The Level Set Kalman Filter for State Estimation of Continuous-Discrete Systems

Ningyuan Wang and Daniel B. Forger

***Abstract*—We propose a new extension of Kalman filtering for continuous-discrete systems with nonlinear state-space models that we name as the level set Kalman filter (LSKF). The LSKF assumes the probability distribution can be approximated as a Gaussian and updates the Gaussian distribution through a time-update step and a measurement-update step. The LSKF improves the time-update step compared to existing methods, such as the continuous-discrete cubature Kalman filter (CD-CKF), by reformulating the underlying Fokker-Planck equation as an ordinary differential equation for the Gaussian, thereby avoiding the need for the explicit expression of the higher derivatives. Together with a carefully picked measurement-update method, numerical experiments show that the LSKF has a consistent performance improvement over the CD-CKF for a range of parameters. Meanwhile, the LSKF simplifies implementation, as no user-defined timestep subdivisions between measurements are required, and the spatial derivatives of the drift function are not explicitly needed.**

***Index Terms*—Bayesian filter, kalman-filter, level set, nonlinear filter.**

I. Introduction

KALMAN Filtering methods are used in many applications. A Bayesian filtering method updates a *state estimation* of the target given knowledge of the system and measurements [1]. The goal of these methods is to estimate the state of a target system where the dynamics are known, using measurements taken a fixed time intervals, and accounting for noise or uncertainty in the system and measurements. There are two parts to these methods. First, a new measurement is used to generate the best possible estimate of the system state. Second, that estimate is propagated forward using the system's dynamics until the subsequent measurement is available. Here, we present a method for accurately implementing that second step in the presence of noise.

The general framework for these problems was first described by Kalman [2]. A Kalman-Bucy type filtering consists of two steps: a *measurement-update* part that updates the estimation using the measurement and a state estimation from previous steps, and a *time-update* step that updates the state estimation between consecutive measurements. The level set Kalman filter (LSKF) method focuses on the improvement of the *time-update* step, and the discussions that follow are restricted to the *time-update* part unless we explicitly mention the *measurement-update*.

Assuming that the dynamics are linear (in space), and all noise is Gaussian, Kalman-Bucy filtering [2] gives an optimal way to estimate the system state for the *time-update*. However, in many cases, we would like to generalize this method to a system where the dynamics of interest are nonlinear. For such a nonlinear system, a Gaussian probability density function (PDF) is no longer preserved, even when the dynamics are quadratic [3]. When the dynamics are approximately linear for the region of state space where most of the PDF lies, Unscented Kalman filtering (UKF) [4] can provide a useful method. Additionally, the UKF is easy to implement since it does not require explicit evaluation of the Jacobian of the velocity field, which is not readily available in practical problems where, for example, the velocity field is implicitly defined.

Researchers have improved how the process noise is incorporated, but so far, methods are significantly more complicated than the UKF (e.g., requiring the explicit calculation of a Jacobian) or only work with specific numerical solvers. Good examples include the continuous-discrete Kalman filter [5] (CDKF) and the continuous-discrete Cubature Kalman filter [6] (CD-CKF). (Note the latter uses the Cubature Kalman transformation as introduced in [7] instead of the unscented Kalman transformation, however, it can be reformulated to use either, as explained in [8].) The CDKF in [5] addresses the continuous nature of the process noise; however, the derivation of their method involves approximations such that their method is not exact even if the dynamics are linear. Moreover, the computation of the prediction is significantly more complicated than the original UKF, eroding its advantage the CDKF offers by removing intermediate timesteps. The CD-CKF uses a 1.5-order Itô-Taylor expansion of the stochastic differential equation, which uses the Jacobian (or approximations of it) that can be difficult to calculate. Though the explicit Jacobian can be avoided by deriving specific Runge-Kutta methods as described in [9], this still complicates programming and limits the type of numerical solvers available.

Here, we propose the LSKF that addresses these issues. Our method: 1) does not require the Jacobian or any spatial partial derivative of the drift function explicitly, 2) allows the use of adaptive ordinary differential equation (ODE) solvers and

Manuscript received February 23, 2021; revised August 6, 2021 and November 14, 2021; accepted November 23, 2021. Date of publication December 9, 2021; date of current version February 4, 2022. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Athanasios A. Rontogiannis. This work was supported by National Science Foundation through under Grants NSF DMS-1714094 and NSF DMS-2052499. *(Corresponding author: Daniel B Forger.)*

Ningyuan Wang is with the Department of Mathematics, University of Michigan, Ann Arbor, MI 48109 USA (e-mail: nywang@umich.edu).

Daniel B. Forger is with the Department of Mathematics, Department of Computational Medicine and Bioinformatics, USA, and also with the Michigan Institute for Data Science, University of Michigan, Ann Arbor, MI 48109 USA (e-mail: forger@umich.edu).

Digital Object Identifier 10.1109/TSP.2021.3133698

frees the user from choosing the time discretization, and 3) shows performance improvements over the CD-CKF, even in the challenging test cases presented in [6]. From a theory point of view, our derivation of the method is based on the apparent velocity of the level set of the probability distribution, which is a novel approach to analyze these problems, and may enable further developments.

II. Problem Statement and Background

(Note on notation: we distinguish matrix or vector-valued quantities $\mathbf{v}$ versus scalar-valued quantities v_1 by using a bold font. A list of symbols is included in Table I in the appendix.)

A. Problem Formulation

A continuous dynamic discrete measurement system includes a continuous-time process described by a Fokker-Planck equation and a discrete measurement process with measurement noise.

The discrete measurement process is defined by a transformation h from state space to the observation space, together with a zero-mean Gaussian observation noise $\boldsymbol{\tau} \sim \mathcal{N}(\mathbf{0}, \mathbf{R})$. Suppose at the time of measurement, the state vector is $\mathbf{x}$, then the measurement $\mathbf{y}$ is given by:

$$\mathbf{y} = h(\mathbf{x}) + \boldsymbol{\tau}, \tag{1}$$

where $\boldsymbol{\tau} \sim \mathcal{N}(\mathbf{0}, \mathbf{R})$.

In between the time where two consecutive measurements are taken, we assume that the process noise is Gaussian, and the system equations are described by the Itô process [10]:

$$\frac{d\mathbf{x}}{dt} = \mathbf{v}(\mathbf{x}) + \sqrt{\mathbf{K}}\frac{d\beta}{dt}, \tag{2}$$

where $\mathbf{v}$ is the **drift function**, or velocity field defined by the dynamics, β is a standard d dimension Brownian process. $\mathbf{K}$ is an $d \times d$ positive semi-definite continuous **process noise matrix**, and $\mathbf{K} = \sqrt{\mathbf{K}}\sqrt{\mathbf{K}}^T$.

Then, the PDF u is described by the Fokker-Planck equation of the following form:

$$\frac{du}{dt} = \frac{1}{2}\nabla \cdot \mathbf{K}\nabla u - \nabla \cdot (\mathbf{v}u). \tag{3}$$

B. Brief Review of Existing Time-Update Methods

Under the assumption that the drift function $\mathbf{v}$ is linear in space and the process noise matrix $\mathbf{K}$ is constant, it can be shown that a Gaussian PDF u is preserved. (A proof of this fact using level sets is in the next section). In [2], the derivation of the *time-update* step is based on this observation.

One often wants to generalize this method to nonlinear models even when Gaussian distributions are no longer exactly preserved. One generalization would be to use the Jacobian of the drift function at the mean of the distribution, which is a key part of the Extended Kalman-Bucy Filter (EKF) method. One disadvantage of the EKF is the need for an explicit formula of the Jacobian of the drift function. The UKF is also derived based on the assumption of a local linearization of velocity; however, the explicit evaluation of the Jacobian is avoided.

In [6], after their comparison between the continuous-discrete cubature Kalman filter (CD-CKF), continuous-discrete unscented Kalman filter (CD-UKF), and continuous-discrete extended Kalman filter (CD-EKF), they concluded that "the CD-CKF is the choice for challenging radar problems". [2, p.4987] In [8], Kulikov and Kulikova presented a new filtering method named the accurate continuous-discrete extended Kalman filter (ACD-EKF), and compared it to the CD-CKF and CD-UKF. Note the implementation of the CD-UKF in [8] is more sophisticated than that in [6] as it uses the IT-1.5 that is the same as presented in [6] for the CD-CKF. With the improved implementation of the CD-UKF, Kulikov and Kulikova reported in [8] that the CD-CKF and the CD-UKF perform similarly. In addition, while the ACD-EKF requires less tuning than the CD-CKF, with sufficient timestep subdivision, the CD-CKF seems to outperform the ACD-EKF, as stated in the conclusion of [8]: *The highest accuracy is provided by the most time-consuming filters CD-CKF256 and CD-UKF256.* Therefore, we conclude that with sufficient timestep subdivision, the CD-CKF is still a benchmark method to compare against.

C. The Time-Update of the CD-CKF With Ito-Taylor Expansion

In [6], the Ito-Taylor expansion of order 1.5 (IT-1.5) is first introduced to the *time-update* step of the continuous-discrete filtering. It is confirmed in [11] that the Unscented Kalman filtering with IT-1.5 achieves similar performance to the CD-CKF. For the purpose of comparing time-update, the performance of the CD-CKF should suffice for a benchmark. Additionally, we noted that while the IT-1.5 should converge to the accurate result with a weak order of convergence 2, the implementation in [6] chooses to only discretize noise once between the measurements and does not converge to this result, presumably as a tradeoff to improve speed. For the sake of complete comparison, we also implemented a version with a proper IT-1.5 expansion that discretizes noise for every timestep subdivision.

