aerospace

MDPI

Article

Sparse-Grid Gaussian Kernel Quadrature Kalman Filter for Nonlinear State Estimation

Yijie Zhao, Hao Wu *, Guoxu Zeng, Minbo Yang, Chaoqi Li and Sahan Rathnayake

Information and Navigation College, Air Force Engineering University, Xi'an 710077, China; zhaoyijie030426@163.com (Y.Z.); zgx1997416@163.com (G.Z.); ymb_study@163.com (M.Y.); lichaoqistudy@126.com (C.L.); sahan.ms@gmail.com (S.R.)

* Correspondence: wuhaostudy@163.com; Tel.: +86-029-84791323

Abstract

Nonlinear state estimation plays an important role in aerospace sensing applications, where estimation accuracy must be balanced against computational efficiency. In this paper, a sparse-grid Gaussian kernel quadrature Kalman filter (SGKQKF) is proposed for discrete-time nonlinear state estimation by combining Gaussian kernel quadrature (GKQ) weighting with a Smolyak sparse-grid construction. The univariate GKQ rule is constructed on scaled Gauss–Hermite nodes through a truncated Mercer eigendecomposition of the Gaussian kernel and is then extended to multivariate cases via the Smolyak construction to alleviate the curse of dimensionality associated with tensor-product rules. The proposed method is positioned within the established sparse-grid filtering framework, with the specific contribution of integrating kernel-adapted quadrature weights into sparse-grid structures for discrete-time nonlinear Gaussian filtering. For fixed nodes, the exact kernel-quadrature weights minimize the worst-case integration error in the reproducing kernel Hilbert space (RKHS) induced by the Gaussian kernel, whereas the closed-form weights used in the implementation are interpreted as a Mercer-based practical approximation to this exact rule, with the approximation error characterized through the Mercer spectral-tail expression of the Gaussian kernel. For sparse grids, where a closed-form RKHS optimality result is not available, numerical maximum mean discrepancy (MMD) evaluations are presented as empirical diagnostics in the tested configurations. Numerical experiments demonstrate that the proposed filter achieves a favorable accuracy–efficiency trade-off compared with conventional deterministic Gaussian filters.

Keywords: nonlinear state estimation; sparse-grid quadrature; Gaussian kernel quadrature; bearings-only tracking; reproducing kernel Hilbert space

Check for updates

Academic Editor: Dario Modenini

Received: 30 March 2026
Revised: 5 May 2026
Accepted: 8 May 2026
Published: 15 May 2026

Copyright: © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license.

1. Introduction

Nonlinear state estimation is a fundamental problem in sensor fusion, navigation, and bearings-only tracking, with important applications in aerospace systems [1–5]. In such problems, the process and measurement models are often strongly nonlinear, while practical implementations are constrained by real-time computation and onboard processing capability. The extended Kalman filter (EKF) has long been one of the most widely used approaches because of its simple structure and modest computational cost. However, its first-order linearization may introduce non-negligible errors in strongly nonlinear regimes commonly encountered in practice. On the other hand, the sequential Monte-Carlo-based approaches, such as the particle filter (PF), can provide better accuracy, but their computational burden is often too high for real-time applications.

For this reason, deterministic sampling filters (DSFs), which approximate Gaussian-weighted integrals by carefully selected deterministic points and weights, have attracted considerable attention. Representative methods include the unscented Kalman filter (UKF) [6], the cubature Kalman filter (CKF) [7], and the Gauss–Hermite quadrature Kalman filter (GHQKF) [8]. These methods typically provide more accurate Gaussian integral approximations than the EKF in strongly nonlinear settings, while remaining much more computationally efficient than sequential Monte-Carlo approaches [9,10].

Among these filters, the GHQKF is generally more accurate and stable than the UKF and CKF because it relies on Gauss–Hermite quadrature to compute Gaussian-weighted integrals. However, its tensor-product construction causes the number of quadrature points to grow exponentially with the state dimension, which makes it computationally expensive in higher-dimensional estimation problems. Sparse-grid rules have therefore been introduced to mitigate this limitation, leading to sparse-grid Gaussian filters with improved computational scalability [11]. However, classical GHQ, Sparse-grid quadrature filter and sparse-GHQ-based filters still employ polynomial-based quadrature weights determined by moment-matching conditions, which limits their flexibility in adapting the quadrature rule to the structure of the underlying integrand.

A different line of research has emerged from the kernel quadrature and probabilistic numerical integration literature, where quadrature weights are derived by minimizing the worst-case integration error in a reproducing kernel Hilbert space (RKHS) [12–14]. For the Gaussian kernel, Karvonen and Särkkä [12] showed that Gaussian kernel quadrature (GKQ) can be constructed on scaled Gauss–Hermite nodes through the Mercer eigendecomposition, yielding kernel-adapted weights with attractive theoretical properties. Briol et al. [14] further established the role of probabilistic integration in statistical computation, providing a broader theoretical foundation for kernel-based quadrature. In nonlinear filtering, Wang et al. [15] and Naik et al. [16] developed a Gaussian kernel quadrature Kalman filter (GKQKF) in the standard tensor-product setting, and Wu et al. [17] further extended the GKQKF to the continuous–discrete system for GNSS interference source tracking.

However, the existing literature applies the GKQ rule mainly in the univariate or tensor-product setting, and its sparse-grid extension for higher-dimensional nonlinear estimation has not been fully developed. In contrast, existing sparse-grid filters, such as the sparse-grid Gauss–Hermite quadrature filter (SGHQKF), alleviate the dimensional growth of tensor-product quadrature, but they still rely on quadrature weights inherited from polynomial-exactness-based univariate rules rather than kernel-adapted RKHS weighting. Therefore, although RKHS optimality is available for fixed nodes and tensor-product GKQ constructions, the behavior of GKQ-type weighting within a Smolyak sparse-grid construction has not been systematically examined in the context of nonlinear Kalman filtering.

To address this gap, this paper proposes a sparse-grid Gaussian kernel quadrature Kalman filter (SGKQKF) for nonlinear state estimation. Starting from a univariate GKQ rule constructed on scaled Gauss–Hermite nodes, the proposed method introduces a Smolyak sparse-grid extension to obtain a multivariate quadrature rule with substantially reduced computational complexity relative to tensor-product GKQ filtering. In this way, the method combines kernel-adapted quadrature weighting with the computational advantage of sparse-grid sampling. In addition, the relationship between the proposed SGKQKF and existing filters, including the GHQKF, the sparse-grid GHQKF, and the UKF under specific parameter settings, is discussed to clarify the scope of the proposed formulation.

The main contributions of this paper are summarized as follows:

(1) A sparse-grid Gaussian kernel quadrature filtering scheme is developed for discrete-time nonlinear state estimation. By combining GKQ-type kernel-adapted weighting with a Smolyak sparse-grid construction, the proposed SGKQKF extends existing GKQ-based

filtering beyond the univariate and full tensor-product settings, while retaining the computational advantage of sparse-grid sampling in higher-dimensional problems.

(2) The RKHS interpretation and the connections with existing deterministic Gaussian filters are clarified. Exact fixed-node kernel-quadrature optimality is distinguished from the closed-form GKQ implementation, while numerical maximum mean discrepancy (MMD) comparisons are used to assess the sparse-grid RKHS discrepancy in the tested configurations. The relationships with GHQKF, SGHQKF, and UKF are discussed under specific parameter settings.

The remainder of this paper is organized as follows. Section 2 reviews the deterministic Gaussian filtering framework and the associated Gaussian-weighted integral approximation problem. Section 3 presents the Gaussian kernel quadrature rule, the sparse-grid extension, and the resulting SGKQKF, followed by the analysis of its relationships with existing filters and its RKHS interpretation. Section 4 reports numerical experiments, including a bearings-only tracking problem with angle-only measurements. Section 5 concludes the paper and discusses future research directions.

2. The Gaussian Nonlinear Filter with Deterministically Sampled Rules

2.1. System Model and Gaussian Deterministically Sampled Filtering Framework

Consider a general nonlinear discrete-time system with additive process noise and measurement noise:

$$\boldsymbol{x}_k = f(\boldsymbol{x}_{k-1}) + \boldsymbol{v}_{k-1} \tag{1}$$

$$\boldsymbol{z}_k = h(\boldsymbol{x}_k) + \boldsymbol{e}_k \tag{2}$$

where $\boldsymbol{x}_k \in \mathbb{R}^{n_x}$ is the state vector, and $\boldsymbol{z}_k \in \mathbb{R}^{n_z}$ is the measurement vector, with n_x and n_z not necessarily equal. The process noise $v_{k-1} \sim N(0, Q_{k-1})$ and the measurement noise $e_k \sim N(0, R_k)$ are assumed to be mutually independent zero-mean Gaussian white-noise sequences with known covariance matrices. The functions $f(\cdot)$ and $h(\cdot)$ denoted the state-transition function and the measurement function, respectively.

Under the additive-noise Gaussian filtering framework, the posterior distribution is approximated by a Gaussian distribution characterized by its mean and covariance. The Kalman-type state observer used in Equations (4)–(17) is therefore a standard deterministic-sampling Gaussian filter, in which nonlinear prediction and update moments are evaluated by deterministic quadrature rules. The focus of this paper is to improve this quadrature approximation by introducing the proposed sparse-grid Gaussian kernel quadrature rule.

The key to the Gaussian filter is how to compute weighted integrals whose integrands are a product of a nonlinear function and a Gaussian density function. They can be approximated by the deterministically sampled (DS) rules. For a generic integrand $g(\cdot)$, the Gaussian-weighted integral with respect to the standard Gaussian distribution $N(x; 0, I)$ can be written as

$$I(g) = \int_{\mathbb{R}^{n_x}} g(\boldsymbol{x}) N(\boldsymbol{x}; \boldsymbol{0}, \boldsymbol{I}) d\boldsymbol{x} \approx \sum_{j=1}^{n_p} w_j g(\xi_j) \tag{3}$$

where $\boldsymbol{\xi} = \left[\xi_1, \cdots, \xi_{n_p}\right]^T$ and $\boldsymbol{w} = \left[w_1, \ldots, w_{n_p}\right]^T$ are the n_p DS points and corresponding weights. $\boldsymbol{I}$ is the identity matrix.

