1}^{n_q^L} \omega_j \mathbf{g}\left(\mathbf{L}_i \mathbf{z}_j + \boldsymbol{\mu}_i\right). \tag{38}$$

It is noted that the total quadrature point number is n_q^L, which grows rapidly as n_q increases, especially when the dimensionality L is large. Thus, there is the curse of dimensionality. To overcome this problem, a sparse grid quadrature rule, such as the Smolyak grid with m_q quadrature points, can be adopted to compute (38), where $m_q << n_q^L$ [31]–[33].

IV. GMM-BASED PCRB APPROXIMATION ALGORITHMS

Based on the aforementioned foundation, we can now develop a GMM-based framework to effectively compute the PCRB for a general nonlinear filtering problem. In summary, we use the GMM to propagate the state uncertainty and the Gauss–Hermite quadrature rule to compute the high-dimensional integrations.

Suppose that at time n, the FIM $\mathbf{J}_n$ is known, and the state pdf $p(\mathbf{x}_n)$ is well approximated by the following GMM:

$$p(\mathbf{x}_n) \approx \sum_{i=1}^{\alpha_n} w_{n,i}^{\mathbf{x}} N\left(\mathbf{x}_n;\, \boldsymbol{\mu}_{n,i}^{\mathbf{x}},\, \boldsymbol{\Sigma}_{n,i}^{\mathbf{x}}\right). \tag{39}$$

We want to obtain the FIM $\mathbf{J}_{n+1}$ and the propagated pdf $p(\mathbf{x}_{n+1})$.

A. Small Component Covariance Case

Let us consider first the small component covariance case, in which the covariance of each Gaussian component in (39) is sufficiently small. From (5)–(8), we must compute $\mathbf{D}_n^{11}$, $\mathbf{D}_n^{12}$, $\mathbf{D}_n^{21}$, and $\mathbf{D}_n^{22}$. Note that $\mathbf{D}_n^{11}$, $\mathbf{D}_n^{12}$, and $\mathbf{D}_n^{21}$ are related only to the state at time n, so we address these three terms first.

For $\mathbf{D}_n^{11}$, substituting (39) into (5) yields

$$\mathbf{D}_n^{11} \approx \sum_{i=1}^{\alpha_n} w_{n,i}^{\mathbf{x}} \int \left[\frac{\partial \mathbf{f}_n^T(\mathbf{x}_n)}{\partial \mathbf{x}_n} \mathbf{Q}_n^{-1} \frac{\partial \mathbf{f}_n(\mathbf{x}_n)}{\partial \mathbf{x}_n^T}\right] \times N\left(\mathbf{x}_n;\, \boldsymbol{\mu}_{n,i}^{\mathbf{x}},\, \boldsymbol{\Sigma}_{n,i}^{\mathbf{x}}\right) \mathrm{d}\mathbf{x}_n. \tag{40}$$

Denote the ith term in the right-hand side of (40) as $\mathbf{D}_{n,i}^{11}$, and let $\boldsymbol{\Sigma}_{n,i}^{\mathbf{x}} = \mathbf{L}_{n,i} \mathbf{L}_{n,i}^T$; then, by using the (sparse-grid) Gauss–Hermite quadrature rule with m_q points, $\mathbf{D}_{n,i}^{11}$ can be calculated as

$$\mathbf{D}_{n,i}^{11} \approx \sum_{j=1}^{m_q} \omega_j \tilde{\mathbf{F}}_n^T\left(\mathbf{L}_{n,i}\mathbf{z}_j + \boldsymbol{\mu}_{n,i}^{\mathbf{x}}\right) \mathbf{Q}_n^{-1} \tilde{\mathbf{F}}_n\left(\mathbf{L}_{n,i}\mathbf{z}_j + \boldsymbol{\mu}_{n,i}^{\mathbf{x}}\right) \tag{41}$$

where

$$\tilde{\mathbf{F}}_n\left(\mathbf{L}_{n,i}\mathbf{z}_j + \boldsymbol{\mu}_{n,i}^{\mathbf{x}}\right) = \left.\frac{\partial \mathbf{f}_n(\mathbf{x}_n)}{\partial \mathbf{x}_n^T}\right|_{\mathbf{x}_n = \mathbf{L}_{n,i}\mathbf{z}_j + \boldsymbol{\mu}_{n,i}^{\mathbf{x}}}. \tag{42}$$

Integrating for all components and summing yields

$$\begin{aligned}\mathbf{D}_n^{11} &\approx \sum_{i=1}^{\alpha_n} w_{n,i}^{\mathbf{x}} \mathbf{D}_{n,i}^{11} \\ &= \sum_{i=1}^{\alpha_n} \sum_{j=1}^{m_q} w_{n,i}^{\mathbf{x}} \omega_j \tilde{\mathbf{F}}_n^T\left(\mathbf{L}_{n,i}\mathbf{z}_j + \boldsymbol{\mu}_{n,i}^{\mathbf{x}}\right) \\ &\quad \times \mathbf{Q}_n^{-1} \tilde{\mathbf{F}}_n\left(\mathbf{L}_{n,i}\mathbf{z}_j + \boldsymbol{\mu}_{n,i}^{\mathbf{x}}\right).\end{aligned} \tag{43}$$

Similarly, $\mathbf{D}_n^{12}$ can be computed as

$$\mathbf{D}_{n,i}^{12} \approx \left[\sum_{j=1}^{m_q} \omega_j \tilde{\mathbf{F}}_n^T \left(\mathbf{L}_{n,i}\mathbf{z}_j + \boldsymbol{\mu}_{n,i}^{\mathbf{x}}\right)\right] \mathbf{Q}_n^{-1} \tag{44}$$

$$\mathbf{D}_n^{12} \approx \sum_{i=1}^{\alpha_n} w_{n,i}^{\mathbf{x}} \mathbf{D}_{n,i}^{12}. \tag{45}$$

From (6), $\mathbf{D}_n^{21}$ is simply the transpose of $\mathbf{D}_n^{12}$.

Next, we must address $\mathbf{D}_n^{22}$. A careful look at (8) reveals that only the second term of $\mathbf{D}_n^{22}$ must be computed. Denote the second term of $\mathbf{D}_n^{22}$ as $\hat{\mathbf{D}}_n^{22}$, which is related only to the state $\mathbf{x}_{n+1}$. Thus, the uncertainty propagation is required before calculating $\hat{\mathbf{D}}_n^{22}$. Suppose that the propagated GMM for $p(\mathbf{x}_{n+1})$ is

$$p\left(\mathbf{x}_{n+1}\right) \approx \sum_{i=1}^{\alpha_{n+1}} w_{n+1,i}^{\mathbf{x}} N\left(\mathbf{x}_{n+1}; \boldsymbol{\mu}_{n+1,i}^{\mathbf{x}}, \boldsymbol{\Sigma}_{n+1,i}^{\mathbf{x}}\right). \tag{46}$$

If the covariance matrices of all components in (39) are sufficiently small, the component number and weights can remain unchanged before and after the propagation [25]; namely

$$\alpha_{n+1} = \alpha_n \tag{47}$$

$$w_{n+1,i}^{\mathbf{x}} = w_{n,i}^{\mathbf{x}}. \tag{48}$$

For each Gaussian component in (39), its propagated version is assumed to be a new Gaussian component in (46). Strictly speaking, the Gaussian-to-Gaussian mapping is only valid for linear systems. For a nonlinear system represented by (1), this mapping is an approximation. However, if the component covariance is sufficiently small, the system can be well represented by a linear model around the corresponding component mean and such an approximation becomes reasonable. Therefore, before the uncertainty propagation step, it is necessary to verify whether (39) has the property of small component covariance.

For the ith Gaussian term in (39), let the nonlinearity metric $\Xi_{n,i}$ corresponding to $N(\boldsymbol{\mu}_{n,i}^{\mathbf{x}}, \boldsymbol{\Sigma}_{n,i}^{\mathbf{x}})$ be defined as follows:

$$\Xi_{n,i} = \sup_{\hat{\mathbf{u}}} \frac{\left\|\tilde{\mathbf{F}}_n\left(\boldsymbol{\mu}_{n,i}^{\mathbf{x}} + 3\sigma_{\hat{\mathbf{u}}}\hat{\mathbf{u}}\right) - \tilde{\mathbf{F}}_n\left(\boldsymbol{\mu}_{n,i}^{\mathbf{x}}\right)\right\|_2}{\left\|\tilde{\mathbf{F}}_n\left(\boldsymbol{\mu}_{n,i}^{\mathbf{x}}\right)\right\|_2} \tag{49}$$

where $3\sigma_{\hat{\mathbf{u}}}\hat{\mathbf{u}}$ denote a point which is along the direction of $\hat{\mathbf{u}}$ and located at the 3-σ error ellipsoid determined by $\boldsymbol{\Sigma}_{n,i}^{\mathbf{x}}$. Note that the nonlinearity metric should be evaluated along several different directions and the maximum value is selected.

For the whole GMM, the nonlinearity metric is defined as the weighted sum of its component nonlinearity metrics, namely

$$\Xi_n = \sum_{i=1}^{\alpha_n} w_{n,i}^{\mathbf{x}} \Xi_{n,i}. \tag{50}$$

A small nonlinearity metric means the nonlinearity of the system has little impact on the Gaussian-to-Gaussian mapping of components involved in the uncertainty propagation process, and the propagated GMM will be a close approximation of the state pdf at time $n+1$ supposing (39) is a good approximation of $p(\mathbf{x}_n)$.

In practice, the nonlinearity metric should be compared with a problem-related threshold. If the metric is less than the threshold, the current iteration is classified into the small component covariance case. Otherwise, it should be treated as the large component covariance case.