Here we restate the *square-root form* of the CD-CKF, as derived in [6].

Time-update: For update with a timestep of Δt, we define the function

$$\mathbf{f}_d(\mathbf{x}, t) := \mathbf{x}(t) + \Delta t \mathbf{v}(\mathbf{x}(t), t) + \frac{1}{2}\Delta t^2(\mathbb{L}_0(\mathbf{v}(\mathbf{x}, t))), \tag{4}$$

where the opertor $\mathbb{L}_0$ is defined as

$$\mathbb{L}_0 := \frac{d}{dt} + \sum_{i=1}^{d} v_i \frac{\partial}{\partial x_i} + \frac{1}{2}\sum_{j,p,q=1}^{d} \sqrt{\mathbf{K}}_{p,j}\sqrt{\mathbf{K}}_{j,q}\frac{\partial^2}{\partial x_p \partial x_q}. \tag{5}$$

We also define the operator $\mathbb{L}(\mathbf{v})$ be the square matrix defined entrywisely with its (i, j)th element being $\mathbb{L}_j v_i$, where

$$\mathbb{L}_j := \sum_{i=1}^{d} \sqrt{\mathbf{K}}_{i,j}\frac{\partial}{\partial x_i}. \tag{6}$$

Then the time-update algorithm is as follows:

Require: Guess of initial state $\mathbf{x}_0$ at time t_0, and a factorization of a guess of covariance matrix $\mathbf{M}$. Drift velocity $\mathbf{v}$, continuous process noise matrix $\mathbf{K}$.

1: Find the $2\,d$ cubature points $\mathbf{x}_i = \mathbf{x}_0 + (\mathbf{M})_i, \mathbf{x}_{i+d} = \mathbf{x}_0 - (\mathbf{M})_i$, where $i = 1, \ldots, d$, and $(\mathbf{M})_i$ denotes the ith column of $\mathbf{M}$.

2: Evaluate the propagated cubature point: $\mathbf{x}_i^* = \mathbf{f}_d(\mathbf{x}_i, t_0)$, where where $i = 1, \ldots, 2\,d$.

3: Estimate the updated mean by the average of the cubature points:

$$\mathbf{x}_0^* := \frac{1}{2\,d}\mathbf{x}_i^*. \tag{7}$$

4: Estimate the factorization of the covariance matrix by the triangularization of the concatenated matrix:

$$\mathbf{M}^* = \text{tria}\left(\left[\mathbf{X}^* | \sqrt{\Delta t}(\sqrt{\mathbf{K}} + \tfrac{\Delta t}{2}\mathbb{L}(\mathbf{v}))| \sqrt{\tfrac{\Delta t^3}{12}}\mathbb{L}(\mathbf{v})\right]\right), \tag{8}$$

where $\text{tria}(\cdot)$ denotes applying a triangularization procedure such as the Gram-Schmidt based QR-decomposition, $\mathbf{X}^*$ is a matrix with ith column being $\mathbf{x}_i^*$, $\mathbb{L}(\mathbf{v}) = \mathbb{L}(\mathbf{v}(\mathbf{x}_0^*, t_0))$.

5: **return** Estimated updated mean $\mathbf{x}_0^*$ and updated factorization of the covariance matrix $\mathbf{M}^*$ at $t_0 + \Delta t$.

D. The Square Root Form of Cubature Kalman Measurement-Update

Here, we discuss the *measurement-update* method used in the CD-CKF and the LSKF. Since the operations from the time-update can cause $\mathbf{M}$ to be positive semi-definite, a measurement-update method that can accommodate a positive semi-definite matrix is required for reliability, as pointed out in [6]. We used the measurement-update method from the square root CD-CKF method, as stated in Appendix B of [6]. Since the notations used are different, the measurement-update of the square root CD-CKF is restated here for reference.

III. Derivation of the Time-Update of the Level Set Kalman Filter

In this section, we focus on deriving the *time-update* of the level set Kalman filter (LSKF). In the first subsection, we show that a Gaussian is preserved by a local linear approximation to the original Fokker-Planck equation by tracking its level set. In this process, we observe that the apparent velocity of the level set is given by the drift function plus an additional term which we name as the **diffusion velocity**. In the second subsection, using the apparent velocity of the level set, we derive a numerical method that tracks such Gaussian particles for the *time-update* step. In the third subsection, we state the **averaged velocity** version of the *time-update* part of the LSKF, which turns out to give better results numerically.

Require: Factorization of the predicted covariance matrix before measurement $\mathbf{M}$, predicted mean before measurement $\bar{\mathbf{x}}$, measurement $\mathbf{y}$, a factorization of the covariance of the measurement noise matrix $\sqrt{\mathbf{R}}$, measurement function $\mathbf{h}$

1: Find the concatenated cubature points matrix of size $d \times 2\,d$:

$$\mathbf{N} = \bar{\mathbf{x}} + \sqrt{2\,d}\left[\mathbf{M}| - \mathbf{M}\right], \tag{9}$$

where the vector-matrix addition is applied as $\bar{\mathbf{x}}$ added to each **column** of the concatenated matrix $[\mathbf{M}| - \mathbf{M}]$

2: Evaluated the propagated cubature points

$$\mathbf{Y} = \mathbf{h}(\mathbf{N}), \tag{10}$$

where the measurement function $\mathbf{h}(\cdot)$ is evaluated on each **column**.

3: Estimate the predicted measurement

$$\bar{\mathbf{y}} = \frac{1}{2\,d}\sum_{i=1}^{2\,d}\mathbf{Y}_i. \tag{11}$$

4: compute matrices $\mathbf{T}_{11}, \mathbf{T}_{21}$, and $\mathbf{T}_{22}$ by the following QR-factorization:

$$\begin{bmatrix}\mathbf{T}_{11} & \mathbf{O}\\ \mathbf{T}_{21} & \mathbf{T}_{22}\end{bmatrix} = \text{qr}\left(\begin{bmatrix}\mathbf{Y} & \sqrt{\mathbf{R}}\\ \mathbf{N} & \mathbf{O}\end{bmatrix}\right), \tag{12}$$

where $\mathbf{O}$ denotes a zero matrix of appropriate size.

5: Estimate the cubature gain

$$\mathbf{W} = \mathbf{T}_{21}/\mathbf{T}_{11}, \tag{13}$$

where $/$ represents solving for $\mathbf{W}$ in $\mathbf{T}_{21} = \mathbf{W}\mathbf{T}_{11}$ using a backward stable solver.

6: Estimate the mean of the corrected state

$$\hat{\mathbf{x}} = \bar{\mathbf{x}} + \mathbf{W}(\mathbf{y} - \bar{\mathbf{y}}). \tag{14}$$

7: Estimate a factorization of the corrected covariance matrix

$$\hat{\mathbf{M}} = \mathbf{T}_{22}. \tag{15}$$

8: **return** Corrected mean $\hat{\mathbf{x}}$ and a factorization of the corrected covariance matrix $\hat{\mathbf{M}}$.

A. Preservation of Gaussian for a Local Linear Approximation

Without loss of generality (WLOG), we may assume the particle of concern is centered at $\mathbf{0}$. Moreover, since we are interested in how the dynamics and diffusion *deform* the distribution, we may also set the drift function at center $\mathbf{v}(\mathbf{0}) = \mathbf{0}$. With these simplifications in mind, the *original* Fokker-Planck equation can be restated as:

$$\frac{du}{dt} = \frac{1}{2}\nabla \cdot \mathbf{K}\nabla u - \nabla \cdot (\mathbf{v}u), \tag{16}$$

where $u = u(\mathbf{x}, t)$ is the PDF, $\mathbf{K}$ is a constant matrix-valued continuous Gaussian **process noise**, and $\mathbf{v} = \mathbf{v}(\mathbf{x})$ is the **drift velocity (field)**, and $\mathbf{v}(\mathbf{0}) = \mathbf{0}$ by our WLOG simplification.

Then, we approximate (16) by taking a linear approximation of $\mathbf{v}$: $\mathbf{v}(\mathbf{x}) \approx \mathbf{J}\mathbf{x}$. ($\mathbf{J}$ is the Jacobian matrix.) Then:

Claim 1: A Gaussian distribution is preserved by a Fokker-Planck equation with a linear drift function.

Stated explicitly: For the following equation:

$$\frac{du}{dt} = \frac{1}{2}\nabla \cdot \mathbf{K}\nabla u - \nabla \cdot (\mathbf{J}\mathbf{x}u), \tag{17}$$

if the initial condition $u(\mathbf{x}, 0)$ is given by a Gaussian function

$$u(\mathbf{x}, 0) = \frac{1}{\sqrt{(2\pi)^d \det(\mathbf{\Sigma})}} \exp\left(-\frac{\mathbf{x}^T \mathbf{\Sigma}^{-1}\mathbf{x}}{2}\right), \tag{18}$$

then $u(\mathbf{x}, t)$ is also a Gaussian distribution.

The rest of this subsection proves this claim.