The Gaussian deterministically sampled filtering framework includes recursive prediction and update procedures [18]. At the prediction step, assume that the posterior density function $p(\boldsymbol{x}_{k-1}|\boldsymbol{z}_{1:k-1}) = N\left(\hat{\boldsymbol{x}}_{k-1|k-1}, \boldsymbol{P}_{k-1|k-1}\right)$ is known, where $\hat{\boldsymbol{x}}_{k-1|k-1}$ is the estimated state and $\boldsymbol{P}_{k-1|k-1}$ is the estimated covariance at time index $k-1$. $\boldsymbol{P}_{k-1|k-1}$ can be factorized as

$$\boldsymbol{P}_{k-1|k-1} = \boldsymbol{S}_{k-1|k-1} \boldsymbol{S}_{k-1|k-1}^T \tag{4}$$

Evaluate the deterministically sampled points $\boldsymbol{X}_{j,k-1|k-1}$ $(j = 1, 2, \cdots, n_p)$ as

$$\boldsymbol{X}_{j,k|k-1} = \boldsymbol{S}_{k-1|k-1}\boldsymbol{\xi}_j + \hat{\boldsymbol{x}}_{k-1|k-1} \tag{5}$$

The propagated sampled points $\boldsymbol{X}^*_{j,k|k-1}$ $(j = 1, 2, \cdots, n_p)$ are:

$$\boldsymbol{X}^*_{j,k|k-1} = f(\boldsymbol{X}_{j,k|k-1}) \tag{6}$$

Accordingly, the predicted state $\hat{\boldsymbol{x}}_{k|k-1}$ and associated covariance $\boldsymbol{P}_{k|k-1}$ can be obtained by

$$\hat{\boldsymbol{x}}_{k|k-1} = \sum_{j=1}^{n_p} w_j \boldsymbol{X}^*_{j,k|k-1} \tag{7}$$

$$\boldsymbol{P}_{k|k-1} = \sum_{j=1}^{n_p} w_j \boldsymbol{X}^*_{j,k|k-1}\boldsymbol{X}^{*\mathrm{T}}_{j,k|k-1} - \hat{\boldsymbol{x}}_{k|k-1}\hat{\boldsymbol{x}}^T_{k|k-1} + \boldsymbol{Q} \tag{8}$$

At the measurement update step, $\boldsymbol{P}_{k|k-1}$ can be factorized as

$$\boldsymbol{P}_{k|k-1} = \boldsymbol{S}_{k|k-1}\boldsymbol{S}^{\mathrm{T}}_{k|k-1} \tag{9}$$

Evaluate the deterministically sampled points $\boldsymbol{X}_{j,k|k-1}$ $(j = 1, 2, \cdots, n_p)$ as

$$\boldsymbol{X}_{j,k|k-1} = \boldsymbol{S}_{k|k-1}\boldsymbol{\xi}_j + \hat{\boldsymbol{x}}_{k|k-1} \tag{10}$$

The propagated sampled points $\boldsymbol{Z}^*_{j,k|k-1}$ $(j = 1, 2, \cdots, n_p)$ are:

$$\boldsymbol{Z}^*_{j,k|k-1} = h\left(\boldsymbol{X}_{j,k|k-1}\right) \tag{11}$$

The predicted measurement $\hat{\boldsymbol{z}}_{k|k-1}$ and the associated covariance $\boldsymbol{P}_{zz,k|k-1}$ are given by

$$\hat{\boldsymbol{z}}_{k|k-1} = \sum_{j=1}^{n_p} w_j \boldsymbol{Z}^*_{j,k|k-1} \tag{12}$$

$$\boldsymbol{P}_{zz,k|k-1} = \sum_{j=1}^{n_p} w_j \boldsymbol{Z}^*_{j,k|k-1}\boldsymbol{Z}^{*\mathrm{T}}_{j,k|k-1} - \hat{\boldsymbol{z}}_{k|k-1}\hat{\boldsymbol{z}}^T_{k|k-1} + \boldsymbol{R}_k \tag{13}$$

and the cross-covariance is

$$\boldsymbol{P}_{xz,k|k-1} = \sum_{j=1}^{n_p} w_j \boldsymbol{X}_{j,k|k-1}\boldsymbol{Z}^{*\mathrm{T}}_{j,k|k-1} - \hat{\boldsymbol{x}}_{k|k-1}\hat{\boldsymbol{z}}^T_{k|k-1} \tag{14}$$

Therefore, the updated estimated state and corresponding error covariance are represented by

$$\hat{\boldsymbol{x}}_{k|k} = \hat{\boldsymbol{x}}_{k|k-1} + \boldsymbol{K}_k(\boldsymbol{z}_k - \hat{\boldsymbol{z}}_{k|k-1}) \tag{15}$$

$$\boldsymbol{P}_{k|k} = \boldsymbol{P}_{k|k-1} - \boldsymbol{K}_k\boldsymbol{P}_{zz,k|k-1}\boldsymbol{K}^T_k \tag{16}$$

where the Kalman gain $\boldsymbol{K}_k$ is

$$\boldsymbol{K}_k = \boldsymbol{P}_{xz,k|k-1}\boldsymbol{P}^{-1}_{zz,k|k-1} \tag{17}$$

2.2. The Deterministically Sampled Rules

The points selection method for deterministically sampled filtering in (3), such as the unscented transformation and cubature rule, can be seen in [18–21]. In this paper, the Gaussian–Hermite quadrature rule is briefly introduced.

When $n_x = 1$, in order to generate the univariate GHQ points, a symmetric and diagonal matrix $\boldsymbol{J}$ is built, where the elements of the diagonal are 0 and $\boldsymbol{J}_{i,i+1} = \sqrt{i/2}$ $(i = 1, 2, \cdots, n_s - 1)$. Calculate the eigenvalues $\boldsymbol{\lambda}$ of the matrix $\boldsymbol{J}$, and the quadrature point $\xi_j^{GH} = \sqrt{2}\lambda_j$, where λ_j is the j^{th} eigenvalue of $\boldsymbol{J}$. Then calculate the normalized eigenvectors $\boldsymbol{v}$ of $\boldsymbol{J}$, and the weights $w_j^{GH} = \left[\left(\boldsymbol{v}_j\right)_1\right]^2$, where $\left(\boldsymbol{v}_j\right)_1$ is the first element of the j^{th} normalized eigenvector. The univariate GHQ rule above can be exact up to the $(2n_s - 1)^{th}$ order of polynomials. In this case, $n_p = n_s$. Particularly, if $n_p = n_s = 1$, then $\xi_1^{GH} = 0$ and $w_1^{GH} = 1$.

When $n_x > 1$, the quadrature points can be obtained by the direct tensor product of the univariate GHQ points. The number of the quadrature points is $n_p = n_s^{n_x}$. For example, if $n_s = 2$ and $n_s = 3$, the univariate GHQ points are ξ_1^{GH} and ξ_2^{GH} with the weights w_1^{GH} and w_2^{GH}. The number of the quadrature points is $n_p = 2^3 = 8$, such that:

$$\begin{aligned}\widetilde{\xi}^{GH} &= \left(\xi_1^{GH}, \cdots, \xi_8^{GH}\right) \\ &= \begin{pmatrix} \left(\xi_1^{GH}, \xi_1^{GH}, \xi_1^{GH}\right), \left(\xi_1^{GH}, \xi_1^{GH}, \xi_2^{GH}\right), \\ \left(\xi_1^{GH}, \xi_2^{GH}, \xi_1^{GH}\right), \left(\xi_1^{GH}, \xi_2^{GH}, \xi_2^{GH}\right) \\ \left(\xi_2^{GH}, \xi_1^{GH}, \xi_1^{GH}\right), \left(\xi_2^{GH}, \xi_1^{GH}, \xi_2^{GH}\right), \\ \left(\xi_2^{GH}, \xi_2^{GH}, \xi_1^{GH}\right), \left(\xi_2^{GH}, \xi_2^{GH}, \xi_2^{GH}\right) \end{pmatrix}\end{aligned}$$

and eight corresponding weights are:

$$\begin{aligned}\widetilde{\boldsymbol{w}}^{GH} &= \left(\widetilde{w}_1^{GH}, \cdots, \widetilde{w}_8^{GH}\right) \\ &= \begin{pmatrix} w_1^{GH} w_1^{GH} w_1^{GH}, w_1^{GH} w_1^{GH} w_2^{GH}, \\ w_1^{GH} w_2^{GH} w_1^{GH}, w_1^{GH} w_2^{GH} w_2^{GH}, \\ w_2^{GH} w_1^{GH} w_1^{GH}, w_2^{GH} w_1^{GH} w_2^{GH}, \\ w_2^{GH} w_2^{GH} w_1^{GH}, w_2^{GH} w_2^{GH} w_2^{GH} \end{pmatrix}\end{aligned}$$

Introducing the multivariate GHQ points and weights into the Gaussian filtering framework involved in (5) to (14), the GHQKF can be obtained.

3. Gaussian Kalman Filter with Sparse Gaussian Kernel Quadrature Rule

As discussed in Section 1, the GHQKF achieves high accuracy but at an exponential computational cost. In this section, we construct the GKQ rule to improve the quadrature weights and then introduce the sparse-grid technique to reduce computational complexity.

