DGLG: A Novel Deep Generalized Legendre–Galerkin Approach to Optimal Filtering Problem

Ji Shi, *Member, IEEE*, Xiaopei Jiao, and Stephen S.-T. Yau, *Life Fellow, IEEE*

***Abstract*—The optimal filtering problem for general nonlinear and continuous state-observation systems attracts lots of attention in the control theory. The essence of optimal filtering requires solving the Duncan–Mortensen–Zakai (DMZ) equation in a computationally feasible way. Under the pioneering work of Yau-Yau filtering, the DMZ equation is reduced to a pathwise computation of a forward Kolmogorov equation with time-varying initial conditions, which is very challenging. To overcome the computational difficulty, in this article, we proposed a new efficient filtering algorithm consisting of a forward Kolmogorov equation solver based on a physics-informed neural network and a probability density approximator based on generalized Legendre polynomials. By utilizing the advanced deep learning method and classical Galerkin approximation, our developed algorithm not only maintains the high accuracy of the spectral method but also removes massive computational loads in the offline part. Furthermore, the convergence of our method is proved. Numerical experiments have been carried out to verify the feasibility of the new method. Regarding accuracy and efficacy, the newly proposed deep generalized Legendre–Galerkin algorithm outperforms other popular suboptimal methods including the extended Kalman filter and particle filter.**

***Index Terms*—Estimation, filtering, generalized Legendre polynomial (GLP), neural networks, nonlinear systems.**

I. Introduction

Nonlinear filtering (NLF) widely arises in many important practical applications, such as target tracking [8], [24], [40], navigation systems [15], [23], robotics [5], [29], and advanced control systems [9], [30]. For general NLF problems, the well-known Duncan–Mortensen–Zakai (DMZ) equation [11], [26], [39] describes the update equation for the unnormalized conditional density. In general, the conditional density cannot be characterized by a finite number of sufficient statistics, i.e., it lives in an infinite-dimensional space. Only in several special cases, such as linear Gaussian filter [18], [19], and Beneš filter [3], [7], solution of DMZ equation can be solved in an explicit form. In the majority of general systems, it is very difficult to solve the DMZ equation straightforwardly.

Received 12 July 2024; revised 13 July 2024; accepted 20 October 2024. Date of publication 25 October 2024; date of current version 31 March 2025. This work was supported in part by the National Natural Science Foundation of China (NSFC) under Grant 12101426 and Grant 11961141005, and in part by Tsinghua University Education Foundation fund under Grant 042202008. Recommended by Associate Editor A. Tanwani. *(Ji Shi and Xiaopei Jiao contributed equally to this work.) (Corresponding author: Stephen S.-T. Yau)*

Ji Shi is with the Academy for Multidisciplinary Studies, Capital Normal University, Beijing 100048, China (e-mail: shiji@cnu.edu.cn).

Xiaopei Jiao is with the Department of Applied Mathematics, University of Twente, Enschede, The Netherlands (e-mail: jack.jiao@utwente.nl).

Stephen S.-T. Yau is with the Beijing Institute of Mathematical Sciences and Applications, Beijing 101400, China, and also with the Department of Mathematical Sciences, Tsinghua University, Beijing 100084, China (e-mail: yau@uic.edu).

Digital Object Identifier 10.1109/TAC.2024.3486650

Due to the difficulty of solving DMZ equation directly, many researchers are devoted to find its approximate solution by various means. In summary, there are three classes of filtering algorithms. The first class is Kalman filter (KF)-based filtering algorithms, e.g., the well-known extended Kalman filter(EKF) [1], the unscented KF [17], ensemble KF [13], etc [2], [4], [16]. For these filters, they all assume that the probability distributions involved should be Gaussian. When the underlying filtering system has a strong nonlinearity, the filters can easily diverge. There have also appeared some new filtering algorithms in recent years combining traditional KF and its extensions with deep learning (DL) methods, e.g., [12], [21], [28]. The second class is particle filter(PF) [14]-based filtering algorithms, including various PF variants, feedback PF [35], etc. The PF imposes no assumption on the distribution of the filtering system, which makes it a universal filter. However, the particles generated by sequential Monte Carlo sampling suffer from "particle degeneracy," thereby leading to the failure of the filter sometimes. The third class aims to solve the DMZ equation directly. The cost of this class of methods is quite large yet the optimality of the solution obtained is guaranteed theoretically.