We define the auxiliary function F by

$$F(\mathbf{x}, t) := \frac{u(\mathbf{x}, t)}{u(\mathbf{0}, t)}. \tag{19}$$

Consider a **level set** of the function F at t defined as

$$\mathcal{L}(t) := \left\{\mathbf{x} \in \mathbb{R}^d | F(\mathbf{x}, t) = c\right\}, \tag{20}$$

where $0 < c < 1$ is some fixed scalar constant. $\mathcal{L}(t)$ is (usually) a surface, and for a Gaussian as defined in $u(\mathbf{x}, 0)$, it is an ellipsoid. As the function F varies in time, the set $\mathcal{L}$ propagates in space. To describe the movement of the set $\mathcal{L}$, we consider the apparent velocity the traveling surface.

In particular, a **velocity of level set** $\mathbf{v}_{\mathcal{L}}$ is *defined* by a velocity field satisfying the **level-set equation**:

$$\frac{dF}{dt} + \mathbf{v}_{\mathcal{L}} \cdot \nabla F = 0. \tag{21}$$

(Note: this can be understood as the chain rule. For a more detailed explanation, refer to equations (1) and (2) in [12]. Also, note the **velocity of the level set** is uniquely defined up to tangential directions since tangential movements along the level set vanish since they preserve the level set.)

To proceed to the proof, we first consider the lemma:

Lemma 1: The velocity field

$$\mathbf{v}_L = \mathbf{J}\mathbf{x} + \frac{1}{2}\mathbf{K}\mathbf{\Sigma}^{-1}\mathbf{x} \tag{22}$$

is a velocity of the level set for $\mathcal{L}(0)$ defined in (20). Also, this velocity of the level set is linear in space.

Proof of Lemma 1: First, we note the velocity field is linear in space. To check that it is a velocity of the level set:

Since F is defined as a quotient of $u(\mathbf{x}, 0)$ and $u(\underline{\mathbf{0}}, 0)$, we may omit the normalizing factor in u, and take

$$u(\mathbf{x}, 0) = \exp\left(-\frac{\mathbf{x}^T \mathbf{\Sigma}^{-1}\mathbf{x}}{2}\right) \tag{23}$$

as the initial condition. Then:

$$\frac{dF}{dt}|_{t=0} = \frac{u'(\mathbf{x}, 0)u(\mathbf{0}, 0) - u'(\mathbf{0}, 0)u(\mathbf{x}, 0)}{u^2(\mathbf{0}, 0)}. \tag{24}$$

By (23), we note that $u(\mathbf{0}, 0) = 1$. We simplify (24) using this substitution:

$$\frac{dF}{dt}|_{t=0} = u'(\mathbf{x}, 0) - \exp\left(-\frac{1}{2}\mathbf{x}^T \mathbf{\Sigma}^{-1}\mathbf{x}\right) u'(\mathbf{0}, 0). \tag{25}$$

Substitute time derivatives u' with (17):

$$\begin{aligned}\frac{dF}{dt}|_{t=0} = &\left(\frac{1}{2}\nabla \cdot \mathbf{K}\nabla u - \nabla \cdot (\mathbf{J}\mathbf{x}u)\right)|_{t=0,\mathbf{x}=\mathbf{x}} \\ &- \exp(\dots)\left(\frac{1}{2}\nabla \cdot \mathbf{K}\nabla u - \nabla \cdot (\mathbf{J}\mathbf{x}u)\right)|_{t=0,\mathbf{x}=\mathbf{0}},\end{aligned} \tag{26}$$

where

$$\exp(\dots) := \exp\left(-\frac{1}{2}\mathbf{x}^T \mathbf{\Sigma}^{-1}\mathbf{x}\right). \tag{27}$$

continuing the computation, we find that:

$$\nabla \cdot \mathbf{K}\nabla u = \nabla \cdot \left(-\mathbf{K}\exp(\dots)\mathbf{\Sigma}^{-1}\mathbf{x}\right) \tag{28}$$

$$= \exp(\dots)\mathbf{\Sigma}^{-1}\mathbf{x} \cdot \mathbf{K}\mathbf{\Sigma}^{-1}\mathbf{x} - \exp(\dots)\ \mathrm{tr}\,(\mathbf{K}\mathbf{\Sigma}^{-1}) \tag{29}$$

$$= \exp(\dots)(\mathbf{\Sigma}^{-1}\mathbf{x} \cdot \mathbf{K}\mathbf{\Sigma}^{-1}\mathbf{x} - \ \mathrm{tr}\,(\mathbf{K}\mathbf{\Sigma}^{-1})). \tag{30}$$

And

$$\nabla \cdot ((\mathbf{J}\mathbf{x}u)) = \nabla u \cdot \mathbf{J}\mathbf{x} + u\nabla \cdot \mathbf{J}\mathbf{x} \tag{31}$$

$$= \exp(\dots)(-\mathbf{\Sigma}^{-1}\mathbf{x}) \cdot \mathbf{J}\mathbf{x} + \exp(\dots)\ \mathrm{tr}\,(\mathbf{J}) \tag{32}$$

$$= \exp(\dots)(\ \mathrm{tr}\,(\mathbf{J}) - \mathbf{\Sigma}^{-1}\mathbf{x} \cdot \mathbf{J}\mathbf{x}). \tag{33}$$

Substitute these two terms into (26), we have

$$\begin{aligned}\frac{dF}{dt} = &\exp(\dots)\left(\frac{1}{2}\mathbf{\Sigma}^{-1}\mathbf{x} \cdot \mathbf{K}\mathbf{\Sigma}^{-1}\mathbf{x} - \frac{1}{2}\ \mathrm{tr}\,(\mathbf{K}\mathbf{\Sigma}^{-1})\right. \\ &\left. +\ \mathrm{tr}\,(\mathbf{J}) - \mathbf{\Sigma}^{-1}\mathbf{x} \cdot \mathbf{J}\mathbf{x}\right) \\ &- \exp(\dots)\left(-\frac{1}{2}\ \mathrm{tr}\,(\mathbf{K}\mathbf{\Sigma}^{-1}) +\ \mathrm{tr}\,(\mathbf{J})\right)\end{aligned} \tag{34}$$

$$= \exp(-\frac{1}{2}\mathbf{x}^T \mathbf{\Sigma}^{-1}\mathbf{x})\left(\frac{1}{2}\mathbf{\Sigma}^{-1}\mathbf{x} \cdot \mathbf{K}\mathbf{\Sigma}^{-1}\mathbf{x} + \mathbf{\Sigma}^{-1}\mathbf{x} \cdot \mathbf{J}\mathbf{x}\right). \tag{35}$$

Meanwhile, we check that

$$\mathbf{v}_L \cdot \nabla F = \mathbf{v}_L \cdot \nabla\left(\exp\left(-\frac{\mathbf{x}^T \mathbf{\Sigma}^{-1}\mathbf{x}}{2}\right)\right) \tag{36}$$

$$= \mathbf{v}_L \cdot (-\mathbf{\Sigma}^{-1}\mathbf{x}\exp(-\frac{1}{2}\mathbf{x}^T \mathbf{\Sigma}^{-1}\mathbf{x})). \tag{37}$$

Substitute the level set velocity term $\mathbf{v}_L$ from (22), we get that

$$\mathbf{v}_L \cdot \nabla F = -\exp(-\frac{1}{2}\mathbf{x}^T \mathbf{\Sigma}^{-1}\mathbf{x})(\mathbf{J}\mathbf{x} + \frac{1}{2}\mathbf{K}\mathbf{\Sigma}^{-1}\mathbf{x}) \cdot \mathbf{\Sigma}^{-1}\mathbf{x}. \tag{38}$$

Comparing the results from (38) and (35), we conclude that

$$\frac{dF}{dt} + \mathbf{v}_L \cdot \nabla F = 0. \tag{39}$$

Therefore $\mathbf{v}_L$ is a velocity of the level set, as defined in (21).

We now proceed to the proof Claim 1: ■

Proof of Claim 1: Lemma 1 shows that every level set is propagated by a linear velocity field independent of the choice of level set (in other words, independent of the choice of c.) In particular, it is propagated by a linear transformation instantaneously. Consequently, between any fixed time 0 and Δt, the Gaussian is mapped by a linear transformation. Since a linear transformation maps Gaussians to Gaussians, the velocity field $\mathbf{v}_L$ is always well-defined as the covariance term in (22) is defined, whereas other terms are known.

Before we proceed, we should note that the claim is a corollary of equation (29) in [2] by Kalman and Bucy that started the discussion of continuous-discrete Kalman filtering. The significance of the proof is that by using a square-root factorization, the transformation is now given by an *explicit formula* that does not involve an integral. We also note that while the formula (22) is restricted to a normal distribution, tracking distribution by analyzing the propagation of level set as defined in (20) can potentially be applied to an α-stable distribution introduced in [13]. This gives a possibility to extend α-stable filtering methods [14], [15] to a continuous-discrete problem. ■

B. Deriving the Time-Update of the LSKF

We now describe the numerical algorithm inspired from the velocity of level set (22). Tracking the movement of the Gaussian is equivalent to tracking one of its ellipsoid level sets (as defined in (20)). If the mean of the Gaussian remains at $\mathbf{0}$, then a factorization of the covariance matrix $\mathbf{\Sigma} = \mathbf{M}\mathbf{M}^T$ can be used to represent the Gaussian. This factorization also represents the unique level set ellipsoid spanned by the columns of the factorization $\mathbf{M}$. More specifically, set

$$\mathbf{M}(0) = \begin{bmatrix} \mathbf{x}_1(0) & \cdots & \mathbf{x}_d(0) \end{bmatrix} \tag{40}$$

as initial conditions, and let $\mathbf{x}_i(t)$ be the solutions of (22). (One may interpret $\mathbf{x}_i(t)$ as a **point on the level set**, which travels at the apparent speed defined by a velocity of level set.)