In the small component covariance case, the Gaussian-to-Gaussian mapping of a component requires propagating its mean and covariance only. If the linear propagator is adopted, then

$$\boldsymbol{\mu}_{n+1,i}^{\mathbf{x}} = \mathbf{f}_n\left(\boldsymbol{\mu}_{n,i}^{\mathbf{x}}\right) \tag{51}$$

$$\boldsymbol{\Sigma}_{n+1,i}^{\mathbf{x}} = \tilde{\mathbf{F}}_n(\boldsymbol{\mu}_{n,i}^{\mathbf{x}})\boldsymbol{\Sigma}_{n,i}^{\mathbf{x}}\tilde{\mathbf{F}}_n^T(\boldsymbol{\mu}_{n,i}^{\mathbf{x}}) + \mathbf{Q}_n. \tag{52}$$

Although the linear propagator is simple, however, it may produce inaccurate or even ill-conditioned solutions if the system is highly nonlinear or the propagation time is too long [34]. When this type of anomaly occurs, we can use the SUT approach to replace the linearization method. For the SUT propagator, the weights and sigma points $\{W_l^m, W_l^c, \boldsymbol{\xi}_{n,l}\}_{l=0}^{2L}$ are first generated based on $N(\mathbf{x}_n; \boldsymbol{\mu}_{n,i}^{\mathbf{x}}, \boldsymbol{\Sigma}_{n,i}^{\mathbf{x}})$, then

$$\boldsymbol{\eta}_{n+1,l} = \mathbf{f}_n\left(\boldsymbol{\xi}_{n,l}\right) \tag{53}$$

$$\boldsymbol{\mu}_{n+1,i}^{\mathbf{x}} = \sum_{l=0}^{2L} W_l^m \boldsymbol{\eta}_{n+1,l} \tag{54}$$

$$\boldsymbol{\Sigma}_{n+1,i}^{\mathbf{x}} = \sum_{l=0}^{2L} W_l^c \left(\boldsymbol{\eta}_{n+1,l} - \boldsymbol{\mu}_{n+1,i}^{\mathbf{x}}\right)\left(\boldsymbol{\eta}_{n+1,l} - \boldsymbol{\mu}_{n+1,i}^{\mathbf{x}}\right)^T + \mathbf{Q}_n. \tag{55}$$

After the uncertainty propagation, $\mathbf{D}_n^{22}$ now can be computed as follows:

$$\begin{aligned}\hat{\mathbf{D}}_{n,i}^{22} &= \mathbf{E}\left\{\frac{\partial \mathbf{h}_{n+1}^T\left(\mathbf{x}_{n+1}\right)}{\partial \mathbf{x}_{n+1}} \mathbf{R}_{n+1}^{-1} \frac{\partial \mathbf{h}_{n+1}\left(\mathbf{x}_{n+1}\right)}{\partial \mathbf{x}_{n+1}^T}\right\} \\ &\approx \sum_{j=1}^{m_q} \omega_j \tilde{\mathbf{H}}_{n+1}^T \left(\mathbf{L}_{n+1,i}\mathbf{z}_j + \boldsymbol{\mu}_{n+1,i}^{\mathbf{x}}\right) \\ &\qquad \times \mathbf{R}_{n+1}^{-1} \tilde{\mathbf{H}}_{n+1}\left(\mathbf{L}_{n+1,i}\mathbf{z}_j + \boldsymbol{\mu}_{n+1,i}^{\mathbf{x}}\right)\end{aligned} \tag{56}$$

$$\mathbf{D}_n^{22} \approx \mathbf{Q}_n^{-1} + \sum_{i=1}^{\alpha_{n+1}} w_{n+1,i}^{\mathbf{x}} \hat{\mathbf{D}}_{n,i}^{22} \tag{57}$$

where

$$\boldsymbol{\Sigma}_{n+1,i}^{\mathbf{x}} = \mathbf{L}_{n+1,i}\mathbf{L}_{n+1,i}^T \tag{58}$$

$$\tilde{\mathbf{H}}_{n+1}\left(\mathbf{L}_{n+1,i}\mathbf{z}_j + \boldsymbol{\mu}_{n+1,i}^{\mathbf{x}}\right) = \left.\frac{\partial \mathbf{h}_{n+1}\left(\mathbf{x}_{n+1}\right)}{\partial \mathbf{x}_{n+1}^T}\right|_{\mathbf{x}_{n+1}=\mathbf{L}_{n+1,i}\mathbf{z}_j+\boldsymbol{\mu}_{n+1,i}^{\mathbf{x}}}. \tag{59}$$

Finally, the FIM is obtained via (4). The complete procedure for one step of the iteration is summarized in Algorithm 1.

Algorithm 1: PCRB With Small Component Covariance.

input : $\left\{\left(w_{n,i}^{\mathbf{x}},\ \boldsymbol{\mu}_{n,i}^{\mathbf{x}},\ \boldsymbol{\Sigma}_{n,i}^{\mathbf{x}}\right)\right\}_{i=1}^{\alpha_n}$, $\mathbf{Q}_n$, $\mathbf{R}_{n+1}$ and $\mathbf{J}_n$
output: $\left\{\left(w_{n+1,i}^{\mathbf{x}},\ \boldsymbol{\mu}_{n+1,i}^{\mathbf{x}},\ \boldsymbol{\Sigma}_{n+1,i}^{\mathbf{x}}\right)\right\}_{i=1}^{\alpha_{n+1}}$ and $\mathbf{J}_{n+1}$

generate quadrature weights and nodes $\left\{(\omega_j,\ \mathbf{z}_j)\right\}_{j=1}^{m_q}$;

$\mathbf{D}_n^{11} \leftarrow 0$;
for $i \leftarrow 1$ **to** α_n **do**
  compute $\mathbf{D}_{n,i}^{11}$ according to Eq.(41);
  $\mathbf{D}_n^{11} \leftarrow \mathbf{D}_n^{11} + w_{n,i}^{\mathbf{x}}\mathbf{D}_{n,i}^{11}$;
end

$\mathbf{D}_n^{12} \leftarrow 0$;
for $i \leftarrow 1$ **to** α_n **do**
  compute $\mathbf{D}_{n,i}^{12}$ according to Eq.(44) ;
  $\mathbf{D}_n^{12} \leftarrow \mathbf{D}_n^{12} + w_{n,i}^{\mathbf{x}}\mathbf{D}_{n,i}^{12}$;
end

$\mathbf{D}_n^{21} \leftarrow \left[\mathbf{D}_n^{12}\right]^T$;

$\alpha_{n+1} \leftarrow \alpha_n$;
for $i \leftarrow 1$ **to** α_{n+1} **do**
  $w_{n+1,i}^{\mathbf{x}} \leftarrow w_{n,i}^{\mathbf{x}}$;
  compute $\boldsymbol{\mu}_{n+1,i}^{\mathbf{x}}$ according to Eq.(51) or Eq.(54);
  compute $\boldsymbol{\Sigma}_{n+1,i}^{\mathbf{x}}$ according to Eq.(52) or Eq.(55);
end

$\mathbf{D}_n^{22} \leftarrow \mathbf{Q}_n^{-1}$;
for $i \leftarrow 1$ **to** α_{n+1} **do**
  compute $\hat{\mathbf{D}}_{n,i}^{22}$ according to Eq.(56);
  $\mathbf{D}_n^{22} \leftarrow \mathbf{D}_n^{22} + w_{n+1,i}^{\mathbf{x}}\hat{\mathbf{D}}_{n,i}^{22}$;
end

compute $\mathbf{J}_{n+1}$ according to Eq.(4);

B. Large Component Covariance Case

The algorithm for the PCRB in the large component covariance case is almost the same as that in the small component covariance case, except for the uncertainty propagation step. When propagating the state GMM, either the nonlinear system equation or the process noise could enlarge the component covariance, which may cause the GMM nonlinearity metric to exceed the threshold. This is referred to as the covariance expansion phenomenon. For example, in (52), if the determinant of the Jacobian matrix $\tilde{\mathbf{F}}_n(\boldsymbol{\mu}_{n,l}^{\mathbf{x}})$ is greater than one, the component covariance $\boldsymbol{\Sigma}_{n,i}^{\mathbf{x}}$ will be enlarged after multiplying it by $\tilde{\mathbf{F}}_n(\boldsymbol{\mu}_{n,i}^{\mathbf{x}})$ and $\tilde{\mathbf{F}}_n^T(\boldsymbol{\mu}_{n,i}^{\mathbf{x}})$, respectively, and the additive term $\mathbf{Q}_n$ will further enlarge the propagated covariance.

When the GMM nonlinearity metric exceeds the threshold, the uncertainty propagation strategy used in the small component covariance case can introduce great error, which finally leads to dramatic accuracy loss in computing $\mathbf{D}_n^{22}$. To overcome this, each component in (39) should be refined according to (20) to reduce its covariance, and the yielding GMM can be expressed as

$$p\left(\mathbf{x}_n\right) \approx \sum_{i=1}^{\alpha_n^+} w_{n,i}^{\mathbf{x}} N\left(\mathbf{x}_n;\ \boldsymbol{\mu}_{n,i}^{\mathbf{x}},\ \boldsymbol{\Sigma}_{n,i}^{\mathbf{x}}\right) \tag{60}$$

where $\alpha_n^+ > \alpha_n$. Furthermore, if the process noise covariance $\mathbf{Q}_n$ is equal or greater than each component covariance in (39), it is suggested that the process noise pdf $p(\mathbf{w}_n)$ should also be refined as

$$p\left(\mathbf{w}_n\right) \approx \sum_{j=1}^{\beta_n} w_{n,j}^{\mathbf{w}} N\left(\mathbf{w}_n;\ \boldsymbol{\mu}_{n,j}^{\mathbf{w}},\ \boldsymbol{\Sigma}_{n,j}^{\mathbf{w}}\right). \tag{61}$$

For the ith component in (60) and the jth component in (61), $\mathbf{x}_n \sim N(\mathbf{x}_n;\ \boldsymbol{\mu}_{n,i}^{\mathbf{x}},\ \boldsymbol{\Sigma}_{n,i}^{\mathbf{x}})$ and $\mathbf{w}_n \sim N(\mathbf{w}_n;\ \boldsymbol{\mu}_{n,j}^{\mathbf{w}},\ \boldsymbol{\Sigma}_{n,j}^{\mathbf{w}})$. Linearizing (1) at $(\boldsymbol{\mu}_{n,i}^{\mathbf{x}},\ \boldsymbol{\mu}_{n,j}^{\mathbf{w}})$ yields

$$\begin{aligned}\mathbf{x}_{n+1} \approx\ & \mathbf{f}_n\left(\boldsymbol{\mu}_{n,i}^{\mathbf{x}}\right) + \tilde{\mathbf{F}}_n\left(\mathbf{x}_n - \boldsymbol{\mu}_{n,i}^{\mathbf{x}}\right) \\ & + \boldsymbol{\mu}_{n,j}^{\mathbf{w}} + \left(\mathbf{w}_n - \boldsymbol{\mu}_{n,j}^{\mathbf{w}}\right).\end{aligned} \tag{62}$$

Therefore, the linear uncertainty propagator should yield the propagated component mean and covariance as

$$\boldsymbol{\mu}_{n+1,ij}^{\mathbf{x}} = \mathbf{f}_n\left(\boldsymbol{\mu}_{n,i}^{\mathbf{x}}\right) + \boldsymbol{\mu}_{n,j}^{\mathbf{w}} \tag{63}$$

$$\boldsymbol{\Sigma}_{n+1,ij}^{\mathbf{x}} = \tilde{\mathbf{F}}_n(\boldsymbol{\mu}_{n,i}^{\mathbf{x}})\boldsymbol{\Sigma}_{n,i}^{\mathbf{x}}\tilde{\mathbf{F}}_n^T(\boldsymbol{\mu}_{n,i}^{\mathbf{x}}) + \boldsymbol{\Sigma}_{n,j}^{\mathbf{w}}. \tag{64}$$