Starting from the first principles, the authors in [36] and [37] developed the Yau–Yau filtering framework for a general class of NLF system. This framework has several advantages. First, its convergence is theoretically guaranteed, under mild conditions. Second, it is real-time and memoryless. By "memoryless" we mean a filtering method that only uses each newly arrived observation to update the estimation of the system's states. Several algorithms have been developed under this framework. In [10] and [25], different spectral methods (SMs) were applied to the NLF problem. In [32], an efficient algorithm was developed from the optimization point of view. The core challenge of this framework lies in solving a parabolic partial differential equation (PDE) with time-varying initial conditions efficiently. Due to the inherent difficulties of this framework, at present, there is still not much research work in this direction.

In recent years, DL method has emerged as a promising alternative to solve PDE numerically. Notably, Raissi et al. [27] proposed the physics-informed neural network (PINN) method in for solving various PDE problems. Lots of work has been seen following this new computation paradigm in the field of numerical computation [20]. By enforcing the physical laws as optimization constraints, the PINN method is trained in an unsupervised way. Compared to classical numerical schemes, this method is very simple, and mesh-free. Therefore, the PINN method sheds light on overcoming the difficulty of the Yau–Yau framework.

Motivated by the Legendre–Galerkin method and the sophisticated DL method nowadays, we will develop an efficient filtering algorithm under the Yau–Yau framework in this article. The convergence of our method is analyzed in detail as stated in Theorem 3.1. The proposed algorithm is efficient and can be implemented in a real-time and memoryless manner. The numerical experiments verified the effectiveness of the proposed method.

The rest of this article is organized as follows. Section II briefly describes the NLF problem we considered and notations. The deep

generalized Legendre–Galerkin algorithm is derived in Section III. In Section IV, several numerical examples were carried out to verify the feasibility and effectiveness of the method. Finally, Section V concludes this article.

II. Basic Concepts and Preliminaries

In this article, we consider the following continuous-type NLF system:

$$\begin{cases} dx_t = f(x_t)dt + \Gamma w_t \\ dy_t = h(x_t)dt + dv_t. \end{cases} \tag{1}$$

Here, $x_t := x(t) \in \mathbb{R}^n$ is the state of the system at time t, $y_t := y(t) \in \mathbb{R}^m$ is the observation with $y_0 = 0$, $f(x_t)$, $h(x_t)$ are $C^\infty(\mathbb{R}^n)$ vector value functions. w_t, v_t are independent Brownian motion processes with variance Q and S, respectively. $\Gamma \in \mathbb{R}^{n\times p}$ is a constant diffusion coefficient matrix such that $G := \Gamma Q \Gamma$ is a positive definite matrix. $\{w_t\}_{t\geq 0}$, $\{v_t\}_{t\geq 0}$ and initial state x_0 are mutually independent.

Given all observations till instant t, i.e., $\mathcal{Y}_t := \{y_s : 0 \leq s \leq t\}$, it is well-known that in minimum variance sense, the conditional probability density function (PDF) $p(x,t)$ of x_t satisfies the Kushner–Stratonovich (K-S) equation [22], [34], which is computationally intractable. Later on, Duncan et al. [11], [26], [39] proposed independently the DMZ equation associated with an unnormalized PDF $\sigma(x,t)$

$$\begin{cases} d\sigma(x,t) = \mathcal{L}\sigma(x,t)dt + \sigma(x,t)h(x)^\top S^{-1} dy_t \\ \sigma(x,0) = \sigma_0(x). \end{cases} \tag{2}$$

Here, the operator $\mathcal{L}(*)$ is defined as follows:

$$\mathcal{L}(*) := \frac{1}{2}\sum_{i,j=1}^{n} G_{ij}\frac{\partial^2(*)}{\partial x_i \partial x_j} - \sum_{i=1}^{n}\frac{\partial(f_i *)}{\partial x_i} \tag{3}$$

and $\sigma_0(x)$ is an unnormalized version density function of the initial state x_0. Compared to K-S equation, this equation is easier to handle, since it is a linear stochastic PDE about $\sigma(x,t)$. It will be our main concern hereinafter.

Assumptions: We assume that the following holds.

1) The conditions of Theorem C and (A.2), (A.17), (C.1)–(C.3) in [36] hold.
2) The assumptions of [33, Thm. 3.6] hold.
3) Ω [in (9)] is a bounded domain in $\mathbb{R}^n$ with smooth boundary.