Then $\mathbf{\Sigma}(t)$ defined as

$$\mathbf{\Sigma}(t) := \mathbf{M}(t)\mathbf{M}(t)^T \tag{41}$$

is the covariance matrix for the Gaussian at time t since it is a similarity transformation. Suppose A is the linear transformation from time 0 to t, then $\mathbf{x}_i(t) = \mathbf{A}\mathbf{x}_i(0)$, and

$$\begin{aligned} \mathbf{\Sigma}(t) &= \mathbf{M}(t)\mathbf{M}(t)^T \\ &= \mathbf{A}\mathbf{M}(0)\mathbf{M}(0)^T\mathbf{A} \\ &= \mathbf{A}\mathbf{\Sigma}(0)\mathbf{A}^T. \end{aligned}$$

Therefore applying linear transformation $\mathbf{A}$ to the ellipse is equivalent to applying it to all column vectors in $\mathbf{M}$.

The Jacobian of the velocity field $\mathbf{J}$ that appears in (22) is not explicitly needed. Instead of direct evaluation of the Jacobian, we approximate the effect of the drift velocity by applying a quadrature rule. In this section, we use the forward difference in space to derive the method. Specifically, we notice $\mathbf{\Sigma}^{-1} = \mathbf{M}^{-T}\mathbf{M}^{-1}$, (where $\mathbf{M}^{-T}$ indicates the inverse transpose) and the $\mathbf{x}_i$s are columns of $\mathbf{M}$. Therefore for all $\mathbf{x}_i$s on the level set ellipsoid, by substituting the terms in (22), we find the apparent velocity of the level set is approximated by

$$\frac{d\mathbf{x}_i}{dt} = \mathbf{v}(\bar{\mathbf{x}} + \mathbf{x}_i) - \mathbf{v}(\bar{\mathbf{x}}) + \frac{1}{2}\mathbf{K}(\mathbf{M}^T)^{-1}\mathbf{e}_i, \tag{42}$$

where $\bar{\mathbf{x}}$ is the **mean** of the Gaussian. In the case the Gaussian is centered at $\mathbf{0}$, $\bar{\mathbf{x}} = \mathbf{0}$. $\mathbf{e}_i$ is the ith unit vector with all entries 0 except that ith entry is 1.

Recall that the $\mathbf{x}_i$s are columns of $\mathbf{M}$. In a matrix short-hand (where the matrix-vector additions are defined entry-wise, and recall vectors are column vectors):

$$\frac{d\mathbf{M}}{dt} = \mathbf{v}(\bar{\mathbf{x}} + \mathbf{M}) - \mathbf{v}(\bar{\mathbf{x}}) + \frac{1}{2}\mathbf{K}(\mathbf{M}^T)^{-1}. \tag{43}$$

Whereas the velocity for center is given by

$$\frac{d\bar{\mathbf{x}}}{dt} = \mathbf{v}(\bar{\mathbf{x}}). \tag{44}$$

Based on the form of the equations, the assumptions $\bar{\mathbf{x}} = \mathbf{0}$ and $\mathbf{v}(\bar{\mathbf{x}}) = \mathbf{0}$ can be dropped.

Concatenating $\bar{\mathbf{x}}$ and $\mathbf{M}$ as a variable $(\bar{\mathbf{x}}|\mathbf{M})$ of dimension $d \times (d+1)$, we obtained a nonlinear ODE in this space. Any standard ODE solver can be applied to this ODE to complete the *time-update* between the measurements.

Note that to evaluate the velocity of one point $\mathbf{x}_i$ on the level set, both the mean $\bar{\mathbf{x}}$ and all other points $\mathbf{x}_j$ for this Gaussian kernel are needed, hence the points on a level set cannot be updated independently (in contrast to the time-update step for both the UKF and the CD-CKF). *This provides intuition about the difference between our method and others: while other methods looks at the past covariance information and rely on an expansion in time, our method uses only the current information about the covariance matrix. Except for the purpose of numerically solving the ODE, our method does not need time-discretization.*

C. Motivating Example: Linear Drift Function

As an illustration for the *time-update* method, we consider the following Fokker-Planck equation with a linear drift function:

$$\frac{du}{dt} = \nabla \cdot \mathbf{K}\nabla u - \nabla \cdot (\mathbf{J}\mathbf{x}u) \tag{45}$$

with parameters

$$\mathbf{K} = \begin{bmatrix} \frac{1}{2} & \frac{1}{4} \\ \frac{1}{4} & \frac{3}{2} \end{bmatrix} \quad \mathbf{J} = \begin{bmatrix} 0 & 0.1 \\ 0 & 0 \end{bmatrix}.$$

We consider the solution of the initial value problem with initial condition

$$u(\mathbf{x}, 0) = \frac{1}{\sqrt{(2\pi)^d \det(\mathbf{\Sigma}_0)}} \exp\left(-\frac{\mathbf{x}^T\mathbf{\Sigma}_0^{-1}\mathbf{x}}{2}\right), \tag{46}$$

where the initial covariance is given by

$$\mathbf{\Sigma}_0 = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}. \tag{47}$$

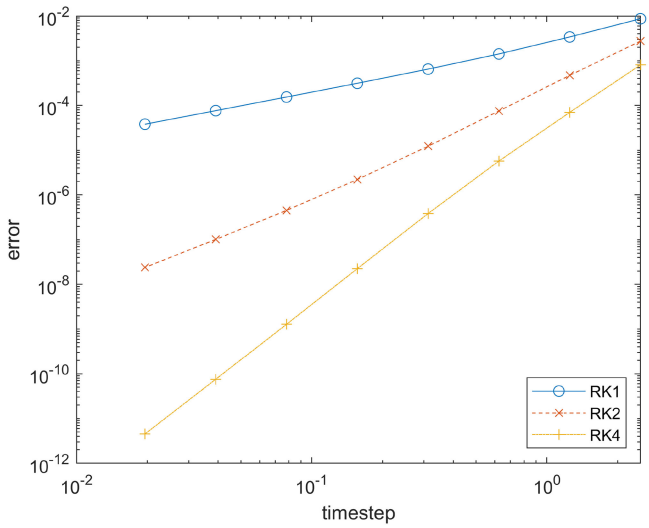

Fig. 1. Log-log graph of the error of covariance matrix in the infinity norm with linear Fokker-Planck equation. This shows that our method preserves the order of accuracy of the ODE solver.

In Section III, we proved that propagating level sets by (22), and consequently (43) exactly solves (45). To give a concrete numerical example of this property, we check the convergence of numerical ODE solvers for the initial value problem and find the error of the density function at $t = 10$ with ODE solvers of different order. We factor $\mathbf{\Sigma}_0 = \mathbf{M}_0\mathbf{M}_0^T$, and set the ODE (43) with the initial condition $\mathbf{M}(0) := \mathbf{M}_0$. To verify that our method is accurate for the linear Fokker-Planck equation (45), we check that when using different numerical ODE solvers, the solution converges to the same value, with the rate of convergence coinciding with the order of the ODE solver.

In Fig. 1, we verify that for the Runge-Kutta methods of order 1, 2, and 4, the error of $u(\cdot, 10)$ measured in infinity norm converges at the same order of the ODE solvers, which is expected if the reformulation (43) is exact for (45).

D. The Averaged Velocity Level Set Time-Update

Here we state the **averaged velocity level set time-update** method, which uses central difference instead of forward difference, and shows better accuracy in numerical experiments (See Appendix A for an example) when compared versus the version in Section III-B.

We set the velocity of the mean $(d\bar{\mathbf{x}})/dt$ by the **averaged velocity**:

$$\frac{d\bar{\mathbf{x}}}{dt} = \mathbf{v}_a(\bar{\mathbf{x}}, \mathbf{M}) := \frac{1}{2\,d}\sum_{i=1}^{d}\left(\mathbf{v}(\bar{\mathbf{x}} + \mathbf{x}_i) + \mathbf{v}(\bar{\mathbf{x}} - \mathbf{x}_i)\right). \quad (48)$$

(Recall that $\mathbf{x}_i$ are columns of the matrix $\mathbf{M}$) and the velocity of the matrix $\mathbf{M}$:

$$\frac{d\mathbf{M}}{dt} = \mathbf{v}(\bar{\mathbf{x}} + \mathbf{M}) - \mathbf{v_a} + \frac{1}{2}\mathbf{K}(\mathbf{M}^T)^{-1}. \quad (49)$$

It can be easily seen that when the drift velocity field $\mathbf{v}$ is linear in space, equations (49) and (43) are identical, and $\mathbf{v}_a(\bar{\mathbf{x}}, \mathbf{M}) = \mathbf{v}(\bar{\mathbf{x}})$.

Using this averaged velocity, here we summarize the LSKF:

Require: Guess of initial state $\hat{\mathbf{x}}_0$ at time t_0, and a factorization of a guess of covariance matrix $\hat{\mathbf{M}}_0$, measurements $\mathbf{y}_1 \ldots, \mathbf{y}_n$ at time $t_1, \ldots, t_n$.
Drift velocity $\mathbf{v}$, continuous process noise matrix $\mathbf{K}$, measurement function h, a factorization of the covariance matrix $\mathbf{R}$ of a zero-mean Gaussian measurement noise.