The propagated GMM for $p(\mathbf{x}_{n+1})$ now becomes

$$p\left(\mathbf{x}_{n+1}\right) \approx \sum_{i=1}^{\alpha_n^+}\sum_{j=1}^{\beta_n} w_{n,i}^{\mathbf{x}} w_{n,j}^{\mathbf{w}} N\left(\mathbf{x}_{n+1};\ \boldsymbol{\mu}_{n+1,ij}^{\mathbf{x}},\ \boldsymbol{\Sigma}_{n+1,ij}^{\mathbf{x}}\right). \tag{65}$$

For the SUT propagator, the weights and sigma points $\{W_l^m, W_l^c, \boldsymbol{\xi}_{n,l}\}_{l=0}^{2L}$ are generated first based on $N(\mathbf{x}_n;\ \boldsymbol{\mu}_{n,i}^{\mathbf{x}},\ \boldsymbol{\Sigma}_{n,i}^{\mathbf{x}})$, and then the propagated mean and covariance are calculated as

$$\boldsymbol{\eta}_{n+1,l} = \mathbf{f}_n\left(\boldsymbol{\xi}_{n,l}\right) + \boldsymbol{\mu}_{n,j}^{\mathbf{w}} \tag{66}$$

$$\boldsymbol{\mu}_{n+1,ij}^{\mathbf{x}} = \sum_{l=0}^{2L} W_l^m \boldsymbol{\eta}_{n+1,l} \tag{67}$$

$$\begin{aligned}\boldsymbol{\Sigma}_{n+1,ij}^{\mathbf{x}} =\ & \sum_{l=0}^{2L} W_l^c\left(\boldsymbol{\eta}_{n+1,l} - \boldsymbol{\mu}_{n+1,ij}^{\mathbf{x}}\right)\left(\boldsymbol{\eta}_{n+1,l} - \boldsymbol{\mu}_{n+1,ij}^{\mathbf{x}}\right)^T \\ & + \boldsymbol{\Sigma}_{n,j}^{\mathbf{w}}.\end{aligned} \tag{68}$$

Although the simultaneous refinement for $p(\mathbf{x}_n)$ and $p(\mathbf{w}_n)$ mitigates the covariance expansion, it causes the Gaussian terms to grow exponentially as time passes, which is known as the growing memory problem. As a result, the GMM for the state pdf soon becomes intractable. To restrict the amount of computation, an additional coarsening operation is needed. Before starting the computation of $\mathbf{D}_n^{22}$, the propagated GMM for $p(\mathbf{x}_{n+1})$ is coarsened by merging the components to a predefined number K. The choice of K should guarantee that the L_2 distance between the propagated and coarsened GMMs does not exceed a threshold. The calculation of the L_2 distance between two GMMs can be found in [34].

The K-means clustering algorithm can be used to effectively coarsen a GMM. A key point in the K-means algorithm is the selection of a proper distance to measure the closeness between a pair of data. However, a component in a GMM is characterized by a triplet $(w_i, \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$, so traditional distances, such as the Euclidian distance, cannot

be used directly for the GMM clustering. Instead, we use the weighted Mahalanobis distance [35] between two Gaussian components, which is defined as

$$d_{ij}^2 = \frac{w_i w_j}{w_i + w_j}\left(\boldsymbol{\mu}_i - \boldsymbol{\mu}_j\right)^T \left(\boldsymbol{\Sigma}_i + \boldsymbol{\Sigma}_j\right)^{-1} \left(\boldsymbol{\mu}_i - \boldsymbol{\mu}_j\right). \quad (69)$$

Another difficulty is how to calculate the cluster center when a set of Gaussian components is confirmed to belong to the same cluster. This is essentially asking how to merge a set of Gaussian components into a single Gaussian term. A merge scheme that preserves the mean and covariance has been proposed as

$$w_C = \sum_{i \in C} w_i \quad (70)$$

$$\boldsymbol{\mu}_C = \frac{1}{w_C} \sum_{i \in C} w_i \boldsymbol{\mu}_i \quad (71)$$

$$\boldsymbol{\Sigma}_C = \frac{1}{w_C} \sum_{i \in C} w_i \left(\boldsymbol{\Sigma}_i + \boldsymbol{\mu}_i \boldsymbol{\mu}_i^T\right) - \boldsymbol{\mu}_C \boldsymbol{\mu}_C^T \quad (72)$$

where C is the index set in which the components are to be merged, and w_C, $\boldsymbol{\mu}_C$, and $\boldsymbol{\Sigma}_C$ are the weight, mean, and covariance of the cluster center, respectively [35].

The detailed steps for an iteration in the large component covariance case are demonstrated in Algorithm 2. In the Appendix, the detailed algorithm for the GMM K-means clustering is presented.

Algorithm 2: PCRB with Large Component Covariance.

input : $\left\{\left(w_{n,i}^{\mathbf{x}},\ \boldsymbol{\mu}_{n,i}^{\mathbf{x}},\ \boldsymbol{\Sigma}_{n,i}^{\mathbf{x}}\right)\right\}_{i=1}^{\alpha_n}$, $\mathbf{Q}_n$, $\mathbf{R}_{n+1}$ and $\mathbf{J}_n$
output: $\left\{\left(w_{n+1,i}^{\mathbf{x}},\ \boldsymbol{\mu}_{n+1,i}^{\mathbf{x}},\ \boldsymbol{\Sigma}_{n+1,i}^{\mathbf{x}}\right)\right\}_{i=1}^{\alpha_{n+1}}$ and $\mathbf{J}_{n+1}$

generate quadrature weights and nodes $\left\{\left(\omega_j,\ \mathbf{z}_j\right)\right\}_{j=1}^{m_q}$;

$\mathbf{D}_n^{11} \leftarrow 0$;
for $i \leftarrow 1$ **to** α_n **do**
  compute $\mathbf{D}_{n,i}^{11}$;
  $\mathbf{D}_n^{11} \leftarrow \mathbf{D}_n^{11} + w_{n,i}^{\mathbf{x}} \mathbf{D}_{n,i}^{11}$;
end

$\mathbf{D}_n^{12} \leftarrow 0$;
for $i \leftarrow 1$ **to** α_n **do**
  compute $\mathbf{D}_{n,i}^{12}$;
  $\mathbf{D}_n^{12} \leftarrow \mathbf{D}_n^{12} + w_{n,i}^{\mathbf{x}} \mathbf{D}_{n,i}^{12}$;
end

$\mathbf{D}_n^{21} \leftarrow \left[\mathbf{D}_n^{12}\right]^T$;

refine the state GMM into $\left\{\left(w_{n,i}^{\mathbf{x}}, \boldsymbol{\mu}_{n,i}^{\mathbf{x}}, \boldsymbol{\Sigma}_{n,i}^{\mathbf{x}}\right)\right\}_{i=1}^{\alpha_n^+}$ by splitting;
split the process noise into a GMM $\left\{\left(w_{n,j}^{\mathbf{w}}, \boldsymbol{\mu}_{n,j}^{\mathbf{w}}, \boldsymbol{\Sigma}_{n,j}^{\mathbf{w}}\right)\right\}_{j=1}^{\beta_n}$;

$\alpha_{n+1}^+ \leftarrow \alpha_n^+ \beta_n$;
for $i \leftarrow 1$ **to** α_n^+ **do**
  for $j \leftarrow 1$ **to** β_n **do**
    $w_{n+1,ij}^{\mathbf{x}} \leftarrow w_{n,i}^{\mathbf{x}} w_{n,j}^{\mathbf{w}}$;
    compute $\boldsymbol{\mu}_{n+1,ij}^{\mathbf{x}}$ according to Eq.(63) or Eq.(67);
    compute $\boldsymbol{\Sigma}_{n+1,ij}^{\mathbf{x}}$ according to Eq.(64) or Eq.(68);
  end
end

$\left\{\left(w_{n+1,i}^{\mathbf{x}}, \boldsymbol{\mu}_{n+1,i}^{\mathbf{x}}, \boldsymbol{\Sigma}_{n+1,i}^{\mathbf{x}}\right)\right\}_{i=1}^{\alpha_{n+1}} \leftarrow$ `KMeans`$\left(\left\{\left(w_{n+1,ij}^{\mathbf{x}}, \boldsymbol{\mu}_{n+1,ij}^{\mathbf{x}}, \boldsymbol{\Sigma}_{n+1,ij}^{\mathbf{x}}\right)\right\}_{ij=1}^{\alpha_{n+1}^+}, \alpha_{n+1}\right)$;

$\mathbf{D}_n^{22} \leftarrow \mathbf{Q}_n^{-1}$;
for $i \leftarrow 1$ **to** α_{n+1} **do**
  compute $\hat{\mathbf{D}}_{n,i}^{22}$;
  $\mathbf{D}_n^{22} \leftarrow \mathbf{D}_n^{22} + w_{n+1,i}^{\mathbf{x}} \hat{\mathbf{D}}_{n,i}^{22}$;
end

compute $\mathbf{J}_{n+1}$ according to Eq.(4);

C. Computational Complexity Analysis

A key performance consideration about the proposed algorithm is its computational complexity. In this section, we present the computational complexity analysis for both the small and large component covariance cases. In each iteration, since computing $\mathbf{D}_n^{11}$, $\mathbf{D}_n^{12}$, $\mathbf{D}_n^{21}$, $\mathbf{D}_n^{22}$ and propagating the state uncertainty consume most of the time, we need only to analyze the operations related to those steps.

In the small component covariance case, the matrix $\mathbf{D}_n^{11}$ is first computed. According to (41), if we treat the operation of multiplying a matrix by a scalar weight as multiplying the matrix by the corresponding diagonal matrix, then for each $\mathbf{D}_{n,i}^{11}$, it requires m_q Jacobian matrix evaluations, m_q matrix transposes, $3m_q$ matrix multiplications, and $m_q - 1$ matrix additions. Thus, computing $\mathbf{D}_n^{11}$ needs total $\alpha_n m_q$ Jacobian matrix evaluations, $\alpha_n m_q$ matrix transposes, $3\alpha_n m_q + \alpha_n$ matrix multiplications, and $\alpha_n m_q - 1$ matrix additions according to (43).

Suppose that all variables involved in computing $\mathbf{D}_n^{11}$ are stored, then computing $\mathbf{D}_n^{12}$ requires additional $\alpha_n m_q + 2\alpha_n$ matrix multiplications and $\alpha_n m_q - 1$ matrix additions according to (44) and (45). After computing $\mathbf{D}_n^{12}$, only one matrix transpose is required to yield the term $\mathbf{D}_n^{21}$.

Similarly, according to (56) and (57), the computation of $\mathbf{D}_n^{22}$ requires $\alpha_{n+1} m_q$ Jacobian matrix evaluations, $\alpha_{n+1} m_q$ matrix transposes, $3\alpha_{n+1} m_q + \alpha_{n+1}$ matrix multiplications, and $\alpha_{n+1} m_q$ matrix additions.