3.1. Gaussian Kernel Quadrature Rule

Defining the Gaussian kernel [12]:

$$k(x, y) = \exp\left(-\frac{(x - y)^2}{2\sigma^2}\right) \tag{18}$$

where the scaled factor $\sigma > 0$. For a univariate generic integrand $g(\cdot)$, the Gaussian integral approximated by the GKQ rule is expressed as

$$\begin{aligned} I(g) &= \int_{\mathbb{R}} g(x) N(x; 0, 1) dx \\ &\approx \sum_{j=1}^{n_s} w_j^{GK} g\left(\xi_j^{GK}\right) \end{aligned} \tag{19}$$

where $\xi^{GK} = \left[\xi_1^{GK}, \xi_2^{GK}, \cdots, \xi_{n_s}^{GK}\right]^{\mathrm{T}}$ and $\boldsymbol{w}^{GK} = \left[w_1^{GK}, w_2^{GK}, \cdots, w_{n_s}^{GK}\right]^{\mathrm{T}}$ are the $\boldsymbol{w}^{GK} = \left[w_1^{GK}, w_2^{GK}, \cdots, w_{n_s}^{GK}\right]^{\mathrm{T}}$ univariate Gaussian kernel points and corresponding weights, respectively.

The weights vector $\boldsymbol{w}^{GK} = \left[w_1^{GK}, w_2^{GK}, \cdots, w_{n_s}^{GK}\right]^{\mathrm{T}}$ can be described as the linear system [12]:

$$\boldsymbol{K}\boldsymbol{w}^{GK} = \boldsymbol{k}_{\mu} \tag{20}$$

where $[\boldsymbol{K}]_{i,j} = k(x_i, x_j)$, $\left[\boldsymbol{k}_{\mu}\right]_i = \int_{\mathbb{R}} k(x_i, x) d\mu(x)$, and μ is the Gaussian measure.

According to Mercer's theorem [22], the Gaussian kernel can be rewritten as

$$k(x, y) = \sum_{j=0}^{\infty} \lambda_j \varphi_j(x) \varphi_j(y) \tag{21}$$

where λ_j are the positive and non-increasing eigenvalues and φ_j are the L_2-orthonormal eigenfunctions. Assume that μ_α is the Gaussian probability measure, such that $d\mu_\alpha(x) = \alpha e^{-\alpha^2 x^2} dx/\sqrt{\pi}$ with variance $1/(2\alpha^2)$, where α is related to the global scale. Considering the convergence property and performance of the nodes, $\alpha = 1/\sqrt{2}$ is chosen here [12].

Defining the constants $\varepsilon = 1/(2\sigma)$,$\beta = \left(1 + (2\varepsilon/\alpha)^2\right)^{1/4}$, and $\delta^2 = \alpha^2(\beta^2 - 1)/2$, the eigenvalues λ_n is

$$\lambda_n = \sqrt{\frac{\alpha^2}{\alpha^2 + \delta^2 + \varepsilon^2}} \left(\frac{\alpha^2}{\alpha^2 + \delta^2 + \varepsilon^2}\right)^{n-1} \tag{22}$$

where ε is related to the local scale of the problem and gives the scale of the kernel, which in turn defines the underlying reproducing kernel Hilbert space along with a length scale reflected in its norm. The auxiliary parameter δ reflects the length scale of the eigenfunctions. Thus, the corresponding eigenfunctions $\boldsymbol{\varphi}_n$ can be obtained by

$$\boldsymbol{\varphi}_n(x) = \sqrt{\frac{\beta}{n!}} e^{-\delta^2 x^2} H_{n-1}\left(\sqrt{2}\alpha\beta x\right) \tag{23}$$

where $n = 1, 2, \ldots$, and $H_n(x)$ is the standard Hermite polynomials defined by their Rodrigues formula:

$$H_n(x) = (-1)^n e^{\frac{x^2}{2}} \frac{d^n}{dx^n} e^{-\frac{x^2}{2}} \tag{24}$$

Using the eigendecomposition of the kernel function $\boldsymbol{K}$, it can be rewritten as

$$\begin{aligned} \boldsymbol{K} &= \begin{pmatrix} K(x_1, x_1) & \cdots & K(x_1, x_{n_s}) \\ \vdots & \ddots & \vdots \\ K(x_1, x_{n_s}) & \cdots & K(x_{n_s}, x_{n_s}) \end{pmatrix} \\ &= \begin{pmatrix} \varphi_1(x_1) & \cdots \\ \vdots & \ddots \\ \varphi_1(x_{n_s}) & \cdots \end{pmatrix} \begin{pmatrix} \lambda_1 & \\ & \ddots \end{pmatrix} \begin{pmatrix} \varphi_1(x_1) & \cdots & \varphi_1(x_{n_s}) \\ \vdots & \ddots & \vdots \end{pmatrix} \end{aligned} \tag{25}$$

The infinite matrix cannot be directly computed. So a truncation n_m is chosen to approximate it. Then we have [23]

$$\begin{aligned} \boldsymbol{K} \approx \begin{pmatrix} \varphi_1(x_1) & \cdots & \varphi_{n_m}(x_1) \\ \vdots & & \vdots \\ \varphi_1(x_{n_s}) & \cdots & \varphi_{n_m}(x_{n_m}) \end{pmatrix} \begin{pmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_{n_m} \end{pmatrix} \cdot \\ \begin{pmatrix} \varphi_1(x_1) & \cdots & \varphi_1(x_{n_s}) \\ \vdots & & \vdots \\ \varphi_{n_m}(x_1) & \cdots & \varphi_{n_m}(x_{n_s}) \end{pmatrix} = \boldsymbol{\phi}\Lambda\boldsymbol{\phi}^T \end{aligned} \tag{26}$$

To avoid the ill-conditioning associated with radial basis interpolation as $\varepsilon \to 0$, $n_m \geq n_s$ is required. Here, we assume that $n_m = n_s$, $\boldsymbol{\phi}$ is turned into a square matrix. According to (23), $\boldsymbol{\phi} = \sqrt{\beta}\boldsymbol{E}^{-1}\boldsymbol{V}$ is obtained, where $\boldsymbol{E}$ is a diagonal matrix $[\boldsymbol{E}]_{ii} = e^{\delta^2 x_i^2}$, and $\boldsymbol{V}$ is the Vandermonde matrix:

$$[\boldsymbol{V}]_{ij} = \frac{1}{\sqrt{(j-1)!}} H_{j-1}\left(\sqrt{2}\alpha\beta x_i\right) \tag{27}$$

In order to normalize the Hermite polynomials of $\boldsymbol{V}$, the univariate GKQ points can be defined as

$$\xi_j^{GK} = \frac{1}{\sqrt{2}\alpha\beta}\xi_j^{GH} \tag{28}$$

where ξ_j^{GH} are the GHQ points. Now $\boldsymbol{V}$ can be rewritten as $[\boldsymbol{V}]_{ij} = 1/\sqrt{(j-1)!}H_{j-1}(\xi_i^{GH})$, which means univariate GKQ points can be obtained from scaled univariate GHQ points.

$\boldsymbol{V}\boldsymbol{V}^T$ is diagonal and the inverse of the diagonal elements are the corresponding weights of the unique GHQ points such that $\boldsymbol{V}\boldsymbol{V}^T = \left(\boldsymbol{W}^{GH}\right)^{-1}$, where $\boldsymbol{W}^{GH} = diag(w_1^{GH}, \cdots, w_{n_s}^{GH})$. Then, $\boldsymbol{V}^{-T} = \boldsymbol{W}^{GH}\boldsymbol{V}$. According to (20), (26), and the above discussions, the corresponding weights of the GKQ can be obtained:

$$\begin{aligned}
&\boldsymbol{w}^{GK} = \boldsymbol{K}^{-1}\boldsymbol{k}_\mu = \left(\boldsymbol{\phi}\Lambda\boldsymbol{\phi}^{\mathrm{T}}\right)^{-1}\boldsymbol{\phi}\Lambda\boldsymbol{\varphi}_\mu = \boldsymbol{\phi}^{-\mathrm{T}}\boldsymbol{\varphi}_\mu \\
&= \frac{1}{\sqrt{\beta}}\boldsymbol{E}\boldsymbol{V}^{-\mathrm{T}}\boldsymbol{\varphi}_\mu = \frac{1}{\sqrt{\beta}}\boldsymbol{E}\boldsymbol{W}^{GH}\boldsymbol{V}\boldsymbol{\varphi}_\mu \\
&= \frac{\mathrm{diag}\left(w_j^{GH}e^{\delta^2(\xi_j^{GK})^2}\right)}{\sqrt{\beta}}
\begin{bmatrix}
\frac{H_0\left(\xi_1^{GH}\right)}{\sqrt{(1-1)!}} & \cdots & \frac{H_0\left(\xi_1^{GH}\right)}{\sqrt{(n_s-1)!}} \\
\vdots & & \vdots \\
\frac{H_{n_s}\left(\xi_{n_s-1}^{GH}\right)}{\sqrt{(1-1)!}} & \cdots & \frac{H_{n_s}\left(\xi_{n_s}^{GH}\right)}{\sqrt{(n_s-1)!}}
\end{bmatrix}
\begin{bmatrix}
\mu\left(\varphi_0^\alpha\right) \\ \vdots \\ \mu\left(\varphi_{n_s-1}^\alpha\right)
\end{bmatrix}
\end{aligned} \tag{29}$$

Therefore,

$$\begin{aligned}
w_j^{GK} &= \frac{1}{\sqrt{\beta}}E_{jj}W_{jj}^{GH}\sum_{i=0}^{n_s-1}\left(V_{j,i+1}\mu\left(\varphi_i^\alpha\right)\right) \\
&= \frac{1}{\sqrt{\beta}}e^{\delta^2(\xi_j^{GK})^2}w_j^{GH}\sum_{i=0}^{n_s-1}\left(\frac{1}{\sqrt{i!}}H_i\left(\xi_j^{GH}\right)\mu\left(\varphi_i^\alpha\right)\right)
\end{aligned} \tag{30}$$

where the eigenfunctions of the Gaussian kernel satisfy [12]:

$$\mu\left(\varphi_{2m+1}^\alpha\right) = 0, \;\; \mu\left(\varphi_{2m}^\alpha\right) = \sqrt{\frac{\beta}{1+2\delta^2}}\frac{\sqrt{(2m)!}}{2^m m!}\left(\frac{2\alpha^2\beta^2}{1+2\delta^2} - 1\right)^m \tag{31}$$