1: **for** j = 1,..., n **do**
2: Set the problem of $\bar{\mathbf{x}}(t)$ and $\mathbf{M}(t)$ given by:

$$\frac{d\bar{\mathbf{x}}}{dt} = \frac{1}{2\,d}\sum_{i=1}^{d}(\mathbf{v}(\bar{\mathbf{x}} + \mathbf{x}_i) + \mathbf{v}(\bar{\mathbf{x}} - \mathbf{x}_i)) \quad (50)$$

$$\frac{d\mathbf{M}}{dt} = \mathbf{v}(\bar{\mathbf{x}} + \mathbf{M}) - \mathbf{v_a} + \frac{1}{2}\mathbf{K}(\mathbf{M}^T)^{-1}, \quad (51)$$

with initial condition $\bar{\mathbf{x}}(t_{j-1}) = \hat{\mathbf{x}}_{j-1}$, and $\mathbf{M}(t_{j-1}) = \hat{\mathbf{M}}_{j-1}$ (Recall: $\mathbf{v_a}$ is defined in (48), also recall that $\mathbf{x}_i$ are columns of $\mathbf{M}$.)
3: *Time-update:* solve the above equation from t_{j-1} to t_j using a numerical ODE solver, and approximate $\bar{\mathbf{x}}_j = \bar{\mathbf{x}}(t_j)$, $\mathbf{M}_j = \mathbf{M}(t_j)$ with the numerical solution.
4: *Measurement-update:* Find the corrected mean $\hat{\mathbf{x}}_j$ and corrected covariance matrix $\hat{\mathbf{M}}_j$ at time t_j by applying the measurement-update algorithm defined in subsection II-D, with input $\bar{\mathbf{x}}_j$, $\mathbf{M}_j$, and $\mathbf{R}$.
5: **end for**
6: **return** Predicted corrected state $\hat{\mathbf{x}}_1, \hat{\mathbf{x}}_n$, at $t_1, \ldots, t_n$, with a factorization of the predicted corrected covariance matrix $\hat{\mathbf{M}}_1, \ldots, \hat{\mathbf{M}}_n$.

E. Comparing Convergence: Achieving Beyond IT-1.5 Without Explicit Higher Derivatives

In Section II-C, we introduced the CD-CKF with IT-1.5. Here, we compare the convergence rate of the time-update of the CD-CKF (as implemented in [6], and with proper IT-1.5) with that of the LSKF.

Consider a simple harmonic oscillator:

$$\mathbf{x}(t) := [\epsilon(t) \quad \dot{\epsilon}(t) \quad \ddot{\epsilon}(t)]^T, \quad (52)$$

where ϵ, $\dot{\epsilon}$, and $\ddot{\epsilon}$ are the position, velocity, and acceleration of the oscillator. Its time derivative is given by

$$\mathbf{v}(\mathbf{x}) = [\dot{\epsilon} \quad \ddot{\epsilon} \quad -\epsilon]^T. \quad (53)$$

This oscillator is also subject to a continuous process noise, defined by the diagonal diffusion matrix:

$$\mathbf{K} = \mathrm{diag}[0.01^2 \quad 0.01^2 \quad 0.02^2]. \quad (54)$$

Since the dynamics are linear, the Gaussian is preserved, and we expect the result from the LSKF and the proper IT-1.5 to converge to the exact solution.

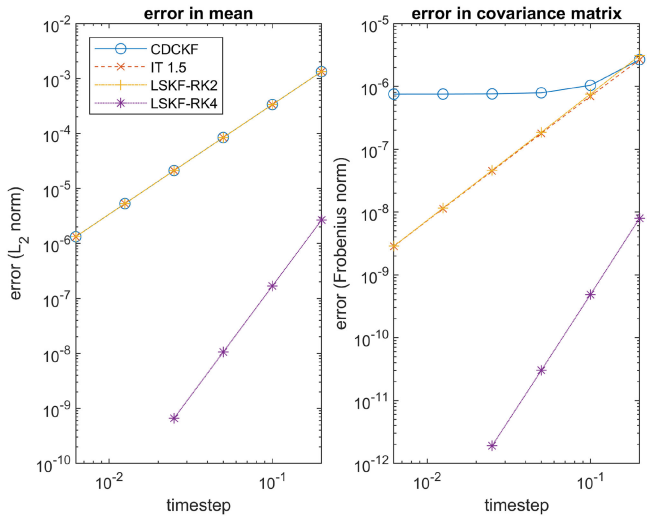

Fig. 2. Convergence of the mean $\mathbf{x}$ and covariance matrix with the CD-CKF and the LSKF. The left panel shows the error (measured in L_2 norm) in mean value as a function of the timestep, whereas the right panel shows error (measured in Frobenius norm) in the covariance matrix. In each panel, CD-CKF (blue) denotes the time-update implemented in [6], IT-1.5 (red) denotes the CD-CKF with the proper IT-1.5 expansion, LSKF-RK2 (yellow), and LSKF-RK4 (purple) denotes time-update of the LSKF with Runge-Kutta solvers of order 2 and 4 respectively. The Runge-Kutta 4 version is cut short due to finite precision linear algebra.

To find the order of convergence, and compare the methods, we consider the following initial condition problem. Given initial condition

$$\mathbf{x}(0) = [1 \quad 0 \quad 0]^T \tag{55}$$

and initial covariance matrix

$$\mathbf{\Sigma}(0) = \text{diag}[0.01^2 \quad 0.01^2 \quad 0.03^2], \tag{56}$$

we would like to find the end state at $t = 0.2$ using the above mentioned methods. By subdividing the timesteps, we arrive at the following convergence result: Similar to Fig. 1, Fig. 2 shows that the time-update of the LSKF converges the same order as the underlying ODE solver. Importantly, note the proper IT-1.5 and the LSKF-RK2 converge to the same limit mean and covariance matrix at a weak order of convergence 2, validating the correctness of both methods. (Note: they **do not** converge to the same square root of the covariance matrix, which is not surprising given that the matrix square roots are not unique.) The time-update of the CD-CKF as implemented in [6] does not converge to the same limit. Comparing the LSKF-RK4 versus the proper IT-1.5, it can be noted that much fewer timestep subdivisions can achieve similar truncation errors. As pointed out in [11], the IT-1.5 already requires explicit first and second derivatives of $\mathbf{v}(\mathbf{x})$, and any higher-order Ito-Taylor expansion is necessarily more complicated. On the contrary, the time-update of the LSKF, as defined in (49) does not require the explicit expression of the derivatives of $\mathbf{v}(\mathbf{x})$, and can achieve a higher order of convergence with the freedom to choose any ODE solver.

IV. Numerical Example: The Radar Tracking Coordinated Turn Test Case

A. Problem Description

Here, we follow the test case presented in [6], considering the scenario where a radar station tracks an aircraft making a coordinated turn. Since the CD-CKF in [6] is claimed to be *the choice for challenging radar problems*, we compare LSKF against CD-CKF in the most challenging scenario they considered with $\omega = 6^\circ/s$ with sampling intervals $T = 2\,s$, $T = 4\,s$, and $T = 6\,s$. (Note: conversion to radians per second is required) Additionally, we consider the more challenging scenario with $\omega = 12^\circ/s$ and $\omega = 24^\circ/s$. We implemented the CD-CKF based on the square root form formulated in [6], using their implementation presented at [16]. The details of the test case are as follows:

The aircraft is described by a 7-dimensional state vector

$$\mathbf{x}(t) := [\epsilon(t) \quad \dot{\epsilon}(t) \quad \eta(t) \quad \dot{\eta}(t) \quad \zeta(t) \quad \dot{\zeta}(t) \quad \omega(t)]^T, \tag{57}$$

where $\epsilon(t), \eta(t), \zeta(t)$ describes the position, in meters, $\dot{\epsilon}(t), \dot{\eta}(t), \dot{\zeta}(t)$ describes the velocity of the aircraft, in meters per second, and the $\omega(t)$ describes the turn rate of the aircraft, in **radians** per second. The dynamics of the aircraft are defined by the following drift equation:

$$\mathbf{v}(\mathbf{x}(t)) = [\dot{\epsilon} \quad -\omega\dot{\eta} \quad \dot{\eta} \quad \omega\dot{\epsilon} \quad \dot{\zeta} \quad 0 \quad 0]^T. \tag{58}$$

The noise term is defined by the following diagonal diffusion matrix:

$$\mathbf{K} = \text{diag}([0 \quad \sigma_1^2 \quad 0 \quad \sigma_1^2 \quad 0 \quad \sigma_1^2 \quad \sigma_2^2]), \tag{59}$$

where $\sigma_1 = \sqrt{0.2}$, and $\sigma_2 = 7 \times 10^{-4}$. (Note: in [6], they suggested $\sigma_2 = 7 \times 10^{-3}$. However, the accompanied code provided by Arasaratnam on his webpage [16] used the parameter $\sigma_2^2 = 5 \times 10^{-7}$, which matches closely with $\sigma_2 = 7 \times 10^{-4}$. Our calculated RMSE also turns to be similar as shown in Figs. 2, 3 and 4 in [6] if 7×10^{-4} is chosen, whereas $\sigma_2 = 7 \times 10^{-3}$ does not give similar results.)