Before computing $\mathbf{D}_n^{22}$, the uncertainty propagation is conducted to yield (46). For the small covariance matrix case, if the linear propagator is adopted, then under the condition of $\alpha_{n+1} = \alpha_n$, propagating component means only need α_n function evaluations according to (51); propagating component covariances need α_n Jacobian matrix evaluations, α_n matrix transposes, $2\alpha_n$ matrix multiplications, and α_n matrix additions according to (52). Otherwise if the SUT propagator is adopted, it takes $\alpha_n(2L+1)$ function evaluations, $\alpha_n(2L+1)$ matrix multiplications, and $2\alpha_n L$ matrix additions to propagate all component means according to (53) and (54). For propagating component covariances, it requires additional $2\alpha_n(2L+1)$ matrix additions, $\alpha_n(2L+1)$ matrix transposes, and $2\alpha_n(2L+1)$ matrix multiplications according to (55).

In general, the number of quadrature nodes m_q is greater than $2L+1$. Then, based on the aforementioned analysis, we can conclude that the computational complexity for the small component covariance case is $O(\alpha_n m_q)$. It is noted that $\alpha_n m_q$ is the total number of quadrature points for computing the high-dimensional integrations of (5)–(8).

When it comes to the large component covariance case, the only difference lies in the uncertainty propagation step. Due to the refinement for $p(\mathbf{x}_n)$ and $p(\mathbf{w}_n)$, the number of components in the propagated GMM will be $\alpha_n^+\beta_n$. For the linear propagator, it takes α_n^+ function evaluations and $\alpha_n^+\beta_n$ matrix additions to obtain all propagated means according to (63), and it takes α_n^+ Jacobian matrix evaluations, α_n^+ matrix transposes, $2\alpha_n^+$ matrix multiplications, and $\alpha_n^+\beta_n$ matrix additions to obtain all propagated covariances according to (64). For the SUT propagator, it takes total $\alpha_n^+(2L+1)$ function evaluations, $\alpha_n^+\beta_n(4L+1)$ matrix additions, and $\alpha_n^+\beta_n(2L+1)$ matrix multiplications to propagate all component means according to (66) and (67), and it takes additional $2\alpha_n^+\beta_n(2L+1)$ matrix additions, $\alpha_n^+\beta_n(2L+1)$ matrix transposes, and $2\alpha_n^+\beta_n(2L+1)$ matrix multiplications to propagate all component covariances according to (68).

After the uncertainty propagation step, it usually needs to invoke the K-means algorithm to reduce the GMM to K components. For typical $\alpha_n^+\beta_n$ and K values, this K-means clustering step can consume the majority of the total computation time. For the K-means algorithm itself, the computation of the weighted Mahalanobis distance (69) determines its computational complexity. In each iteration of K-means, it computes the distance $\alpha_n^+\beta_n K$ times. In the worst case, it takes $I_{\max}$ iterations before the clustering result converges, where $I_{\max}$ is the maximum allowed number of iterations. Therefore, the worst-case computational complexity is $O(\alpha_n^+\beta_n K I_{\max})$.

The proposed method is expected to be more efficient than Monte Carlo for integrating (5)–(8), especially in the small component covariance case. This is because the GMM is more effective for approximating the state pdf than Monte Carlo, which approximates $p(\mathbf{x}_n)$ as

$$p(\mathbf{x}_n) \approx \frac{1}{N}\sum_i \delta\left(\mathbf{x}_n - \boldsymbol{\chi}_{n,i}\right) \tag{73}$$

where N is the number of samples, $\boldsymbol{\chi}_{n,i}$ is the ith realization of $\mathbf{x}_n$. The GMM can retain the pdf value at any point, whereas the Monte Carlo approach preserves the pdf values only at discrete points. The computational complexity for Monte Carlo integration is $O(N)$, whereas the computational complexity for the proposed method is $O(\alpha_n m_q)$ in the small component covariance case. Due to the effectiveness of GMM in representing the state uncertainty, it usually has $\alpha_n m_q < N$ to achieve the same computational accuracy. In special cases where the terms in (5)–(8) for mathematical expectations can be approximated well by polynomials, these expectations can be accurately obtained from weighted sums of higher order moments of the Gaussian components.

D. Discussion

In the GMM-based PCRB computation algorithms, there are multiple types of approximations. The first approximation is representing the true state pdf $p(\mathbf{x}_n)$ as a GMM. Then, the computation of $\mathbf{D}_n^{ij}(i=1,2;\ j=1,2)$ comes down to computing the high-dimensional integration of the product of a nonlinear function and a general Gaussian pdf. This integration is approximately solved by the (sparse-grid) Gauss–Hermite quadrature rule. Before computing $\mathbf{D}_n^{22}$, an uncertainty propagation step is required to obtain $p(\mathbf{x}_{n+1})$. This step is done by propagating each Gaussian component separately. For propagating a Gaussian component, either the linearization or the SUT approach is used to approximate the propagated mean and covariance.

In addition to the aforementioned summary for the approximation techniques, we have some additional notes for the proposed algorithms, which are as follows.

1) The term $\partial\mathbf{f}_n(\mathbf{x}_n)/\partial\mathbf{x}_n^T$ appearing in (5)–(7) is also referred to as the state transition matrix. In situations where the evolution of the state is described by a group of ordinary differential equations rather than an explicit form of $\mathbf{f}_n(\cdot)$, the state transition matrix can be obtained by numerically integrating the variational equation. The reference state should be the corresponding Gauss–Hermite quadrature point.
2) The proposed method can be viewed as a special implementation of the sigma-point method. We use deterministically chosen samples of $p(\mathbf{x}_n)$ to compute the expectations of nonlinear functions of $\mathbf{x}_n$. However, the pdf $p(\mathbf{x}_n)$ is difficult to sample due to its unknown expression. Instead, we construct a GMM as a surrogate model for $p(\mathbf{x}_n)$ and sample from the GMM. As a contrast, Monte Carlo based approaches use stochastically chosen samples of $p(\mathbf{x}_n)$, as given by (73).

V. COMPUTATION EXAMPLES

A. Linear System

Consider the following linear system and measurement models:

$$\mathbf{x}_{n+1} = \mathbf{F}_n\mathbf{x}_n + \mathbf{w}_n \tag{74}$$

$$\mathbf{z}_{n+1} = \mathbf{H}_{n+1}\mathbf{x}_{n+1} + \mathbf{v}_{n+1} \tag{75}$$

where $\mathbf{F}_n$ and $\mathbf{H}_{n+1}$ are the system and measurement matrices, respectively. This linear model is a special case of the general nonlinear model expressed by (1) and (2). The PCRB for $\mathbf{x}_n$ can be easily determined.

From (5), we can directly obtain $\mathbf{D}_n^{11} = \mathbf{F}_n^T\mathbf{Q}_n^{-1}\mathbf{F}_n$. From (43), $\mathbf{D}_n^{11}$ is computed as

$$\mathbf{D}_n^{11} \approx \sum_{i=1}^{\alpha_n}\sum_{j=1}^{m_q} w_{n,i}^{\mathbf{x}}\omega_j\left(\mathbf{F}_n^T\mathbf{Q}_n^{-1}\mathbf{F}_n\right) = \mathbf{F}_n^T\mathbf{Q}_n^{-1}\mathbf{F}_n. \tag{76}$$

This is because both the sums of $\{w_{n,i}^{\mathbf{x}}\}_{i=1}^{\alpha_n}$ and $\{\omega_j\}_{j=1}^{m_q}$ equal to one. Thus, the proposed method has no approximation

error when computing $\mathbf{D}_n^{11}$ in the linear case. The same conclusion holds for $\mathbf{D}_n^{12}$ and $\mathbf{D}_n^{21}$.

For $\mathbf{D}_n^{22}$, we should have $\mathbf{D}_n^{22} = \mathbf{Q}_n^{-1} + \mathbf{H}_{n+1}^T \mathbf{R}_{n+1}^{-1} \mathbf{H}_{n+1}$. In the small component covariance case

$$\begin{aligned}\mathbf{D}_n^{22} &\approx \mathbf{Q}_n^{-1} + \sum_{i=1}^{\alpha_{n+1}} \sum_{j=1}^{m_q} w_{n+1,i}^{\mathbf{x}} \omega_j \left(\mathbf{H}_{n+1}^T \mathbf{R}_{n+1}^{-1} \mathbf{H}_{n+1}\right) \\ &= \mathbf{Q}_n^{-1} + \mathbf{H}_{n+1}^T \mathbf{R}_{n+1}^{-1} \mathbf{H}_{n+1} \end{aligned} \tag{77}$$

while in the large component covariance case

$$\begin{aligned}\mathbf{D}_n^{22} &\approx \mathbf{Q}_n^{-1} + \sum_{i=1}^{\alpha_n} \sum_{j=1}^{\beta_n} \sum_{k=1}^{m_q} w_{n,i}^{\mathbf{x}} w_{n,j}^{\mathbf{w}} \omega_k \left(\mathbf{H}_{n+1}^T \mathbf{R}_{n+1}^{-1} \mathbf{H}_{n+1}\right) \\ &= \mathbf{Q}_n^{-1} + \mathbf{H}_{n+1}^T \mathbf{R}_{n+1}^{-1} \mathbf{H}_{n+1}. \end{aligned} \tag{78}$$

Therefore, our method has no approximation error for the PCRB in linear cases. It has been proved that in linear cases, the yielding PCRB according to (4) equals the well-known Kalman mse matrix [15], so the Kalman filter is said to be optimal.

B. 1-D Benchmark Problem

Now, consider the following 1-D benchmark problem:

$$x_{n+1} = \frac{x_n}{2} + \frac{25x_n}{1+x_n^2} + 8\cos(1.2n) + w_n \tag{79}$$

$$z_{n+1} = \frac{x_{n+1}^2}{20} + v_{n+1}. \tag{80}$$

This model has been used already for testing PFs [6] and the SMC-based PCRB approximation algorithm [21]. First, the model parameters are set as

$$x_0 \sim N(0, 1) \tag{81}$$

$$\sigma_w^2 = 10, \quad \sigma_v^2 = 1. \tag{82}$$

In this model, determining the PCRB for estimating x_n is challenging because both the system and measurement equations are highly nonlinear, and the covariance of the process noise is large—even larger than the initial state covariance.