Assume that $\lfloor\cdot\rfloor$ is the value of a number rounded downwards to the nearest integer, and substitute (31) into (30), the weights of the GKQ points are

$$\begin{aligned}
w_j^{GK} &= \left(\frac{1}{1+2\delta^2}\right)^{1/2}w_j^{GH}e^{\delta^2(\xi_j^{GK})^2}\cdot \\
&\sum_{m=0}^{\lfloor(N-1)/2\rfloor}\frac{1}{2^m m!}\left(\frac{2\alpha^2\beta^2}{1+2\delta^2} - 1\right)^m H_{2m}\left(\xi_j^{GH}\right)
\end{aligned} \tag{32}$$

Equations (28) and (32) define the univariate GKQ nodes and the associated closed-form approximate weights derived from the truncated Mercer decomposition on scaled Gauss–Hermite nodes. It should be emphasized that these closed-form weights are not obtained by directly solving $Kw = k_\mu$ for the full Gaussian kernel. Rather, they are derived from the scaled Gauss–Hermite/Mercer construction and are used here as a computationally convenient approximation to the exact fixed-node kernel-quadrature weights. Now

consider the n_x dimensional multivariate integral in (3), and the GKQ integral can be shown as

$$I(g) = \sum_{i_1=1}^{n_s} \cdots \sum_{i_{n_x}=1}^{n_s} g\left(\left[\xi_{i_1}^{GK}, \cdots, \xi_{i_{n_x}}^{GK}\right]\right) w_{i_1}^{GK} \cdots w_{i_{n_x}}^{GK} \tag{33}$$

The tensor product rule can be utilized to obtain w_j^{GK} and ξ_j^{GK}. Substitute the multivariate GKQ points and weights into the Gaussian filter framework, and the Gaussian kernel quadrature filter (GKQKF) can be obtained. The GKQKF can be regarded as a kind of scaled multivariate GHQ rule. It means that the computational cost of GKQKF is similar to that of GHQKF, but the accuracy can be potentially improved based on the parameters α, β, and δ for a certain n_s.

The exact fixed-node KQ weights are determined by minimizing the worst-case integration error over the RKHS of the Gaussian kernel. The closed-form GKQ weights used in Equations (28) and (32) are Mercer-based practical approximations to these exact weights. This shift from polynomial-exactness-based weighting to kernel-adapted weighting is the main distinction between the GHQ and GKQ paradigms.

3.2. Sparse-Grid Gaussian Kernel Quadrature Kalman Filter

The GKQKF with $(n_s)^{n_x}$ points becomes computationally expensive as the state dimension or quadrature accuracy increases. This is unfavorable for the practical applications of sensor information fusion, especially in high-dimensional cases. The reason lies in the fact that the tensor product rule is not exact in a class of polynomials of a bounded total, but for a tensor product of univariate polynomials [24]. As a result, the sparse-grid rule is used to extend the univariate GKQ to the multivariate system here. Thus, its computational cost grows much more slowly than that of the full tensor-product rule.

According to Smolyak's rule, the multivariate Gaussian integral with the GKQ rule is given by

$$I(g) = \sum_{q=L-n_x}^{L-1} (-1)^{L-1-q} \boldsymbol{C}_{n_x-1}^{L-1-q} \sum_{\Xi \in \mathbb{N}_q^{n_x}} \left(I_{i_1}^{GK} \otimes \cdots I_{i_{n_x}}^{GK}\right)(g) \tag{34}$$

where $L \in N$ is the accuracy level means that $I(g)$ is exact for all polynomials of the form $x_1^{i_1}, x_2^{i_2}, \cdots x_{n_x}^{i_{n_x}}$ with $\sum_{j=1}^{n_x} i_j \leq 2L-1$. The $C_{n_x-1}^{L-1-q}$ is a binomial coefficient. $\otimes$ stands for tensor product. $I_{i_j}^{GK}$ is the univariate GKQ rule with the accuracy level $i_j \in \Xi \triangleq (i_1, \ldots, i_{n_x})$. It is exactly to at least the $(2i_j - 1)^{th}$ order of all polynomials, where $\Xi \triangleq (i_1, \ldots, i_{n_x})$ is an accuracy level sequence. $\mathbb{N}_q^{n_x}$ is a set of accuracy level sequences defined by

$$\begin{cases} \mathbb{N}_q^{n_x} = \left\{\Xi : \sum_{j=1}^{n_x} i_j = n_x + q\right\}, & q \geq 0 \\ \mathbb{N}_q^{n_x} = \varnothing, & q < 0 \end{cases} \tag{35}$$

where $\varnothing$ is the empty set and q is ranging from $L - n_x$ to $L - 1$.

The integral approximation, which uses the sparse-grid rule, can be expressed explicitly as

$$I_{n_x,L}(g) = \sum_{q=L-n_x}^{L-1} \sum_{\Xi \in \mathbb{N}_q^{n_x}} \sum_{\zeta_1 \in \zeta_{i_1}^{GK}} \cdots \sum_{\zeta_{n_x} \in \zeta_{n_x}^{GK}} g(\xi_{s_1}, \cdots, \xi_{n_x}) \cdot \left\{(-1)^{L-q-1} C_{n_x-1}^{L-1-q} \prod_{j=1}^{n_x} w_{i_j}\right\} \tag{36}$$

where $\zeta_{i_j}^{GK}$ is the univariate GKQ point set with the accuracy level i_j, which contains i_j or more points. $[\xi_{s_1}, \cdots, \xi_{n_x}]^T$ is an SGKQ point and $\xi_{s_j} \in \zeta_{i_j}^{GK}$.

The set of sparse Gaussian kernel points is given by

$$\zeta_{n_x,L}^{SGK} = \bigcup_{q=L-n_x}^{L-1} \bigcup_{\Xi \in \mathbb{N}_q^{n_x}} \left(\xi_{i_1} \otimes \xi_{i_2} \otimes \ldots \otimes \xi_{i_{n_x}}\right) \tag{37}$$

where $\cup$ stands for union set. To obtain $\xi_{n_x,L}^{SGK}$, pick all the values of q successively such that $L-n_x, L-n_x+1, \ldots, L-1$, and select all the univariable GKQ points combination for each q. During the process, if the point is new, it is added to the set $\xi_{n_x,L}^{SGK}$, and then the weight is

$$w_j^{SGK} = (-1)^{L-1-q} C_{n_x-1}^{L-1-q} \prod_{s=1}^{n_x} w_{i_s} \tag{38}$$

If the point has been in $\xi_{n_x,L}^{SGK}$, the number of the set is not changed, but the corresponding weight will be recalculated according to the previous weight of the point, such that:

$$w_j^{SGK} = w_j^{SGK} + (-1)^{L-1-q} C_{n_x-1}^{L-1-q} \prod_{s=1}^{n_x} w_{i_s} \tag{39}$$

Therefore, the SGKQ points and weights can be generated using the following Algorithm 1. The corresponding flowchart is shown in Figure 1.

Algorithm 1: Generate SGKQ points and weights

Input : System dimension n_x and accuracy level L.
Output: The SGKQ points $\xi_{n_x,L}^{SGK}$ and corresponding weights.
Step 1: Generate univariate GKQ points and weights with level $1, \ldots, 2L-1$ based on univariate GHQ rules with (28) and (32).
Step 2: For each specific $q(L-n_x$ to $L-1)$, substitute it with (35), and the corresponding $\mathbb{N}_q^{n_x}$ can be obtained, and then all of elements combination in $\mathbb{N}_q^{n_x}$ are determined.
Step 3: For each point $[\xi_{s_1}, \cdots, \xi_{n_x}]^T$ in $\xi_{i_1} \otimes \xi_{i_2} \otimes \ldots \otimes \xi_{i_{n_x}}$, if it is new, add it to $\xi_{n_x,L}^{SGK}$, and calculate the corresponding weight according to (38). If the point already exists, recalculate the weight based on (39).
Repeat Steps 2 to 3 until all the points for each specific q is checked.

Substitute the SGKQ points and weights into the Gaussian filter framework, and the new filter named SGKQKF, can be obtained.

In a certain accuracy level, the number of points for the SGKQ increases polynomially with dimension n_x, but the number of points for the GKQ and GHQ increases exponentially with dimension n_x. Table 1 shows the number of points comparison between SKGQKF and GKQKF/GHQKF when the accuracy level is 2 and 3.

Table 1. The point number comparisons between SGKQKF and GKQKF/GHQKF when the accuracy level is 2 and 3.

	L = 2	**L = 3**
SGKQKF	$2n_x+1$	$2n_x^2+4n_x+1$
GKQKF/GHQKF	2^{n_x}	3^{n_x}

As shown in Table 1, when $L = 2$, if $n_x > 2$, the number of points for SGKQKF is less than that of the GKQKF. When $L = 3$, if $n_x > 3$, the number of points for SGKQKF is less than that of the GKQKF or GHQKF. This shows that the SGKQKF can substantially reduce

the number of quadrature points, and hence the computational cost, in higher-dimensional problems relative to the GHQKF and GKQKF.