The measurement is from a single radar station located at $\mathbf{s} = [1500 \quad 10 \quad 0]$. The radar station measures the distance r, azimuth angle θ and elevation angle ϕ relative to the radar station. The measurement function is therefore given by:

$$\begin{bmatrix} r \\ \theta \\ \phi \end{bmatrix} = \begin{bmatrix} \sqrt{(\epsilon - 1500)^2 + (\eta - 10)^2 + \zeta^2} \\ \arctan(\frac{\eta - 10}{\epsilon - 1500}) \\ \arctan\left(\frac{\zeta}{\sqrt{(\epsilon-1500)^2 + (\eta-10)^2}}\right) \end{bmatrix} + \mathbf{w}. \tag{60}$$

where the measurement noise $\boldsymbol{\tau} \sim \mathcal{N}(0, \mathbf{R})$, with measurement noise matrix $\mathbf{R} = \text{diag}([\sigma_r^2, \sigma_\theta^2, \sigma_\phi^2])$, where $\sigma_r = 50, \sigma_\theta = 0.1^\circ, \sigma_\phi = 0.1^\circ$ (Note: the standard deviations σ_θ and σ_ϕ are measured in **degrees**, and a unit conversion is needed).

For the test scenario, the aircraft starts with the initial state

$$\mathbf{x}_0 = [1000 \quad 0 \quad 2650 \quad 150 \quad 200 \quad 0 \quad \omega_0]^T, \tag{61}$$

where ω_0 is the initial turn rate, and the measurement is taken with a constant time interval T. The total time for simulation is chosen to be 120 seconds. The turn rate and measurement

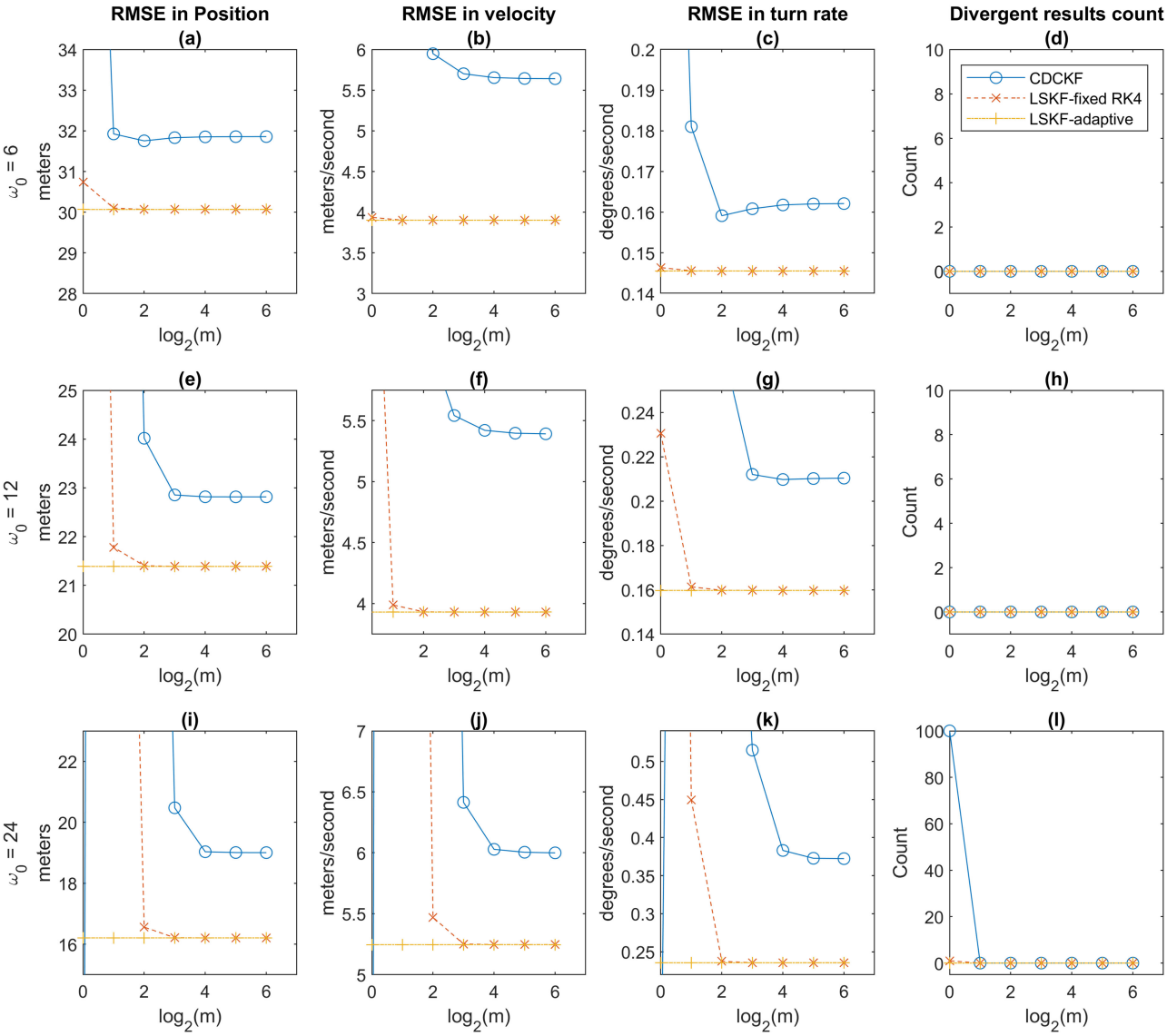

Fig. 3. RMSE and count of divergence results, for a fixed measurement interval $T = 6\,s$ varying m and ω_0, where m is the number of timestep subdivisions between measurements, and ω_0 is the initial turn rate. Each row gives the performance metrics for the same initial turn rate, whereas each column contrasts the same performance measurement across different initual turn rates. Note for $\omega_0 = 24°/s$ and $m = 1$, all results from the CD-CKF are divergent.

interval vary across test cases to examine the performance of the filters. The initial covariance is taken as

$$\mathbf{\Sigma} = \text{diag}([100 \quad 1 \quad 100 \quad 1 \quad 100 \quad 1 \quad 0.01]), \tag{62}$$

based on a physically realistic assumption: from an observer on the ground, one would have a reasonably good guess about its position with standard deviation $\sigma = 10$ meters, and a good guess about its velocity through differentiation with $\sigma = 1$ meters per second, but a rather bad guess for the turn rate with $\sigma = 0.1$ radian per second, or approximately 5.73 degrees per second.

B. Numerical Results

With the problem description complete, we now turn to present our numerical results. $N = 100$ experiments are executed for each set of parameters, and the same set of experiments is applied to all candidate filters. The main performance metric used is the Root-mean square error (RMSE) for position, velocity and turn rate. For example, RMSE for position is defined as:

$$\sqrt{\frac{1}{NK}\sum_{n=1}^{N}\sum_{k=1}^{K}\left((\epsilon_k^n - \hat{\epsilon}_k^n)^2 + (\eta_k^n - \hat{\eta}_k^n)^2 + (\zeta_k^n - \hat{\zeta}_k^n)^2\right)}, \tag{63}$$

where N is the number of experiments, and K is the number of measurements in each experiment.

Another metric we consider is the number of divergent results, which we define as any result that has an error larger than 500 or ends prematurely due to a `not a number` error. Following [6], we evaluate the performance of each method