Next, the PCRB is computed by the plain Monte Carlo integration and the proposed method, respectively, and the results are compared. For the GMM approach, the component number is selected as $K = 64$ at first. The first splitting library presented in Section III-A with $\sigma = 1/2$ and $M = 8$ is applied four times for refining $p(x_0)$, yielding $8^4 = 4096$ components. The GMM for $p(x_0)$ is then merged to 64 components, and the iteration for computing PCRB begins. Due to the large process noise, after completing the computation of $\mathbf{D}_n^{11}$, $\mathbf{D}_n^{12}$, and $\mathbf{D}_n^{21}$ in each iteration, the state pdf is refined by the first splitting library with $\sigma = 1/40$ and $M = 16$, yielding $64 \times 16 = 1024$ components for the state. Meanwhile, the process noise pdf is refined by the first splitting library with $\sigma = 1/2$ and $M = 8$. The linear propagator is used for the uncertainty propagation, and the total number Gaussian terms after the uncertainty propagation is $1024 \times 8 = 8192$. The 8192 components are subsequently merged into 64 components. The numerical integrations

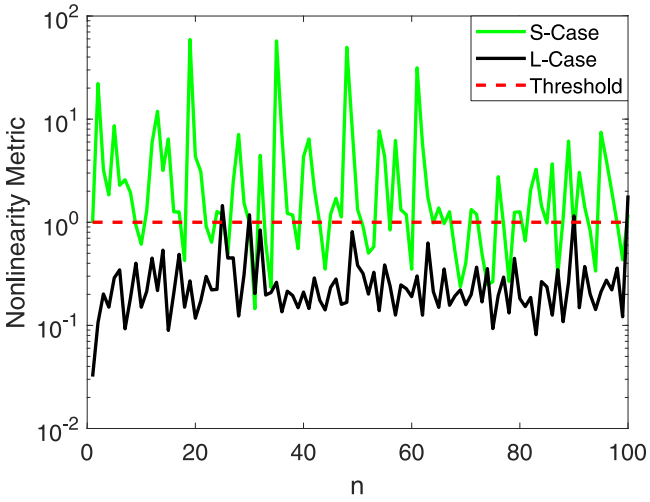


Fig. 1. GMM nonlinearity metrics under small and large component covariance assumptions.

for computing $\mathbf{D}_n^{11}$, $\mathbf{D}_n^{12}$, $\mathbf{D}_n^{21}$, and $\mathbf{D}_n^{22}$ are conducted by invoking the integral function built in MATLAB. The total computation time of this 64-component GMM approach is 1400.712 s. However, the profiler indicates that 99.7% of the computation time is spent on the clustering process. Therefore, an improved high-efficiency implementation of the K-means algorithm can greatly reduce the time consumption. Setting $K = 32$ and retaining all other configurations leads to a total computation time of 225.141 s, 99.0% of which is spent on the clustering.

For the Monte Carlo integration, two experiments with different sample volumes are conducted. Here, the word "experiment" denotes computing the integrals of (5)–(8) using a set of Monte Carlo samples. First, the experiment with 10^7 samples is performed to generate a reference PCRB to validate the accuracy of the proposed algorithm. Then, the experiment with a reduced sample volume of 1.2×10^4 is conducted. This reduced sample volume Monte Carlo consumes 4.157 s, which is almost the same as that of the 64-component GMM approach after excluding the time used on K-means clustering. The PCRB relative errors of the reduced sample volume Monte Carlo simulation and the GMM approach are compared to verify the computational efficiency of the proposed algorithm.

Fig. 1 depicts the nonlinearity metrics with different configurations. Since (79) is highly nonlinear, the threshold for distinguishing the small and large component covariance cases is set to be 1. The legend "S-Case" denotes the metric under the assumption of small component covariance case. Without refinement for the state and process noise pdfs, the nonlinearity metric frequently exceeds the threshold, which will lead to dramatic accuracy loss. In contrast, the legend "L-Case" denotes the metric corresponding to the aforementioned settings. By refining $p(\mathbf{x}_n)$ and $p(\mathbf{w}_n)$ in each time step, the nonlinearity metric is confined below the threshold almost everywhere except for a few points.

Fig. 2 shows GMM-approximated state pdfs versus Monte Carlo histograms at different times. It is clear

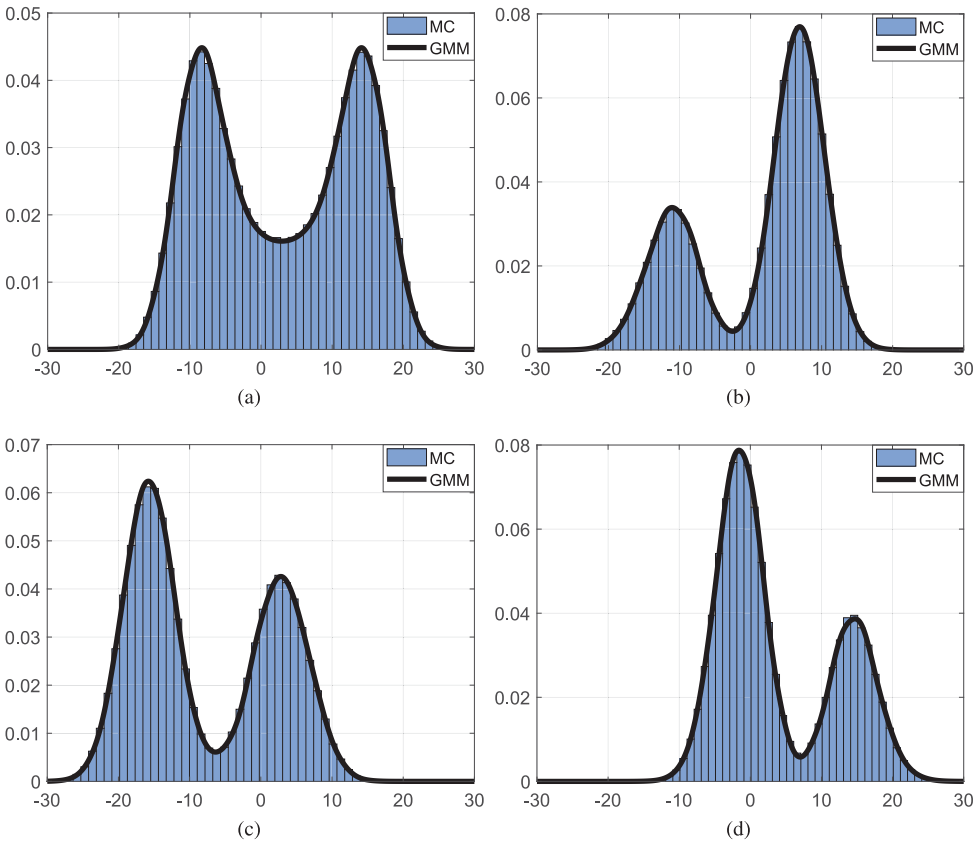


Fig. 2. GMM pdfs versus Monte Carlo samples at various times. (a) $n = 1$. (b) $n = 33$. (c) $n = 66$. (d) $n = 99$.

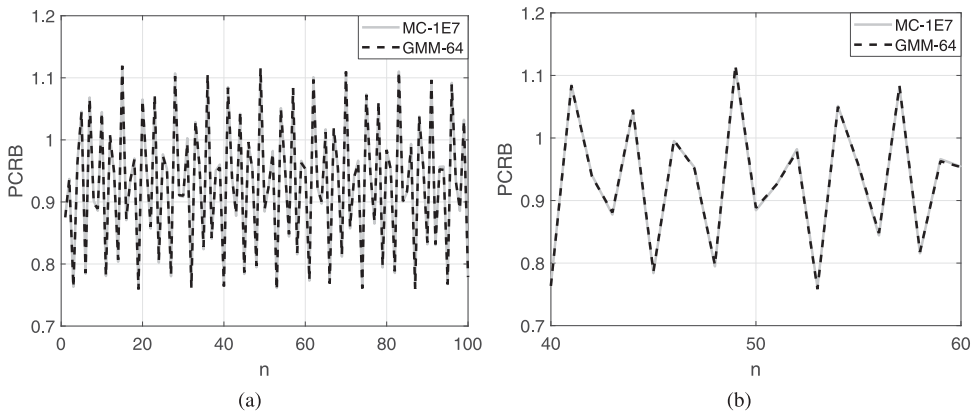


Fig. 3. Comparison of GMM and Monte Carlo approximated PCRBs. (a) Full time steps. (b) Segment zoom in.

that the GMM approximation agrees well with the Monte Carlo samples, and no degradation in accuracy occurs, although the process noise is quite large. This means the extra refine-coarsen step for the uncertainty propagation successfully eliminates the phenomenon of covariance expansion.

Fig. 3 shows the minimum root-mean-square error (RMSE) curves computed by both the GMM and Monte Carlo. Here, we define the state minimum RMSE as the square root of the PCRB. Fig. 3(a) is a local zoom of Fig. 3(b). The two subfigures show that the GMM computed PCRB is consistent with the Monte Carlo experiment.

Fig. 4 shows the computed PCRB relative errors for the GMM and reduced-volume Monte Carlo approaches with respect to the reference PCRB. It is clear that the GMM method is more accurate. Furthermore, the maximum PCRB relative error for this 64-component GMM is no more than 1%, which indicates that the accuracy of the proposed method is high.

Fig. 5 shows the effect of reducing the number of components. A 64-component GMM yields a maximum PCRB relative error of no more than 1%, whereas this quantity for the 32-component GMM is no more than 1.5%. This indicates that the GMM approach is effective in quantifying the state uncertainty.

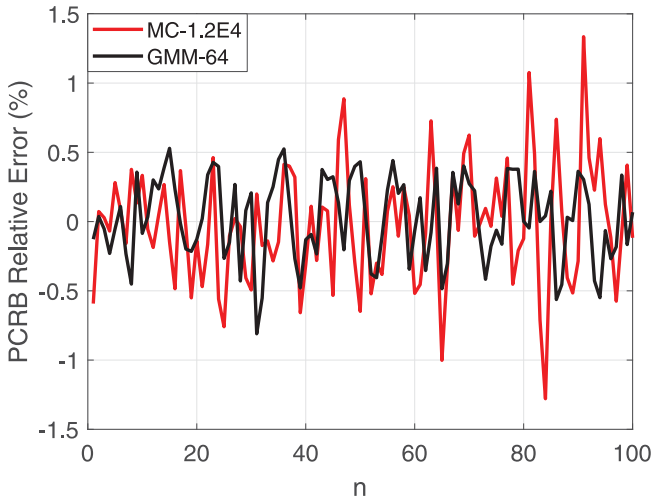


Fig. 4. Comparison of GMM and Monte Carlo PCRB computation accuracies.

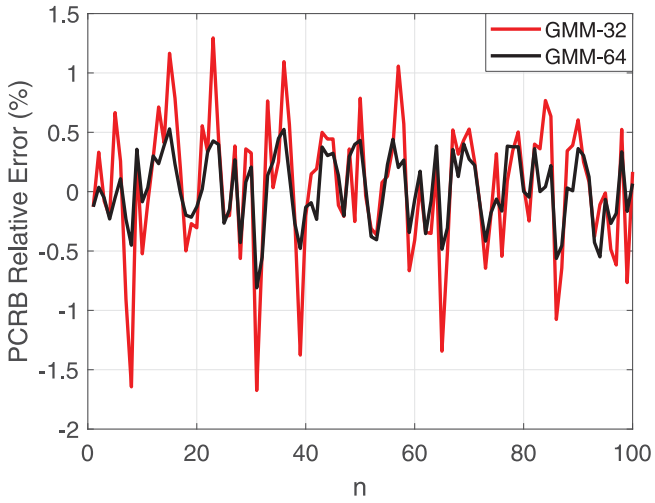


Fig. 5. PCRB computation accuracies for GMMs with different component numbers.