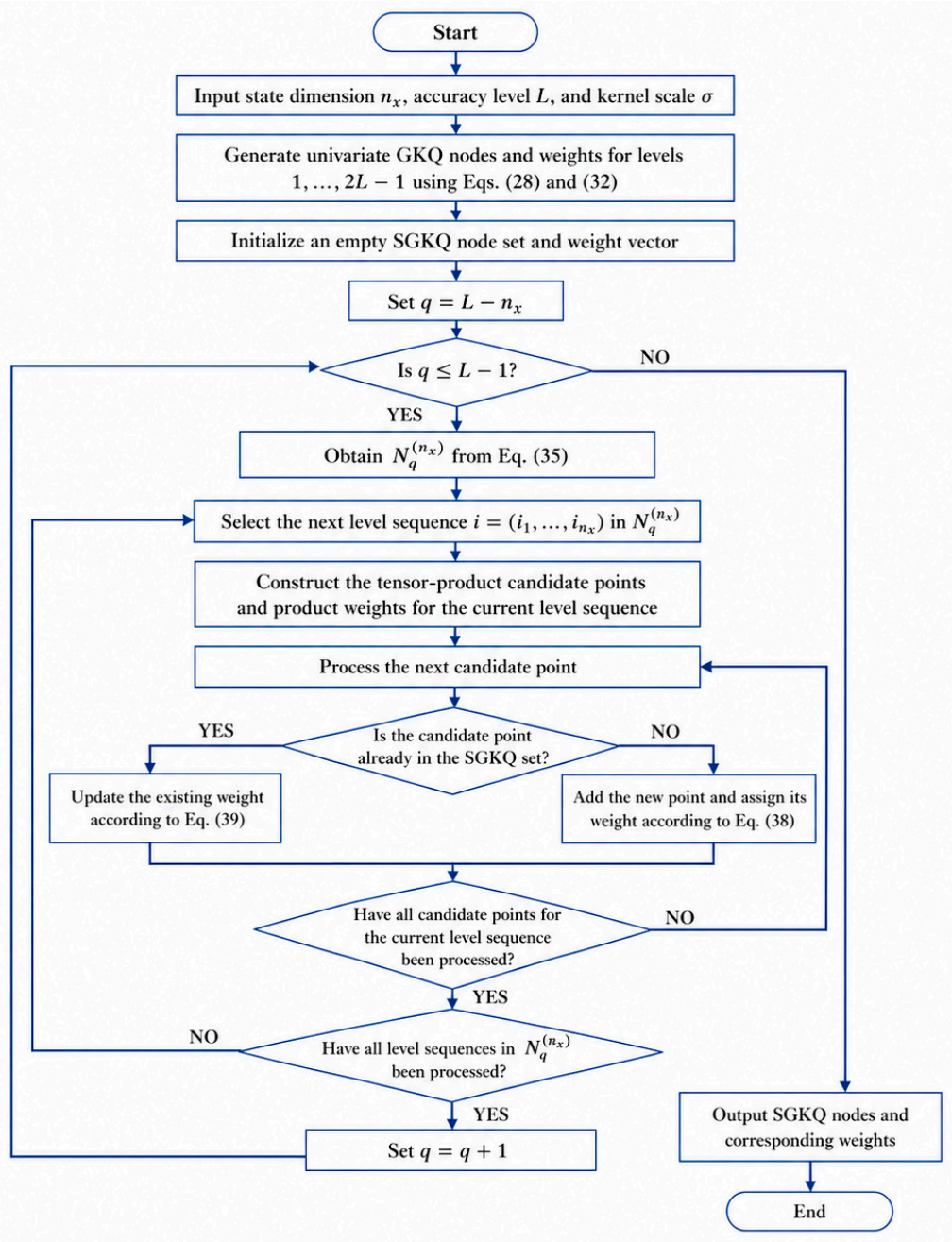


Figure 1. Flowchart of the SGKQ points and weights generator.

3.3. Relationships Among SGKQKF, GKQKF, and the Other Filters

As discussed, GKQKF uses the GKQ rule, which can be regarded as a scaled GHQ rule with σ. If $\sigma \to \infty$, then $\varepsilon = 0$, $\beta = 1$, and $\delta = 0$. Substituting them into (28) and (32), we have

$$\xi_j^{GK} = \frac{1}{\sqrt{2}\alpha\beta}\xi_j^{GH} = \xi_j^{GH}, w_j^{GK} = w_j^{GH} \tag{40}$$

That is, when $\sigma \to \infty$, the GKQ rule is degraded into the GHQ rule, so that the GHQKF is a subset of GKQKF. In this sense, the GKQKF is more flexible than the GHQKF, because a suitable value of σ can be chosen to further improve the filtering accuracy with the same number of points.

The proposed SGKQKF uses the sparse-grid rule instead of the tensor product rule to alleviate the curse of dimensionality problem. Compared with the sparse-grid GHQKF (SGHQKF), the SGKQKF uses the GKQ rule instead of the GHQ rule, which may improve accuracy without increasing the number of quadrature points. Additionally, the SGKQKF will be degraded into the SGHQKF when $\sigma \to \infty$. That is, the SGHQKF is a subset of the proposed SGKQKF.

It has been shown that the UKF with the parameters $\kappa = 3 - n_x$ and $\kappa = 1 - n_x$ are subsets of the SGHQKF at the level-2 accuracy [24]. Also, the UKF with those specific parameters can be regarded as a subset of the proposed SGKQKF. It is because the SGKQ rule will become the SGHQ rule when $\sigma \to \infty$. Compared with the UKF, the SGKQKF is generally more accurate when the accuracy level is greater than 2. Consequently, the SGKQKF has potential superiority in the nonlinear filtering problems.

It is noted that Smolyak-type sparse-grid constructions may involve negative weights. In the present work, no loss of positive semi-definiteness was observed in the reported experiments, although a general theoretical guarantee remains an open question.

3.4. Worst-Case Integration Error Analysis of the GKQ and SGKQ Rules

In this subsection, we analyze the approximation quality of the GKQ and SGKQ rules from the perspective of the reproducing kernel Hilbert space (RKHS), which provides a principled way to compare quadrature rules beyond polynomial exactness.

Let $H(k)$ denote the RKHS associated with the Gaussian kernel $k(\cdot,\cdot)$ in (20). Consider a quadrature rule

$$Q(f) = \sum_{j=1}^{n} w_j f(\varepsilon_j) \tag{41}$$

approximating the Gaussian integral $I(f) = \int f(x)d\mu(x)$. The worst-case integration error (WCE) is defined as

$$e(Q, H(k))^2 = c_0 - 2k_\mu^T w + w^T K w \tag{42}$$

where $c_0 = \int\int k(x,y)d\mu(x)d\mu(y), K \in R^{n\times n}$ with $[K]_{i,j} = k(\xi_i, \xi_j)$, and $k_\mu \in R^n$ with $[k_\mu]_j = \int k(\xi_j, y)d\mu(y)$. For the Gaussian kernel and distinct nodes, K is strictly positive definite, and hence (42) is a strictly convex quadratic function of w.

Proposition 1. *Optimality of the exact fixed-node kernel-quadrature weights [12].*

For a fixed node set $\Xi = \{\xi_1, \ldots, \xi_n\}$, the exact fixed-node kernel-quadrature weight vector

$$w^* = K^{-1}k_\mu \tag{43}$$

uniquely minimizes the RKHS worst-case integration error $e(Q, H(k))$ among all quadrature rules supported on Ξ. Consequently, for any alternative weights $\widetilde{w}$ on the same nodes,

$$e(Q^*, H(k)) \le e(\widetilde{Q}, H(k)) \tag{44}$$

with equality if and only if $\widetilde{w} = w^*$. The above optimality result applies to the exact fixed-node kernel-quadrature weights $w^* = K^{-1}k_\mu$. In contrast, the weights used in the implemented SGKQKF, given in Equations (28) and (32), are closed-form weights obtained from the scaled Gauss–Hermite/Mercer construction. Therefore, these closed-form weights are interpreted in this paper as a practical approximation to the exact fixed-node KQ weights, rather than as the exact minimizer of the full RKHS worst-case error. The limiting relation between the implemented closed-form GKQ rule and the GHQ rule is discussed separately in Section 3.3.

Proposition 2. *Tensor-grid case for exact fixed-node kernel quadrature.*

Assume a product measure $\mu(x) = \Pi\mu^d(x^d)$, a tensor-product kernel $k(x,y) = \Pi k^d(x^d, y^d)$, and a tensor-grid node set $\Xi = \Xi_1 \cdot \cdots \cdot \Xi_{n_x}$. Then, the kernel matrix and kernel-mean vector satisfy

$$K = K_1 \otimes \cdots \otimes K_{n_x}, k_\mu = k_{\mu,1} \otimes \cdots \otimes k_{\mu,n_x} \tag{45}$$

and the unique WCE-minimizing weight vector is

$$w^* = K^{-1}k_\mu = (K_1^{-1}k_{\mu,1}) \otimes \cdots \otimes (K_{n_x}^{-1}k_{\mu,n_x}) \tag{46}$$

Therefore, the tensor product of the univariate exact fixed-node KQ weight vectors is the unique WCE minimizer among all quadrature rules supported on the same tensor-grid nodes.

Proof. For a tensor-product kernel and tensor-grid nodes, the (*i*,*j*)-th entry of the multivariate kernel matrix satisfies

$$k(\xi_i, \xi_j) = \prod k^d(\xi_i^d, \xi_j^d) \tag{47}$$

which yields the Kronecker factorization $K = K_1 \otimes \cdots \otimes K_{n_x}$. Similarly, by the product structure of the measure, $[k_\mu]_j = \int \prod k^d(\xi_j^d, y^d) d\mu^d(y^d) = \prod \int k^d(\xi_j^d, y^d) d\mu^d(y^d)$, giving $k_\mu = k_{\mu,1} \otimes \cdots \otimes k_{\mu,n_x}$. By the standard Kronecker inverse property $(A \otimes B)^{-1} = A^{-1} \otimes B^{-1}$, we obtain

$$w^* = K^{-1}k_\mu = (K_1^{-1}k_{\mu,1}) \otimes \cdots \otimes (K_{n_x}^{-1}k_{\mu,n_x}) = w_1^* \otimes \cdots \otimes w_{n_x}^* \tag{48}$$

This shows that the tensor product of the univariate exact fixed-node KQ weight vectors is the unique minimizer of the strictly convex quadratic (42) on the full tensor grid. Since the GHQ tensor-product weights correspond to a different weight vector on the same node set, Proposition 1 guarantees that their WCE is no smaller.