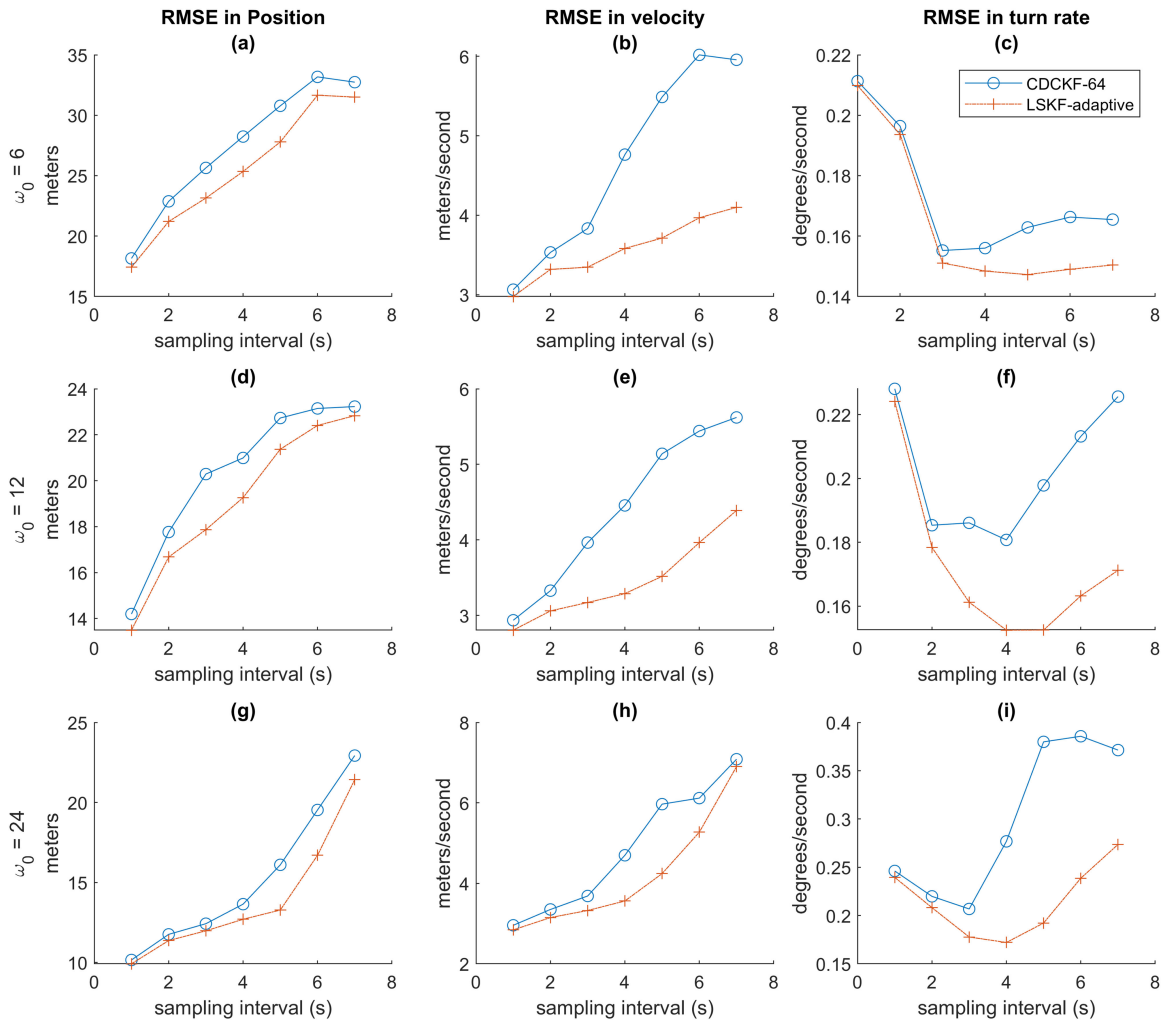

Fig. 4. RMSE with varying the measurement interval from 1 through 7 seconds, and varying the initial turn rate ω_0. For the CD-CKF, a timestep subdivision of $m = 64$ is chosen to ensure that the CD-CKF is performing optimally. For the LSKF, an adaptive ODE solver is used and no timestep subdivisions are manually inserted. All results from the CD-CKF and the LSKF are convergent.

at different subdivisions, m, of the timestep between measurements. Since our method is defined purely as a reformulated ODE, the subdivision of the timestep is the same as a timestep in a fixed timestep ODE solver, such as the widely-used Runge-Kutta 4 method. Additionally, to verify our claim that the choice of timestep subdivision can be completely passed to the ODE solver, we also use an adaptive solver, `ode113`, which is integrated into the `MATLAB` software package. In this sense, we introduced 2 implementations of the LSKF, which we call the *LKSF-RK4* and *LSKF-adaptive* respectively. In the following examples, we will verify that the additional subdivisions do not affect the numerical results from the LSKF-adaptive.

Our numerical results in Fig. 3 show that our methods consistently outperform the CD-CKF in this test case, across all choices of angular velocity and timestep subdivisions. Importantly, note that the performance of the CD-CKF cannot match that of the LSKF-adaptive even if sufficient timestep subdivisions are introduced. We suspect this is due to the fact that the CD-CKF as introduced in [6] only uses the IT-1.5 expansion at the beginning of each *time-update* step but not at the subdivided timesteps, whereas our method is defined using instantaneous information and is not subject to this limitation.

Equally importantly, note that the LSKF-adaptive version of our method gives the same result independent of the subdivisions introduced. Additionally, the fixed-timestep LSKF-RK4 converges to the LSKF-adaptive result, as expected for a consistent ODE solver. In practice, a user can always use the LSKF-adaptive version with the choice of adaptive ODE solvers that gives the best performance without needing to consider timestep subdivisions manually. In Fig. 4, we verify that even with sufficient timestep subdivision for the CD-CKF, the LSKF still outperforms the CD-CKF over all the parameters chosen, even when no intermediate timesteps are manually inserted. Additionally, the difference in performance between the CD-CKF and the LSKF is more significant when the measurement interval T is large, whereas the results are similar when $T = 1\,s$. With this in mind, we proceed further into the numerical experiments, using sufficient timestep subdivisions ($m = 64$) for the CD-CKF, whereas no additional pre-defined timestep subdivisions

($m = 1$) for the LSKF-adaptive, and vary the measurement interval T from 1 s to 7 seconds with an increment of 1 s.

In conclusion, for the test case picked by [6], our LSKF method consistently outperforms the CD-CKF across the challenging scenarios introduced by them. In addition, our method requires less input from an end-user, as our method only requires knowledge of the *drift function* explicitly, whereas the CD-CKF also requires the first and second spatial derivatives of the *drift function*, as well as a user-defined timestep subdivision parameter m. Finally, the elegance of reforming the system as an ODE without introducing expansion in time gives more room for possible future improvement.

V. Conclusion

In this paper, we derived a novel Level Set Kalman Filter method for nonlinear continuous-discrete systems. From a theory standpoint, our derivation is based on the movement of a level set instead of the moments of a distribution. Our method reformulates the *time-update* of the filtering as an ODE. As a consequence of this formulation, our description is instantaneous, in contrast to existing methods that use some expansion in time to approximate the continuous process noise.

From a practical point, for the radar tracking coordinated turn test case, our method consistently outperforms the CD-CKF over a range of challenging scenarios. Additionally, our method requires less explicit information about the model, and our instantaneous formulation allows a user to easily pass the task of choosing a timestep to the well-established field of adaptive ODE solvers. The numerical results indicate that our method is a good candidate for challenging tracking problems, especially if an appropriate timestep cannot be determined *a priori*.

Appendix A
Comparison of the standard, averaged and partially averaged time-update of the LSKF

In equations (43) and (49), we defined the **(standard)** *time-update* and the **averaged velocity** *time-update* equations. Here we use a numerical example to illustrate the difference in behavior between these methods. Additionally, we introduce the **partially averaged velocity** *time-update* equations as a trade-off option between the standard and the averaged velocity version. Using the same notations as in (43) and (49), the **partially averaged velocity** is defined as

$$\mathbf{v_p} := \frac{1}{2\,d}\left(\sum_{i=1}^{d}(\mathbf{v}(\mathbf{x}+\mathbf{x}_i)) + d \times \mathbf{v}(\mathbf{x}-\mathbf{x}_1)\right). \tag{64}$$

Correspondingly, in matrix form, the *time-update* ODE using partially averaged velocity is defined by

$$\frac{d\mathbf{M}}{dt} = \mathbf{v}(\mathbf{x}+\mathbf{M}) - \mathbf{v_a} + \frac{1}{2}\mathbf{K}(\mathbf{M}^T)^{-1}. \tag{65}$$

As an illustrative example to show the effects of using an *averaged velocity* or *partially averaged velocity* versus evaluating the velocity at the center when a nonlinear drift velocity is present, we consider the following system:

$$\frac{du}{dt} = -\nabla\cdot(\mathbf{v}u),$$

where a, b are positive constants, and

$$\mathbf{v}(x,y,0) = (0, x^2)$$
$$u(x,y,0) = \frac{1}{2\pi ab}\exp\left(-\frac{\frac{x^2}{a^2}+\frac{y^2}{b^2}}{2}\right).$$

(Note x, y in the following equation are not in bold, and are scalars) The analytic solution to this transport equation is given by

$$u(x,y,0) = \frac{1}{2\pi ab}\exp\left(-\frac{\frac{x^2}{a^2}+\frac{(y-xt)^2}{b^2}}{2}\right).$$

To compare the performance of the three LSKF methods, we compute the averaged $\mathbf{L^2}$ error of the results with the analytical result, with randomly chosen matrix square root $\mathbf{MM}^T = \mathbf{\Sigma}$. As can be observed from Fig. 5, the partially averaged velocity and averaged velocity version has less RMSE than the standard method. The averaged velocity method is less sensitive than the partially averaged method in that the covariance entries are less dependent on the choice of matrix square root. However, since the partially averaged method requires $d+1$ evaluations of the drift velocity whereas averaged method requires $2\,d$ evaluations, the partially averaged method is still useful as it can be considered as an efficient improvement from the standard version.

Appendix B
Computational cost of the LSKF

A count of FLOPs would be misleading for this method, as the main function reformulates the function as an ordinary differential equation, and the number of steps used is highly dependent on the choice of the numerical ODE solver and the numerical properties of the problem when an adaptive solver is used. When using an ODE solver, most of the computational cost is associated with evaluating the derivative. Therefore, we find the number of evaluation of the **drift velocity** $\mathbf{v}$, and FLOPs needed for a single derivative evaluation in (49). The computation cost for the *time-update* is then mainly decided by the number of derivative evaluations needed and the length of the timestep. The *measurement-update* is identical to that of the CD-CKF, which is listed in Table V of [6] and is omitted here.

Computations needed for (49) are:

1) Evaluate the **averaged velocity**:
$$\frac{d\bar{\mathbf{x}}}{dt} = \mathbf{v}_a(\bar{\mathbf{x}}, \mathbf{M}) := \frac{1}{2\,d}\sum_{i=1}^{d}(\mathbf{v}(\bar{\mathbf{x}}+\mathbf{x}_i) + \mathbf{v}(\bar{\mathbf{x}}-\mathbf{x}_i)). \tag{66}$$
$2\,d$ **drift velocity** evaluations and $O(d^2)$ FLOPs.
2) Find $\mathbf{K}(\mathbf{M}^T)^{-1}$: if solved by LU factorization, forward substitution, and back substitution: $\frac{8}{3}d^3 + O(d^2)$ FLOPs.
3) Evaluate (49): $O(d^2)$ FLOPs.