C. Satellite Tracking

Finally, let us consider a more practical problem. In space situational awareness, due to the imperfect dynamic model and the measurement noise introduced by observation sensors, a satellite must be tracked periodically to restrict the unlimited growth of the uncertainty contained in its state. The inter-observation interval is controlled by the sensor tasker and can be very long between two consecutive observation arcs, because the observation resources are always scarce. To save observation time, we want to task a sensor that can best reduce the state uncertainty. The PCRB can serve as a basis for sensor tasking, since it can be computed offline.

The satellite state consists of its position and velocity vectors, namely, $\mathbf{x} = [\mathbf{r}, \mathbf{v}]^T$. The evolution of the discrete state $\mathbf{x}_n$ is described by

$$\mathbf{x}_{n+1} = \phi\left(\mathbf{x}_n;\ t_n, t_{n+1}\right) + \mathbf{w}_n \tag{83}$$

where $\phi(\cdot)$ is the numerical integration operator. In this example, we assume that the satellite follows a Keplerian orbit between two consecutive time instants. Thus, $\mathbf{x}_{n+1}$ can be obtained by numerically integrating the following dynamic equation from t_n to t_{n+1}:

$$\ddot{\mathbf{r}} = -\frac{\mu}{r^3}\mathbf{r} \tag{84}$$

where μ is the earth's gravitational constant, and $r = \|\mathbf{r}\|$. Furthermore, under the two-body dynamics assumption, the time-consuming numerical integration can be replaced with numerical root-finding for the Kepler equation. Although this model is simple, and it may be inadequate for practical applications, it is sufficient for testing the proposed algorithm.

A ground radar system can produce the slant range R, the azimuth angle A, and the elevation angle E by using a single pulse. If the radar works in the coherent integration mode, the high-precision radial velocity $\dot{R}$ can also be obtained by using a pulse train. The radar is assumed to be located in Shaanxi province, China, and its geodetic coordinates are shown in Table I.

TABLE I
Radar Geodetic Coordinates

Latitude (deg)	Longitude (deg)	Altitude (m)
34.490	109.340	0

TABLE II
Satellite Reference State

	X	Y	Z	Unit
Position	2338.826455	-6729. 856951	-0.308879	km
Velocity	-1.037072	-0.351360	7.401615	km/s

TABLE III
Satellite Reference Orbital Elements

Orbital elements	Value
Semi-major axis	7129.326543 km
Eccentricity	0.001353
Inclination	98.414902°
Right ascension of the ascending node	289.163500°
Argument of perigee	61.278714°
True anomaly	298.718775°

Table II shows the satellite reference state at $t = -33.320833$ h, that is, the time interval between the last orbit determination epoch and the first observation of the current arc is approximately 33 h. Table III lists the equivalent orbital elements.

The reference state contains a zero-mean Gaussian uncertainty with a covariance of

$$\mathbf{\Sigma}_0 = \text{diag}\left\{10^{-2}, 10^{-2}, 10^{-2},\ 10^{-8}, 10^{-8}, 10^{-8}\right\} \tag{85}$$

in which the position and velocity are in kilometer and kilometer per second, respectively. The process and measurement noise covariance matrices are

$$\mathbf{Q}_n = \text{diag}\left\{10^{-6}, 10^{-6}, 10^{-6},\ 10^{-12}, 10^{-12}, 10^{-12}\right\} \tag{86}$$

$$\mathbf{R}_n = \text{diag}\left\{4 \times 10^{-6}, 10^{-6}, 10^{-6}, 10^{-12}\right\} \tag{87}$$

in which the unit for angle is rad.

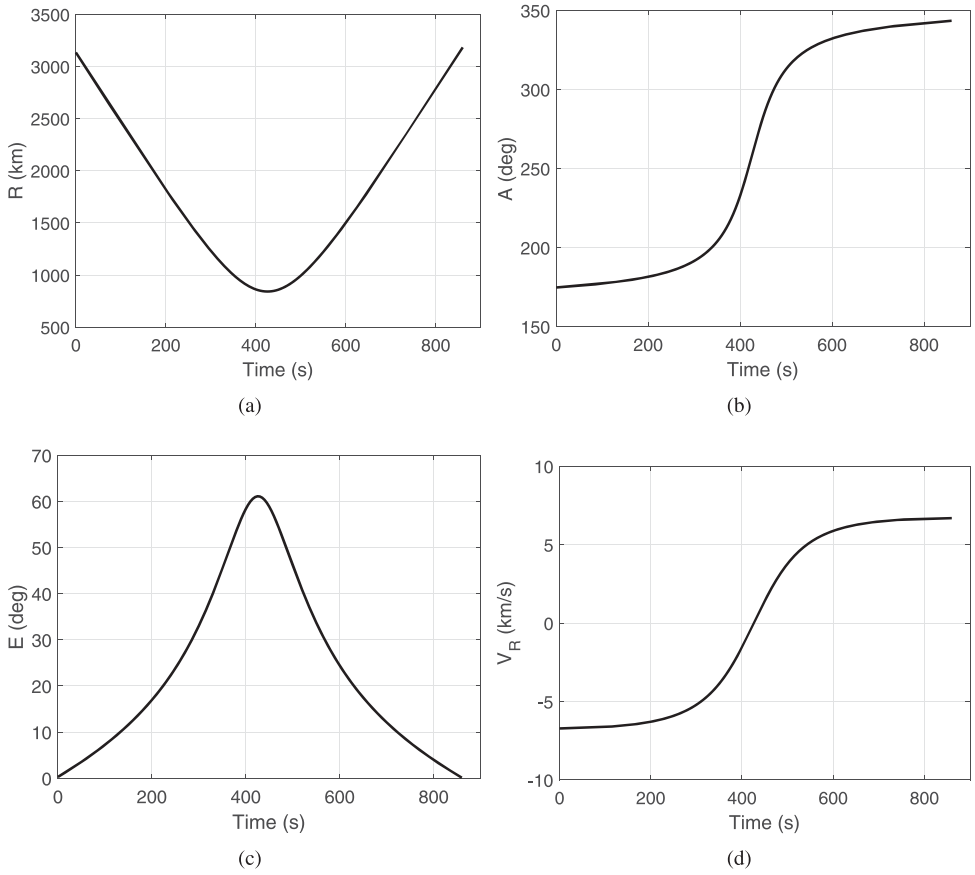


Fig. 6. Radar measurements versus time. (a) Radial range. (b) Azimuth angle. (c) Elevation angle. (d) Radial velocity.

Fig. 6 shows the radar measurements for the current visible arc. These measurements are generated every 5 s.

The PCRB for tracking this satellite is first computed using only R, A, and E. For the GMM approach, since Horwood *et al.* proved that the error in the semimajor axis drives the mean longitude error to grow quadratically, we first linearly convert the initial covariance matrix $\mathbf{\Sigma}_0$ from the Cartesian space to the Keplerian orbit element space; the second splitting library presented in Section III-A with $\sigma = 1/4$ is then applied along the semimajor axis dimension, yielding a 49-component GMM. Finally, the GMM in the orbit element space is converted back to Cartesian space using linear uncertainty mapping. In each iteration, a Smolyak sparse-grid Gauss–Hermite quadrature rule with an accuracy level of two is adopted to compute the high-dimensional integrations. This rule contains 13 quadrature points. Thus, the total number of sigma points is $49 \times 13 = 637$. The overall computation time using the GMM approach is 34.091 s.

As in the 1-D benchmark problem, two Monte Carlo experiments with different sample volumes are conducted. One is the experiment with 10^5 samples to generate the reference PCRB for validating the accuracy of the proposed algorithm, the other is the experiment with 1200 samples for verifying the computational efficiency. The 1200-sample Monte Carlo consumes 35.039 s, which is almost the same as the time consumed by the 49-component GMM.

Fig. 7 presents the uncertainty propagation results in the radial–in track (R-S) plane using the GMM and Monte Carlo methods at $t = 0$, which is the time instant corresponding to the first observation. From this figure, it is obvious that the propagated pdf is no longer Gauss-shaped, since the propagation time is long (approximately 33 h). Furthermore, the propagated GMM shows good consistency with the Monte Carlo samples.

Fig. 8 demonstrates the nonlinearity metrics with different settings. In this example, only the state prior pdf $p(\mathbf{x}_0)$ is refined. This means the PCRB for this example is computed under the assumption of small component covariance. In this figure, the legend "GMM-01" denotes the nonlinearity metric corresponding to a single Gaussian pdf, whereas the legend "GMM-49" denotes the metric corresponding to the GMM with the aforementioned splitting strategy. Due to the relative mild nonlinearity of satellite dynamics, the threshold is assigned with 0.01. From this figure, the nonlinearity metric for the 49-component GMM is less than the threshold all the time, which indicates the validity of the small component covariance assumption. Furthermore, we note that the nonlinearity metric at $t = 0$ is much higher than that at other time instants, this is because the propagation time corresponding to the first measurement (approximately 33 h) is much longer than that corresponding to other measurements (5 s). For a propagation interval of 5 s, the nonlinearity of the satellite motion can be safely ignored, thus we

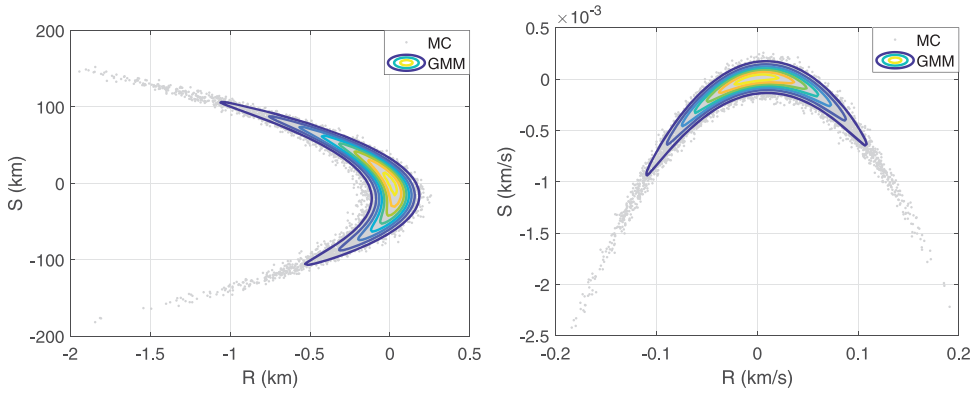


Fig. 7. Satellite state uncertainty in the R-S plane. (a) Position uncertainty. (b) Velocity uncertainty.