Since each K_d is strictly positive definite for the Gaussian kernel with distinct nodes, the Kronecker product $K = K_1 \otimes \cdots \otimes K_{n_x}$ is also strictly positive definite. Hence, the quadratic form remains strictly convex in the multivariate case, and the minimizer $w^* = K^{-1}k_\mu$ is unique. This proposition characterizes the exact fixed-node KQ rule on tensor-grid nodes. The tensor-product GKQKF implemented in this paper uses the closed-form scaled Gauss–Hermite/Mercer weights as a practical approximation to these exact tensor-grid KQ weights. □

Remark 1. *On the Sparse-Grid Case.*

Proposition 2 applies to the tensor-product construction. For the Smolyak sparse-grid construction, the quadrature rule is a signed linear combination of tensor-product components, and, therefore, a simple closed-form factorization of the RKHS worst-case error is generally unavailable. Nevertheless, for any given (n_x, L, σ), the squared MMD can be computed directly from Equation (42) using the SGKQ or SGHQ nodes and weights. Therefore, Table 2 reports numerical MMD^2 comparisons between SGKQ and SGHQ as empirical diagnostics of the RKHS discrepancy in the tested configurations, rather than as a formal sparse-grid RKHS optimality theorem. The pseudo-inverse fixed-node KQ diagnostic is included only to assess the numerical difference between the implemented closed-form Mercer-based weights and the corresponding fixed-node KQ weights on the same SGKQ node sets.

Table 2. MMD^2 comparison among SGHQ, the implemented closed-form SGKQ rule, and the fixed-node KQ diagnostic on the same SGKQ node sets.

n_x	L	σ	MMD^2 (SGHQ)	MMD^2 (SGKQ, w^{GK})	MMD^2 (SGKQ, Fixed-Node KQ Diagnostic)	Reduction
3	2	1	5.87×10^{-2}	1.02×10^{-2}	1.02×10^{-2}	82.65%
3	2	2	2.92×10^{-3}	7.32×10^{-4}	7.32×10^{-4}	74.97%
3	2	4	2.96×10^{-5}	8.49×10^{-6}	8.49×10^{-6}	71.35%
5	2	2	1.92×10^{-2}	1.50×10^{-3}	1.50×10^{-3}	92.16%
5	3	2	4.19×10^{-4}	3.58×10^{-5}	3.56×10^{-5}	91.47%
10	2	2	1.99×10^{-1}	2.22×10^{-3}	2.22×10^{-3}	98.88%

As shown in Table 2, the MMD^2 values obtained with the implemented closed-form SGKQ weights are very close to those obtained with the pseudo-inverse fixed-node KQ diagnostic on the same node sets in all tested configurations. In addition, SGKQ reduces the MMD^2 by approximately 71.35–98.88% relative to SGHQ, indicating a substantial empirical reduction in RKHS discrepancy. These results support the practical effectiveness of the proposed SGKQ rule without implying exact sparse-grid RKHS optimality.

Remark 2. *Role of the Scaled Factor* σ.

The WCE depends on σ through the kernel and the resulting GKQ weights. As $\sigma \to \infty$, the GKQ rule reduces to the GHQ rule in our formulation (Equation (40)), while finite σ yields kernel-adapted weights that often improve the accuracy–efficiency trade-off, consistent with the experiments in Section 4.

Remark 3. *Quantitative Bound in the Univariate Case.*

For the univariate case, the squared WCE admits the Mercer–eigenpair representation:

$$e(Q, H(k))^2 = \sum_{n \geq 1} \lambda_n \left(\mu(\phi_n) - \sum_{j=1}^{n_s} w_j \phi_n(\xi_j) \right)^2 \tag{49}$$

In particular, for the GKQ design in [12], the above expression yields a truncation-type bound dominated by the spectral tail $\Sigma_{n \geq n_s} \lambda_n [\mu(\phi_n)]^2$. Since λ_n decays geometrically for the Gaussian kernel, the WCE decreases rapidly with n_s, which helps explain the improved accuracy–efficiency trade-off observed in Section 4.

4. Numerical Experiments

In this section, one and high-dimensional nonlinear problems are used to test the proposed algorithms. The CKF is adopted as the representative third-degree deterministic Gaussian filtering baseline. Then a typical target tracking problem is built [25], and the algorithms, such as the CKF, GHQKF, GKQKF and SGKQKF, are compared. All numerical experiments were implemented in MATLAB R2025b and run on a computer equipped with an Intel(R) Core(TM) Ultra 7 258V processor, 32 GB RAM, and Windows 11. No parallel acceleration was used.

4.1. Performance Test of the Algorithms in One and High-Dimensional Nonlinear Problems

Experiment 1. *A 1-D Example. This one-dimensional experiment serves as a basic validation of the proposed GKQ-based filtering framework in a simple nonlinear setting. It should be noted that, in*

the univariate case, the Smolyak sparse-grid construction reduces to the corresponding underlying one-dimensional quadrature rule rather than producing a genuinely sparse multidimensional grid. Therefore, this experiment is not intended to demonstrate a sparse-grid advantage over the tensor-product construction. Instead, it is used to examine the behavior of the GKQ-based weights and the influence of the kernel scale parameter σ in a low-dimensional setting. Considering the 1-D nonlinear problem:

$$x_k = 0.5x_{k-1} + \frac{25x_{k-1}}{1 + x_{k-1}^2} + v_{k-1} \tag{50}$$

$$y_k = 5\sin(2x_k) + e_k \tag{51}$$

where x_k denotes the state at time index k, $v_k \sim N(0, Q)$ is the Gaussian process noise with $Q = 10$, and $e_k \sim N(0, R_k)$ is the Gaussian measurement noise with $R_k = 1$.

The root mean square error (RMSE) at time index k ($k = 1, 2, \ldots, 100$) is defined as

$$\mathrm{RMSE}_k = \sqrt{\frac{1}{n_L}\sum_{j=1}^{L}(\hat{x}_{k,j} - x_j)^2} \tag{52}$$

where $n_L = 200$ is Monte-Carlo times and $i = 1, 2, \ldots, n_L$.

The CKF, GHQKF, SGHQKF, GKQKF, and SGKQKF are compared. The GHQKF and GKQKF use three univariate points with the tensor product rule. The SGHQKF and SGKQKF use the accuracy level $L = 3$. The GKQKF and the SGKQKF use the same scaled factor $\sigma = 1$ in this case.

As shown in Figure 2, the CKF provides the lowest accuracy for this nonlinear problem. Compared with the GHQKF, the GKQKF achieves better accuracy in this tested setting. The SGKQKF also performs better than the SGHQKF, indicating that the kernel-adapted weights can improve the corresponding quadrature rule under the present parameter setting. Additionally, the SGKQKF performs better under some values of the scaled factor σ. This indicates that the scaled factor σ can be utilized to obtain better accuracy.

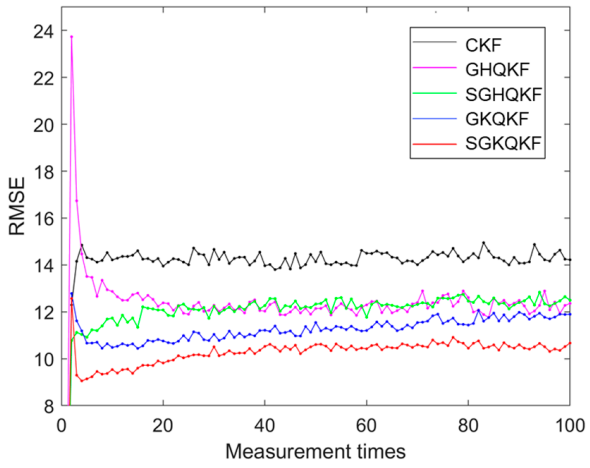


Figure 2. The RMSEs of different filtering algorithms in Experiment 1.

Experiment 2. *A high-dimensional nonlinear filtering problem. The state-space model is:*

$$x_k = 2\cos(x_{k-1}) + v_k \tag{53}$$

$$y_k = \sqrt{(1 + x_k^T x_k)} + e_k \tag{54}$$

Experiment 2 considers a five-dimensional nonlinear system with a scalar measurement. The state vector is denoted by $x_k \in \mathbb{R}^5$, and the measurement is $y_k \in \mathbb{R}$. Thus, $n_x = 5$ and $n_z = 1$, and the measurement function maps the state space to a scalar observation. The process-noise covariance is set to $Q = 2.56I_5$, and the measurement-noise variance is $R_k = 25$. The initial state and covariance are $x_0 = [1, 2, 3, 4, 5]^T$ and $P_0 = 10I_5$. The CKF, GHQKF, and SGHQKF are set to the same as Experiment 1. The GKQKF and the SGKQKF use the scaled factor $\sigma = 2$, and the other parameters are the same as Experiment 1. Also, the Monte-Carlo times are 200.

The RMSEs of different algorithms are shown in Figure 3. Table 3 shows the computational time investigated relative to that of the CKF.