In total, $2\,d$ **drift velocity** evaluations and $\frac{8}{3}d^3 + O(d^2)$ FLOPs are needed for evaluating (49). When using a fixed

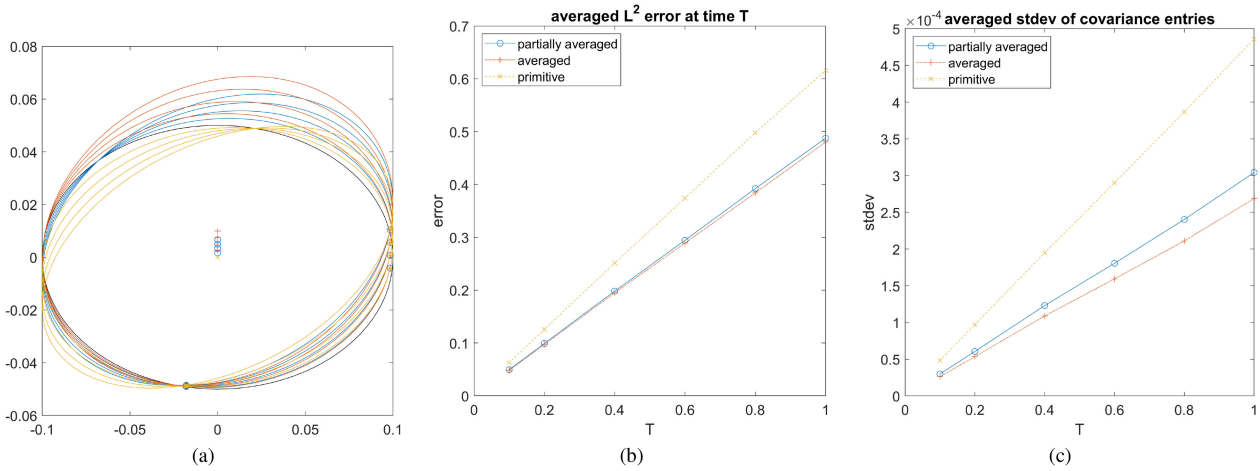

Fig. 5. A comparison between the partially averaged (blue), averaged (red), and primitive (yellow) LSKF applied to a nonlinear system. (a) shows the trajectories of the center and level set. (b) shows the averaged $\mathbf{L^2}$-norm of error over 1024 trials. (c) shows the averaged standard deviation of the covariance matrix.

Runge-Kutta 4 method, 4 such evaluations are needed per timestep, which results in $8\,d$ **drift velocity** evaluations and $\frac{32}{3}d^3 + O(d^2)$ FLOPs per *time-update*. If the measurement interval is short, then a fixed Runge-Kutta 2 method can be used, with half evaluations needed. For most scenarios, we suggest using an adaptive solver to automatically determine the appropriate timestep given a target error bound.

Appendix C
List of Symbols and Notations

All symbols and notations used in more than one locations are listed in Table I.

TABLE I
List of Symbols and Notations

τ	measurement noise
$\mathcal{N}(\mu, \mathbf{R})$	multivariate normal distribution with mean μ and covariance $\mathbf{R}$
$\mathbf{R}$	covariance of the measurement noise
d	dimension of the state vector
$\mathbf{x}$	state vector
$\mathbf{y}$	measurement vector
$h(\cdot)$	measurement function
$\mathbf{v}$	the (drift) velocity of the system dynamics
Δt	the timestep taken with the *time-update* step of the CD-CKF and the LSKF
$\mathbf{f}_d(\mathbf{x}, t)$	a function associated with the *time-update* step of the CD-CKF with a timestep of Δt
$\mathbf{K}$	$d \times d$ covariance matrix of the continuous process noise
$\sqrt{\mathbf{K}}$	A non-unique matrix square root such that $\mathbf{K} = \sqrt{\mathbf{K}}\sqrt{\mathbf{K}}^T$
$u = u(\mathbf{x}, t)$	the probability density function in state space
x_i	ith component of the state variable $\mathbf{x}$
$\mathbf{x}_0$	the initial condition for the mean of the estimation
$\mathbf{\Sigma}_0$	the initial condition for the covariance of the estimation
$\mathbf{x}_i$	ith cubature or off-center points in the state space ($i = 1..2d$)
$\mathbf{\Sigma}$	the covariance matrix in the estimation
$\mathbf{M}$	a square root factorization of the covariance matrix $\mathbf{\Sigma}$
$\mathbf{J}$	Jacobian matrix of the drift velocity field $\mathbf{v}$
$\bar{\mathbf{x}}$	the mean value in an estimation, during *time-update*
$\hat{\mathbf{x}}$	the mean value in an estimation, after *measurement-update*

Acknowledgment

D. B. Forger is the CSO and holds equity in Arcascope. The authors especially thank the anonymous reviewers and the associated editor, whose advice helped to improve the numerical stability of the method, in addition to the general improvement in the structure of the manuscript.

References

[1] S. Särkkä, *Bayesian Filtering and Smoothing*, vol. 3. Cambridge, U. K.: Cambridge Univ. Press, 2013.
[2] R. E. Kalman and R. S. Bucy, "New results in linear filtering and prediction theory," *J. Basic Eng.*, vol. 83, pp. 95–108, 1961.
[3] F. Gustafsson and G. Hendeby, "Some relations between extended and unscented Kalman filters," *IEEE Trans. Signal Process.*, vol. 60, no. 2, pp. 545–555, Feb. 2011.
[4] S. J. Julier and J. K. Uhlmann, "Unscented filtering and nonlinear estimation," *Proc. IEEE*, vol. 92, no. 3, pp. 401–422, Mar. 2004.
[5] S. Särkkä, "On unscented Kalman filtering for state estimation of continuous-time nonlinear systems," *IEEE Trans. Autom. Control*, vol. 52, no. 9, pp. 1631–1641, Sep. 2007.
[6] I. Arasaratnam, S. Haykin, and T. R. Hurd, "Cubature Kalman filtering for continuous-discrete systems: Theory and simulations," *IEEE Trans. Signal Process.*, vol. 58, no. 10, pp. 4977–4993, Oct. 2010.
[7] I. Arasaratnam and S. Haykin, "Cubature Kalman filters," *IEEE Trans. Autom. Control*, vol. 54, no. 6, pp. 1254–1269, Jun. 2009.
[8] G. Y. Kulikov and M. V. Kulikova, "Accurate continuous-discrete unscented Kalman filtering for estimation of nonlinear continuous-time stochastic models in radar tracking," *Signal Process.*, vol. 139, pp. 25–35, 2017.
[9] N. J. Newton, "Asymptotically efficient Runge-Kutta methods for a class of Ito and Stratonovich equations," *SIAM J. Appl. Math.*, vol. 51, no. 2, pp. 542–567, 1991. [Online]. Available: https://doi.org/10.1137/0151028
[10] A. H. Jazwinski, *Stochastic Processes and Filtering Theory*. North Chelmsford, MA, USA: Courier Corporation, 2007.
[11] S. Särkkä and A. Solin, "On continuous-discrete cubature Kalman filtering," *IFAC Proc. Volumes*, vol. 45, no. 16, pp. 1221–1226, 2012.
[12] J. A. Sethian, "Curvature and the evolution of fronts," *Commun. Math. Phys.*, vol. 101, no. 4, pp. 487–499, 1985.
[13] G. Samorodnitsky, M. S. Taqqu, and R. Linde, "Stable non-Gaussian random processes: Stochastic models with infinite variance," *Bull. London Math. Soc.*, vol. 28, no. 134, pp. 554–555, 1996.
[14] S. Leglaive, U. Şimşekli, A. Liutkus, R. Badeau, and G. Richard, "Alpha-stable multichannel audio source separation," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process*. 2017, pp. 576–580.

[15] S. P. Talebi, S. Werner, and D. P. Mandic, "Distributed adaptive filtering of α-stable signals," *IEEE Signal Process. Lett.*, vol. 25, no. 10, pp. 1450–1454, Oct. 2018.
[16] I. Arasaratnam, "Square-root cubature information filter, Accessed: Jan. 28, 2021. [Online]. Available: https://haranarasaratnam.com/docs/M_file_CDCKF.zip

Ningyuan Wang received the B.S. degree in mathematics from the Chinese University of Hong Kong, Hong Kong, in 2015, the Ph.D. degree in applied and interdisciplinary mathematics from the University of Michigan, Ann Arbor, MI, USA. He is currently a Project Assistant Professor with the University of Tokyo, Tokyo, Japan. His research interests include statistical differential equations, scientific computing, and modeling of sleep.

Daniel B. Forger received the B.S. and M.S. degrees in applied mathematics from Harvard College, Cambridge, MA, USA, in 1999, and the Ph.D. degree from Courant Institute, New York University, New York, NY, USA, in 2003. He is currently the Robert W. and Lynn H. Browne Professor of Science, a Professor of Mathematics, and a Research Professor of Computational Medicine and Bioinformatics with the University of Michigan, Ann Arbor, MI, USA. His research interests include biological timekeeping, nonlinear dynamics, sleep, and wearable research.