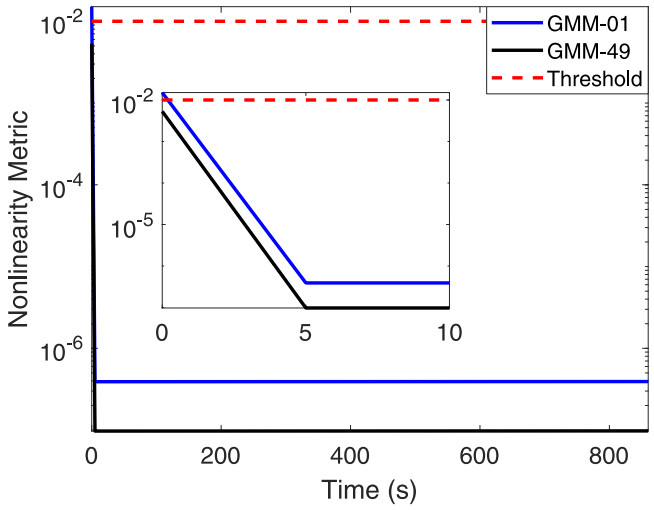


Fig. 8. GMM nonlinearity metrics for GMMs with different component numbers.

only need to split the initial state pdf. This is the principle behind the small component covariance assumption in this example.

Fig. 9 depicts the position and velocity PCRB and their relative error curves. Here, we define the position PCRB (PPCRB) as the square root of the trace of the PCRB positional submatrix. The velocity PCRB (VPCRB) is defined similarly. From Fig. 9(a) and (b), we find that the PCRB obtained by using the GMM approach is very close to the reference PCRB. Furthermore, from Fig. 9(c) and (d), the 49-component GMM is more accurate than the 1200-sample Monte Carlo. Since the computation time for the two approaches is approximately equal, the GMM approach is considered to be more efficient. An interesting phenomenon is that the minimum PPCRB is obtained at approximately $t = 600$ s, rather than the end of this visible arc.

Next, we want to investigate the effect of incorporating the high-precision radial velocity measurement. The aforementioned configurations are preserved except for adding the new measurement $\dot{R}$. The average computation time for the reduced volume Monte Carlo experiment now becomes 33.307 s, whereas the time for the GMM approach becomes 33.977 s.

Fig. 10 shows the PPCRB and VPCRB for estimating the satellite state using R, A, E, and $\dot{R}$. Compared with the results without using $\dot{R}$, both the PPCRB and VPCRB decrease. This clearly indicates that there is additional information contained in $\dot{R}$. Thus, a radar system working in coherent integration mode can not only improve its detection ability for small targets but also generate high-precision radial velocity measurements to prompt the estimated orbit accuracy.

VI. CONCLUSION

In this article, we propose a novel framework to compute the PCRB for the general nonlinear filtering problem with the AWGN. This approach uses the GMM to effectively represent and propagate the uncertainty contained in the state vector and adopts the Gauss–Hermite quadrature rule to compute the high-dimensional integrations. The overall algorithm can be viewed as a deterministic sampling simulation. Three computation examples are presented: the linear model, the 1-D benchmark problem, and the satellite tracking problem. The results show that the proposed method is accurate and more efficient than the plain Monte Carlo integration, especially in the small component noise case. This is because the GMM can approximate a pdf more effectively than a set of Monte Carlo particles.

In the large component covariance case, an extra refine-coarsen step is inserted into the uncertainty propagation stage to eliminate the covariance expansion phenomenon and restrict the exponential growth of computation. In the current work, the K-means clustering algorithm is used to coarsen a GMM, which consumes most of the total computation time. In the future work, a more effective mixture reduction method can be adopted to replace the K-Means clustering to accelerate the execution of the algorithm. Furthermore, this research is based on the AWGN assumption. Although this assumption is widely accepted, it could become a major restriction in some applications. Generalizing this framework to nonlinear, non-Gaussian cases still requires further investigation.

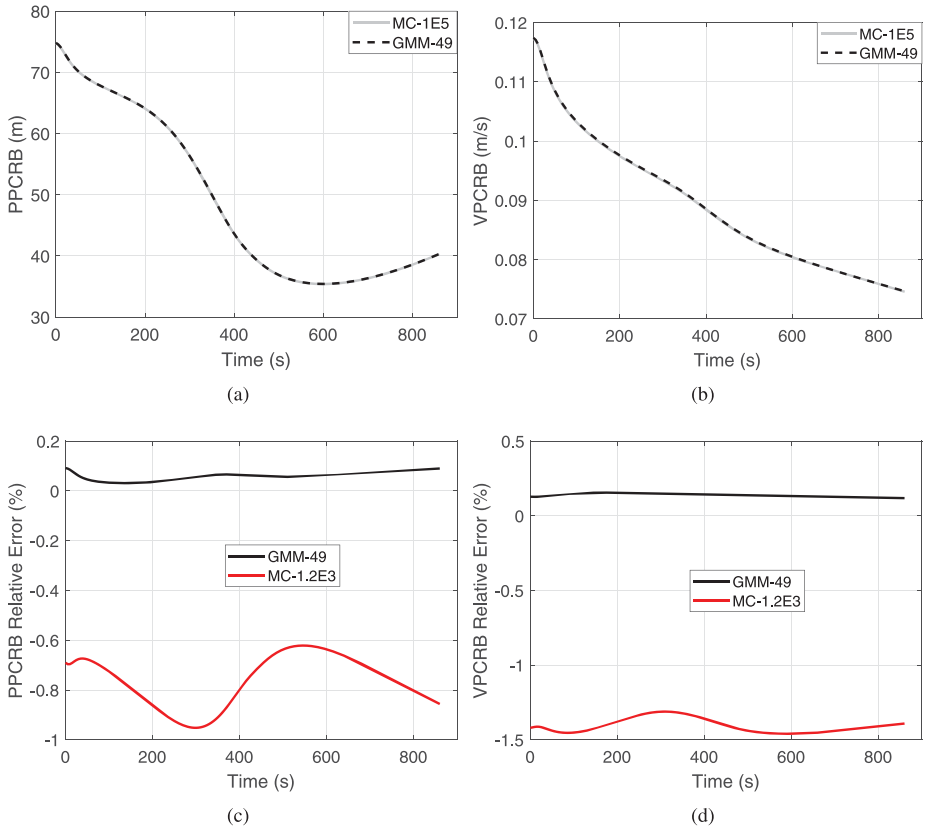


Fig. 9. Comparison of the GMM and Monte Carlo computed PCRBs using R, A, and E. (a) Position PCRB. (b) Velocity PCRB. (c) Position PCRB relative error. (d) Velocity PCRB relative error.

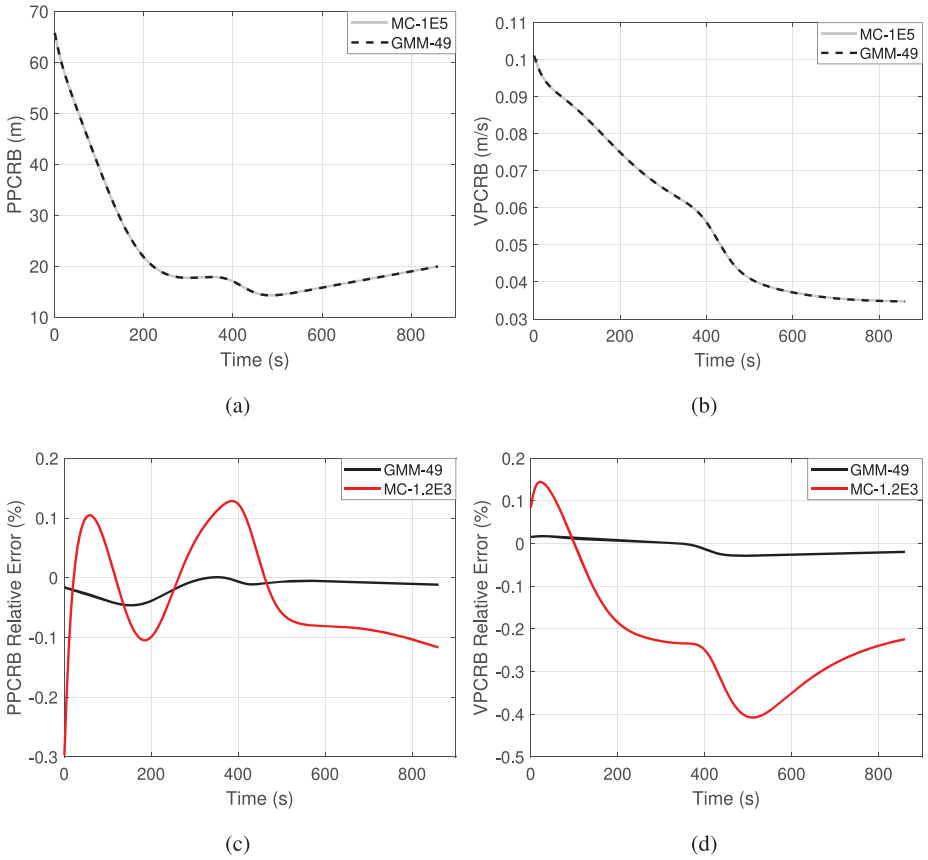


Fig. 10. Comparison of the GMM and Monte Carlo computed PCRBs using R, A, E, and $\dot{R}$. (a) Position PCRB. (b) Velocity PCRB. (c) Position PCRB relative error. (d) Velocity PCRB relative error.

Algorithm 3: GMM K-Means Clustering.

input : $\left\{\left(w_{n,i}^{\mathbf{x}},\ \boldsymbol{\mu}_{n,i}^{\mathbf{x}},\ \boldsymbol{\Sigma}_{n,i}^{\mathbf{x}}\right)\right\}_{i=1}^{\alpha_n^+}$ and K

output: $\left\{\left(w_{n,i}^{\mathbf{x}},\ \boldsymbol{\mu}_{n,i}^{\mathbf{x}},\ \boldsymbol{\Sigma}_{n,i}^{\mathbf{x}}\right)\right\}_{i=1}^{K}$

```
// initialization
k ← GetMaxValueIndex({w_{n,i}^x}_{i=1}^{α_n^+});
newClusterCenterList ← (w_{n,k}^x, μ_{n,k}^x, Σ_{n,k}^x);
for i ← 2 to K do
    for j ← 1 to α_n^+ do
        minDistanceList (j) ←
            GetMinDistance((w_{n,j}^x, μ_{n,j}^x, Σ_{n,j}^x),
            newClusterCenterList);
    end
    k ← GetMaxValueIndex(minDistanceList);
    add (w_{n,k}^x, μ_{n,k}^x, Σ_{n,k}^x) into
        newClusterCenterList;
end

// iteratively clustering
repeat
    oldClusterCenterList ← newClusterCenterList;
    for i ← 1 to α_n^+ do
        for j ← 1 to K do
            distance (i, j) ←
                GetDistance((w_{n,i}^x, μ_{n,i}^x, Σ_{n,i}^x),
                oldClusterCenterList (j));
        end
        k ←
            GetMinValueIndex({distance (i, j)}_{j=1}^{K});

        mark the i-th element belonging to the k-th
            cluster;
    end
    for i ← 1 to K do
        merge all the elements belonging to the i-th
            cluster;
        update the i-th cluster center in
            newClusterCenterList;
    end
until oldClusterCenterList = newClusterCenterList;
```

APPENDIX K-MEANS CLUSTERING ALGORITHM FOR GMM REDUCTION

The pseudocodes for coarsening a GMM are listed in Algorithm 3.