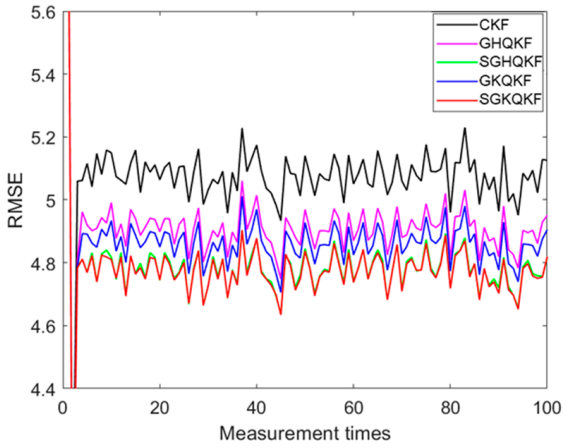


Figure 3. The RMSEs of different filtering algorithms in Experiment 2.

Table 3. The relative computation time (RCT) of different filtering algorithms.

Filters	RCT
CKF	1
GHQKF	1.947
GKQKF	1.945
SGHQKF	1.288
SGKQKF	1.283

As shown in Figure 3, it is not surprising that the SGKQKF performs better than SGHQKF and GHQKF, which means the GKQ rule is more accurate than the GHQ rule. Additionally, as shown in Table 3, the SGKQKF runs much faster than the GHQKF and the GKQKF because it uses 71 points by means of the sparse-grid rule, which is more computationally efficient than the GKQKF and the GHQKF with 243 points.

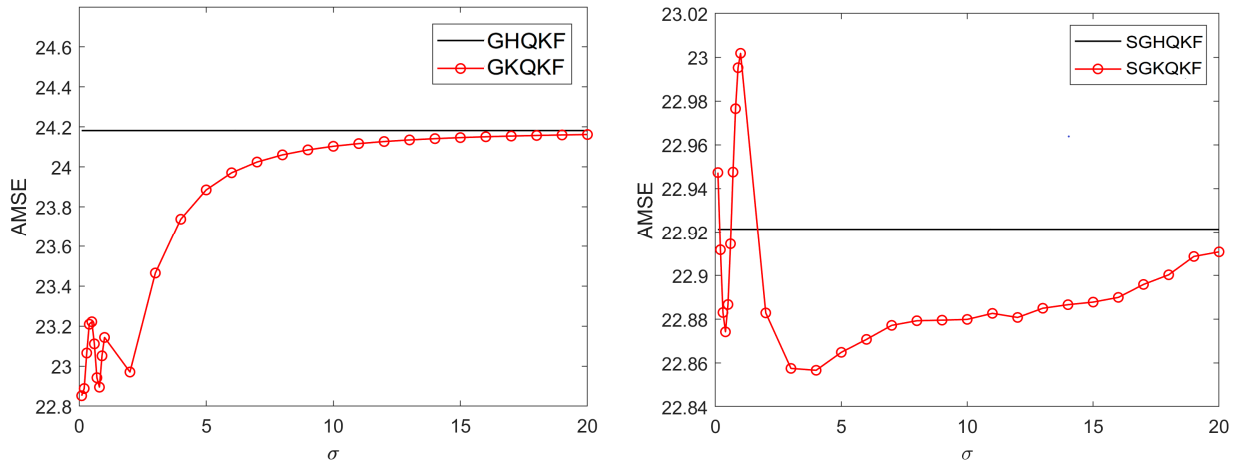
To further illustrate the effect of the scaled factor σ on the GKQKF and the SGKQKF, comparisons of the GKQKF and the SGKQKF with different σ values are considered. The initial parameters of the system and filters are set to the same as Experiment 2.

The mean squared error (MSE) for each Monte-Carlo run is defined as

$$\mathrm{MSE}_i = \frac{1}{n}\sum_{k=1}^{n}\left[\sum_{j=1}^{n_x}\left(x_{j,k} - \hat{x}_{j,k|k,i}\right)^2\right] \tag{55}$$

where $n_L = 200$ is Monte-Carlo times and $i = 1, 2, \cdots, n_L$.

The average MSEs (AMSEs) of the GKQKF and the SGKQKF with different σ parameters are compared. In this experiment, $\sigma \in (0, 20]$. When $\sigma \leq 1$, the test step is chosen as 0.1. When $\sigma > 1$, the test step is 1. The results are shown in Figure 4a,b.



(**a**) (**b**)

Figure 4. (**a**) Comparisons of GHQKF and GKQKF with different values of σ. (**b**) Comparisons of SGHQKF and SGKQKF with different values of σ.

As shown in Figure 4, when $\sigma < 1$, the performance of the GKQKF and SGKQKF is relatively unstable. This behavior is consistent with the fact that a small kernel scale may compress the effective GKQ node locations, so that the quadrature points may not sufficiently cover the posterior density. For moderate and larger values of σ, the GKQ-based filters become more stable and achieve accuracy comparable to, and in some tested cases better than, their GHQ-based counterparts. As σ further increases, the implemented GKQ rule gradually approaches the corresponding GHQ rule, which is consistent with the limiting behavior of the implemented closed-form scaled Gauss–Hermite construction. Therefore, the performance gap between GKQKF and GHQKF becomes smaller for large σ. Compared with the SGHQKF, the SGKQKF uses the same sparse-grid structure and the same number of points, while introducing the kernel scale parameter σ, which provides additional flexibility in the quadrature weights. These observations are empirical and depend on the tested nonlinear model and parameter range. In the present experiments, σ is selected empirically within the tested range; developing an adaptive or theoretically guided selection rule for σ is left for future work.

4.2. Bearings-Only Tracking with Angle Measurements

Bearings-only tracking (BOT) using passive angle sensors is a representative nonlinear estimation problem that arises in surveillance and aerospace sensing applications [25]. The highly nonlinear arctangent measurement model makes BOT a challenging benchmark for nonlinear filters.

The target is assumed to be in linear motion with nearly constant velocity for convenience. The state vector of the target at time index k ($k = 1, 2, \cdots n$) can be given by $x_k = [x_k^t, y_k^t, \dot{x}_k^t, \dot{y}_k^t]^T$, where $x_k^t = [x_k^t, y_k^t]^T$ is the position vector and $\dot{x}_k^t = [\dot{x}_k^t, \dot{y}_k^t]^T$ is the velocity vector. The single observer measures the angle information of the target. It is assumed to be maneuvering where the start coordinate is $(0, 0)$, and the state vector is expressed by $x_{k,i}^o = [x_{k,i}^o, y_{k,i}^o, \dot{x}_{k,i}^o, \dot{y}_{k,i}^o]^T$, where $i = 1, 2, \cdots, n_z$. Thus, the system model can be written as

$$x_k = Fx_{k-1} + v_{k-1} \tag{56}$$

$$z_{k,i} = \arctan\left(\frac{x_k^t - x_{k,i}^o}{y_k^t - y_{k,i}^o}\right) + e_{k,i} \tag{57}$$

where F is the transition matrix, $z_{k,i}$ is the i^{th} component of z_k at time index k such that $z_k = [z_{k,1}, z_{k,2}, \ldots z_{k,n_z}]^T$, v_{k-1} and $e_k = [e_{k,1}, e_{k,2}, \ldots e_{k,n_z}]^T$ are generally assumed independent and identically distributed zero mean Gaussian noise with the covariance matrix of Q and $R \sim N(0, (1.5\text{deg})^2)$, respectively. Defining $\Delta = 1$ min as the measurement interval, $p = 10^{-11}$ km^2/s^3 is the process noise intensity, and Q and F can be given by

$$Q = \begin{bmatrix} \Delta^3/3 & 0 & \Delta^2/2 & 0 \\ 0 & \Delta^3/3 & 0 & \Delta^2/2 \\ \Delta^2/2 & 0 & \Delta & 0 \\ 0 & \Delta^2/2 & 0 & \Delta \end{bmatrix} p, \; F = \begin{bmatrix} 1 & 0 & \Delta & 0 \\ 0 & 1 & 0 & \Delta \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Equation (57) is written using the arctangent relation as a compact mathematical expression of the bearing angle, but the numerical implementation evaluates the bearing using the two-argument atan2 function to preserve the correct quadrant information. In all BOT simulations, the bearing innovation is wrapped into $(-\pi, \pi]$ before the Kalman update.

The root mean square error in position (RMSE$_{pos}$) at time index k is defined as

$$\text{RMSE}_{pos,k} = \sqrt{\frac{1}{L}\sum_{j=1}^{L}(\hat{x}_{k,j}^t - x_{k,j}^t)^2 + (\hat{y}_{k,j}^t - y_{k,j}^t)^2} \tag{58}$$

where $L = 200$ is Monte-Carlo times and $j = 1, 2, \cdots, L$.

For fair comparison, the same initializations are set to different algorithms. The RMSE$_{pos}$ of the CKF, GHQKF, GKQKF with $\sigma = 4$, and SGKQKF with $\sigma = 4$ are shown in Figure 5. Table 4 shows the computational time investigated relative to that of the CKF.

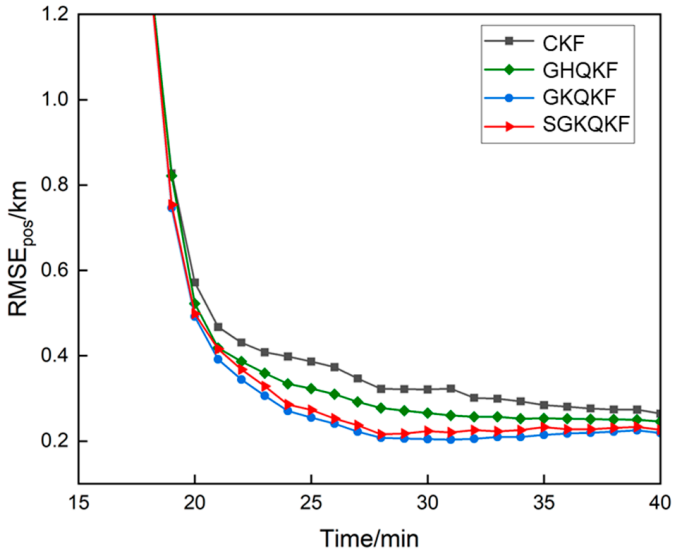


Figure 5. The RMSE$_{pos}$ of different filtering algorithms in bearings-only tracking problem.