REFERENCES

[1] F. E. Daum
Exact finite-dimensional nonlinear filters
IEEE Trans. Autom. Control, vol. AC-31, no. 7, pp. 616–622, Jul. 1986.

[2] F. Daum
Nonlinear filters: Beyond the Kalman filter
IEEE Aerosp. Electron. Syst. Mag., vol. 20, no. 8, pp. 57–69, Aug. 2005.

[3] B. D. O. Anderson and J. B. Moore
Optimal Filtering. Englewood Cliffs, NJ, USA: Prentice-Hall, 1979, ch. 8, pp. 195–205.

[4] S. Julier, J. Uhlmann, and H. Durrant-Whyte
A new method for the nonlinear transformation of means and covariances in filters and estimators
IEEE Trans. Autom. Control, vol. 45, no. 3, pp. 477–482, Mar. 2000.

[5] S. Julier and J. Uhlmann
Unscented filtering and nonlinear estimation
Proc. IEEE, vol. 92, no. 3, pp. 401–422, Mar. 2004.

[6] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp
A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking
IEEE Trans. Signal Process., vol. 50, no. 2, pp. 174–188, Feb. 2002.

[7] H. L. Van Trees
Detection, Estimation, and Modulation Theory. Part I. Detection, Estimation, and Linear Modulation Theory. Hoboken, NJ, USA: Wiley, 2001, ch. 5, pp. 423–466.

[8] N. Bergman
Posterior Cramér-Rao bounds for sequential estimation
in *Sequential Monte Carlo Methods in Practice*. Berlin, Germany: Springer, 2001, pp. 321–338.

[9] B. Ristic, S. Arulampalam, and N. Gordon
Beyond the Kalman Filter. Particle Filters for Tracking Applications. Norwood, MA, USA: Artech House, 2004, ch. 4, pp. 67–82.

[10] P. H. Leong, S. Arulampalam, T. A. Lamahewa, and T. D. Abhayapala
A Gaussian-sum based cubature Kalman filter for bearings-only tracking
IEEE Trans. Aerosp. Electron. Syst., vol. 49, no. 2, pp. 1161–1176, Apr. 2013.

[11] R. Tharmarasa, T. Kirubarajan, M. L. Hernandez, and A. Sinha
PCRLB-based multisensor array management for multitarget tracking
IEEE Trans. Aerosp. Electron. Syst., vol. 43, no. 2, pp. 539–555, Apr. 2007.

[12] M. Hernandez, T. Kirubarajan, and Y. Bar-Shalom
Multisensor resource deployment using posterior Cramér-Rao bounds
IEEE Trans. Aerosp. Electron. Syst., vol. 40, no. 2, pp. 399–416, Apr. 2004.

[13] Z. Wang, X. Shen, P. Wang, and Y. Zhu
The Cramér-Rao bounds and sensor selection for nonlinear systems with uncertain observations
Sensors, vol. 18, no. 4, 2018, Art. no. 1103.

[14] K. L. Bell, Y. Steinberg, Y. Ephraim, and H. L. Van Trees
Extended Ziv-Zakai lower bound for vector parameter estimation
IEEE Trans. Inf. Theory, vol. 43, no. 2, pp. 624–637, Mar. 1997.

[15] P. Tichavský, C. H. Muravchik, and A. Nehorai
Posterior Cramér-Rao bounds for discrete-time nonlinear filtering
IEEE Trans. Signal Process., vol. 46, no. 5, pp. 1386–1396, May 1998.

[16] A. Farina, B. Ristic, and L. Timmoneri
Cramér-Rao bound for nonlinear filtering with $P_d < 1$ and its application to target tracking
IEEE Trans. Signal Process., vol. 50, no. 8, pp. 1916–1924, Aug. 2002.

[17] M. L. Hernandez, A. Farina, and B. Ristic
PCRLB for tracking in cluttered environments: Measurement sequence conditioning approach
IEEE Trans. Aerosp. Electron. Syst., vol. 42, no. 2, pp. 680–704, Apr. 2006.

[18] C. Hue, J.-P. Le Cadre, and P. Perez
Posterior Cramér-Rao bounds for multi-target tracking
IEEE Trans. Aerosp. Electron. Syst., vol. 42, no. 1, pp. 37–49, Jan. 2006.

[19] T. Bréhard and J.-P. Le Cadre
Closed-form posterior Cramér-Rao bounds for bearings-only tracking
IEEE Trans. Aerosp. Electron. Syst., vol. 42, no. 4, pp. 1198–1223, Oct. 2006.

[20] M. Lei, B. J. van Wyk, and Y. Qi
Online estimation of the approximate posterior Cramér-Rao lower bound for discrete-time nonlinear filtering
IEEE Trans. Aerosp. Electron. Syst., vol. 47, no. 1, pp. 37–57, Jan. 2011.

[21] A. Tulsyan, B. Huang, R. B. Gopaluni, and J. F. Forbes
A particle filter approach to approximate posterior Cramér-Rao lower bound: The case of hidden states
IEEE Trans. Aerosp. Electron. Syst., vol. 49, no. 4, pp. 2478–2495, Oct. 2013.

[22] D. Alspach and H. Sorenson
Nonlinear Bayesian estimation using Gaussian sum approximations
IEEE Trans. Autom. Control, vol. AC-17, no. 4, pp. 439–448, Aug. 1972.

[23] J. T. Horwood and A. B. Poore
Adaptive Gaussian sum filters for space surveillance
IEEE Trans. Autom. Control, vol. 56, no. 8, pp. 1777–1790, Aug. 2011.

[24] J. T. Horwood, N. D. Aragon, and A. B. Poore
Gaussian sum filters for space surveillance: Theory and simulations
J. Guid., Control, Dyn., vol. 34, no. 6, pp. 1839–1851, 2011.

[25] G. Terejanu, P. Singla, T. Singh, and P. Scott
Uncertainty propagation for nonlinear dynamic systems using Gaussian mixture models
J. Guid., Control, Dyn., vol. 31, no. 6, pp. 1623–1633, 2008.

[26] S. J. Julier
The scaled unscented transformation
in *Proc. IEEE Amer. Control Conf.*, 2002, vol. 6, pp. 4555–4559.

[27] E. Wan and R. Van Der Merwe
The unscented Kalman filter for nonlinear estimation
in *Proc. IEEE Adaptive Syst. Signal Process., Commun., Control Symp.*, 2000, pp. 153–158.

[28] R. Van der Merwe
Sigma-point Kalman filters for probabilistic inference in dynamic state-space models
Ph.D. dissertation, Oregon Health Sci. Univ., Beaverton, OR, USA, 2004.

[29] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery
Numerical Recipes. The Art of Scientific Computing, 3rd ed. Cambridge, U.K.: Cambridge Univ. Press, 2007, ch. 4, pp. 155–200.

[30] P. Jäckel
A note on multivariate Gauss-Hermite quadrature
ABN-Amro, London, U.K., 2005.

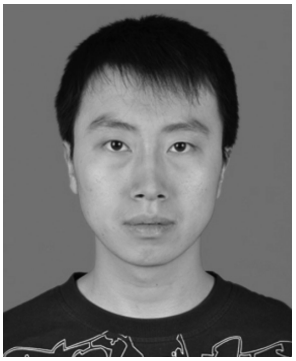
[31] S. A. Smolyak
Quadrature and interpolation formulas for tensor products of certain classes of functions
Dokl. Akad. Nauk SSSR, vol. 148, no. 5, pp. 1042–1045, 1963.

[32] F. Heiss and V. Winschel
Likelihood approximation by numerical integration on sparse grids
J. Econometrics, vol. 144, no. 1, pp. 62–80, 2008.

[33] B. Jia, M. Xin, and Y. Cheng
Sparse-grid quadrature nonlinear filtering
Automatica, vol. 48, no. 2, pp. 327–341, 2012.

[34] K. J. DeMars
Nonlinear orbit uncertainty prediction and rectification for space situational awareness
Ph.D. dissertation, Univ. Texas Austin, Austin, TX, USA, 2010.

[35] W. I. Tam
Tracking filters for radar systems
Master's thesis, Dept. Elect. Comput. Eng., Univ. Toronto, Toronto, ON, Canada, 1997.



Shuo Zhang was born in Baoding, China, in 1992. He received the B.S. degree in communication engineering, in 2014 from the Beijing Institute of Technology, Beijing, China, where he is currently working toward the Ph.D. degree in information and communication engineering.

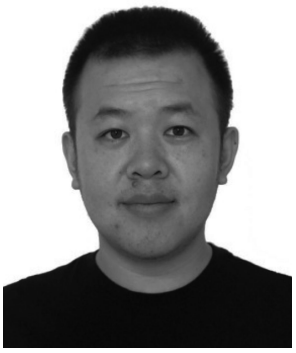
His research interests include space situational awareness and statistical signal processing.

Defeng Chen was born in Longyan, China, in 1983. He received the B.S. and Ph.D. degrees in electronic engineering from the Beijing Institute of Technology, Beijing, China, in 2006 and 2011, respectively.

He is currently an Associate Professor with the School of Information and Electronics, Beijing Institute of Technology. His research interests include radar signal processing and its applications in space target detection.

Tuo Fu was born in Shenyang, China, in 1977. He received the Ph.D. degree in electronic engineering from the Beijing Institute of Technology, Beijing, China, in 2004.

From 2004 to 2006, he served as a Postdoctoral Researcher with the National Mobile Communications Research Laboratory, Southeast University, Nanjing, China. He is currently a Full Professor with the School of Information and Electronics, Beijing Institute of Technology. His research interests include radar system analysis and statistical signal processing.



Huawei Cao received the B.S. degree in electronic science and technology from Yanshan University, Qinhuangdao, China, in 2009, and the Ph.D. degree in electronic engineering from the Beijing Institute of Technology, Beijing, China, in 2016.

He is currently a Radar System Engineer with the School of Information and Electronics, Beijing Institute of Technology. His research interests include radar system design, weak target detection, and multitarget tracking.