Table 4. The RCT of different filtering Algorithms in BOT problem.

Filters	RCT
CKF	1
GHQKF	7.063
GKQKF	7.125
SGHQKF	3.833
SGKQKF	3.875

Figure 5 shows the tracking results from 15 to 40 min. This interval excludes the initial transient stage and focuses on the period in which the filters exhibit distinguishable convergence and tracking behavior. After approximately 40 min, all filters have reached stable tracking behavior and their relative trends remain almost unchanged. As shown in Figure 5 and Table 4, the performance of the proposed GKQKF and SGKQKF is better than that of the traditional CKF and GHQKF. The GKQKF and the SGKQKF perform similarly, but the SGKQKF requires approximately 54.4% of the runtime of the GKQKF, which means the SGKQKF strikes an appropriate balance between accuracy and computational efficiency.

Although the numerical results demonstrate the favorable accuracy–efficiency trade-off of the proposed SGKQKF in the tested examples, the evaluation remains mainly empirical. In particular, the kernel scale parameter σ is selected by scanning a finite set of candidate values, and confidence intervals are not reported for all Monte-Carlo performance metrics. Therefore, the reported results should be interpreted as empirical evidence in the tested configurations rather than as a formal finite-sample performance guarantee.

5. Conclusions

In this paper, we proposed the SGKQKF for nonlinear state estimation. By combining Gaussian kernel-based quadrature weighting with a Smolyak sparse-grid construction, the proposed method extends GKQ-type filtering from tensor-product grids to sparse-grid structures. We also clarified the distinction between the exact fixed-node KQ rule and the closed-form Mercer-based implementation: the former has an RKHS worst-case-error optimality property, while the latter is used as a practical approximation. Numerical results indicate that the proposed SGKQ rule reduces the MMD relative to the standard SGHQ rule in the tested configurations. The filtering experiments further demonstrate that the SGKQKF achieves a favorable accuracy–efficiency trade-off compared with conventional deterministic Gaussian filters, mainly by reducing the number of quadrature points through the sparse-grid construction.

This study still has several limitations. The sparse-grid MMD analysis should be interpreted as empirical diagnostic evidence rather than as a formal RKHS optimality theorem. Moreover, the kernel scale parameter σ is selected empirically in the numerical experiments, and confidence intervals are not reported for all Monte-Carlo performance metrics. Future work will focus on rigorous RKHS error analysis for Smolyak sparse-grid rules, adaptive or theoretically guided selection of σ, uncertainty quantification over Monte-Carlo runs, and validation under higher-dimensional or more challenging non-Gaussian and outlier-contaminated noise conditions.

Author Contributions: Conceptualization, H.W.; methodology, H.W. and Y.Z.; software, Y.Z.; validation, G.Z., M.Y. and C.L.; resources, Y.Z. and G.Z.; writing—original draft preparation, H.W.; writing—review and editing, Y.Z. and S.R. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China, grant numbers [61703420, 62073337], the Youth Innovation Team of Shaanxi Universities under grant 2023015, and the Natural Science Foundation of Shaanxi Province, grant number [2025JC-YBMS-749].

Data Availability Statement: Dataset available on request from the authors.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

DSFs	deterministically sampled filters
PF	particle filter
UKF	unscented Kalman filter
CKF	cubature Kalman filter
RKHS	reproducing kernel Hilbert space
MMD	maximum mean discrepancy
GKQ	Gaussian kernel quadrature
GHQ	Gaussian Hermite quadrature
SGKQ	Sparse Gaussian kernel quadrature
SGHQ	Sparse Gaussian Hermite quadrature
GKQKF	Gaussian kernel quadrature Kalman filter
GHQKF	Gaussian Hermite quadrature Kalman filter
SGKQKF	Sparse Gaussian kernel quadrature Kalman filter
SGHQKF	Sparse Gaussian Hermite quadrature Kalman filter
RCT	the relative computation time

References

1. Wang, N.; Forger, D.B. The Level Set Kalman Filter for State Estimation of Continuous-Discrete Systems. *IEEE Trans. Signal Process.* **2022**, *70*, 631–642. [CrossRef]
2. Liu, X.; Liu, X.; Yang, Y.; Guo, Y.; Zhang, W. Variational Bayesian-Based Robust Cubature Kalman Filter with Application on SINS/GPS Integrated Navigation System. *IEEE Sens. J.* **2022**, *22*, 489–500. [CrossRef]
3. Wu, H.; Chen, S.; Yang, B.; Chen, K. Robust Derivative-Free Cubature Kalman Filter for Bearings-Only Tracking. *J. Guid. Control Dyn.* **2016**, *39*, 1865–1870. [CrossRef]
4. Jwo, D.-J.; Chang, Y.; Hsu, Y.-H.; Biswal, A. Student's t Kernel-Based Maximum Correntropy Criterion Extended Kalman Filter for GPS Navigation. *Appl. Sci.* **2025**, *15*, 8645. [CrossRef]
5. Li, S.; Guo, C. A novel inverse-wishart-student's t mixture distribution-based variational Bayesian Kalman filter. *IEEE Sens. J.* **2024**, *24*, 39325–39333. [CrossRef]
6. Julier, S.J. The scaled unscented transformation. In Proceedings of the 2002 American Control Conference, Anchorage, AK, USA, 8–10 May 2002; Volume 6, pp. 4555–4559. [CrossRef]
7. Arasaratnam, I.; Haykin, S. Cubature Kalman Filters. *IEEE Trans. Autom. Control* **2009**, *54*, 1254–1269. [CrossRef]
8. Arasaratnam, I.; Haykin, S.; Elliott, R.J. Discrete-time nonlinear filtering algorithms using Gauss–Hermite quadrature. *Proc. IEEE* **2007**, *95*, 953–977. [CrossRef]
9. Liu, H.; Wang, X.; Zhang, Y.; Chen, Y.; Li, X.; Liu, S. Mode-matching universal Kalman filter based on closed skew-normal distribution. *Adv. Space Res.* **2025**. [CrossRef]
10. Wang, G.W.; Cui, B.B.; Tang, C.Y. Robust cubature Kalman filter based on maximum correntropy and resampling-free sigma-point update framework. *Digit. Signal Process.* **2022**, *126*, 103495. [CrossRef]
11. Qian, C.; Zhang, E.; Gao, Y.; Chen, Q. An improved weight adaptive Gaussian sum algorithm based on sparse-grid quadrature filter for non-Gaussian models. *IET Control Theory Appl.* **2025**, *19*, e70019. [CrossRef]
12. Karvonen, T.; Särkkä, S. Gaussian kernel quadrature at scaled Gauss–Hermite nodes. *BIT Numer. Math.* **2019**, *59*, 877–902. [CrossRef]
13. Karvonen, T.; Kanagawa, M.; Särkkä, S. On the positivity and magnitudes of Bayesian quadrature weights. *Stat. Comput.* **2019**, *29*, 1317–1333. [CrossRef]
14. Briol, F.-X.; Oates, C.J.; Girolami, M.; Osborne, M.A.; Sejdinovic, D. Probabilistic integration: A role in statistical computation? *Stat. Sci.* **2019**, *34*, 1–22. [CrossRef]
15. Wang, J.; Chen, S.; Wu, H.; He, R.; Xu, H. Target tracking algorithm using square-root Gaussian kernel quadrature filters. *J. Xi'an Jiaotong Univ.* **2022**, *56*, 157–164. (In Chinese) [CrossRef]
16. Naik, A.K.; Upadhyay, P.K.; Singh, A.K. Gaussian kernel quadrature Kalman filter. *Eur. J. Control* **2023**, *71*, 100805. [CrossRef]
17. Wu, H.; Hu, H.; Chen, S.; Zeng, G. A GNSS interference source tracking method using the continuous-discrete Gaussian kernel quadrature Kalman filter. *GPS Solut.* **2023**, *27*, 182. [CrossRef]
18. Arasaratnam, I.; Haykin, S. Square-Root Quadrature Kalman Filtering. *IEEE Trans. Signal Process.* **2008**, *56*, 2589–2593. [CrossRef]

19. Li, W.; Wei, G.; Han, F.; Liu, Y. Weighted Average Consensus-Based Unscented Kalman Filtering. *IEEE Trans. Cybern.* **2016**, *46*, 558–567. [CrossRef]
20. Jia, B.; Xin, M.; Cheng, Y. High-degree cubature Kalman filter. *Automatica* **2013**, *49*, 510–518. [CrossRef]
21. Zhang, X.; Yang, Z.; Lin, X.; Xiang, W. Bootstrap and High-Degree Cubature Particle Filters for Nonlinear Systems With Cor-related Noise and Missing Measurements. *IEEE Sens. J.* **2025**, *25*, 13219–13231. [CrossRef]
22. Steinwart, I.; Scovel, C. Mercer's Theorem on General Domains: On the Interaction between Measures, Kernels, and RKHSs. *Constr. Approx.* **2012**, *35*, 363–417. [CrossRef]
23. Fasshauer, G.E.; McCourt, M.J. Stable Evaluation of Gaussian Radial Basis Function Interpolants. *SIAM J. Sci. Comput.* **2012**, *34*, A737–A762. [CrossRef]
24. Heiss, F.; Winschel, V. Likelihood approximation by numerical integration on sparse grids. *J. Econom.* **2008**, *144*, 62–80. [CrossRef]
25. Yang, X.; Yin, H.; He, S.; Xie, Y.; Shin, H. Trajectory Optimization for Multi-Sensor Multi-Target Search and Tracking with Bearing-Only Measurements. *Aerospace* **2023**, *10*, 652. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.