Notations: $L^2(\Omega)$ and $C^\infty(\Omega)$ denote the set of square integrable and smooth functions in a domain Ω, respectively. Inner product between two functions f, g is represented by $\langle f, g\rangle := \int_\Omega fg dx$. ∇ denotes the gradient operator. Δ denotes the Laplacian operator. $\mathcal{N}(\mu, \Sigma)$ denotes a Gaussian distribution with mean μ and variance Σ.

III. Numerical Solution of DMZ Equation Based on Deep Neural Network

In this section, we will develop a new filtering algorithm based on a deep forward Kolmogorov equation (FKE) solver with the generalized Legendre–Galerkin approximation. The convergence of our method is proved in the end.

A. Reduction of DMZ Equation to FKE

Note in the DMZ equation, the observation term dy_t will render underlying filtering algorithms lacking robustness. By making the following gauge transformation:

$$u(x,t) = \exp(-h(x)^\top(x) S^{-1} y_t)\sigma(x,t) \tag{4}$$

the DMZ equation is transformed into a deterministic PDE with stochastic coefficients

$$\begin{cases} \frac{\partial u}{\partial t}(x,t) = \exp(-h(x)^\top S^{-1} y_t)\left(\mathcal{L} - \frac{1}{2}h(x)^\top S^{-1}h(x)\right) \\ \qquad\qquad \cdot \left[\exp(h(x)^\top S^{-1} y_t)u(x,t)\right] \\ u(x,0) = \sigma_0(x). \end{cases} \tag{5}$$

Equation (5) is called the "pathwise-robust" DMZ equation. Generally speaking, the equation (5) does not have a closed-form solution, thus usually we seek efficient algorithms to construct a good approximation solution.

Let us assume the total filtering time is T. The observations occur at time sequence $P_N = \{0 = \tau_0 < \tau_1 < \cdots < \tau_N = T\}$, $\Delta\tau = \tau_i - \tau_{i-1} = T_0$. Let u_i be the solution of the robust DMZ equation (5) with observation process fixed on the interval $\tau_{i-1} \leq t < \tau_i$ by $y_t = y_{\tau_{i-1}}$, i.e.,

$$\begin{cases} \frac{\partial u_i}{\partial t}(x,t) = \exp(-h(x)^\top S^{-1} y_{\tau_{i-1}})\left(\mathcal{L} - \frac{1}{2}h^\top S^{-1}h\right) \\ \qquad\qquad \cdot \left[\exp(h(x)^\top S^{-1} y_{\tau_{i-1}})u_i(x,t)\right] \\ u_1(x,0) = \sigma_0(x) \\ u_i(x,\tau_{i-1}) = u_{i-1}(x,\tau_{i-1}), \text{ for } i \geq 2. \end{cases} \tag{6}$$

Note the observations y_{τ_i} are contained in the coefficients of (6), which brings a fundamental challenge for real-time computation. By the following proposition in [36], the equation is transformed into an observation-independent PDE.

Proposition 3.1: For each $\tau_{i-1} \leq t < \tau_i, i = 1, 2, \ldots, N$, $u_i(x,t)$ satisfies (6) if and only if

$$\rho_i(x,t) = \exp\left(h(x)^\top S^{-1} y_{\tau_{i-1}}\right) u_i(x,t) \tag{7}$$

satisfies the following FKE:

$$\frac{\partial \rho_i}{\partial t}(x,t) = \left(\mathcal{L} - \frac{1}{2}h(x)^\top S^{-1}h(x)\right)\rho_i(x,t) \tag{8}$$

where $\mathcal{L}$ is defined in (3).

Equation (8) is linear, and it does not depend on the information of the observations $\{y_{\tau_i}\}_{i=1}^N$. This property enables us to solve the FKE (8) in advance. Of each time interval $[\tau_{i-1}, \tau_i)$, the initial distribution varies. By projecting the initial condition $\rho(x, \tau_{i-1})$ onto a finite dimension subspace of $L^2(\mathbb{R}^n)$ once a new observation arrives, the station estimation can be implemented in real-time.

Remark 3.1: In practice, we always do filtering estimation in a finite time T, hence the states are in a bounded domain $\Omega = [a,b]^n$ over the entire filtering time interval. Therefore, it is reasonable to assume the state density $\rho(x,t)$ is supported on a considered bounded domain $\Omega = [a,b]^n$. Without loss of generality, we only consider one dimension case, i.e., $n = 1$ in the following.

B. Deep FKE Solver and Filtering Algorithm

In this part, we shall develop a new efficient algorithm to compute the FKE equation (8) with time-varying initial conditions by a deep neural network. For notation convenience, we omit the subscript in (8) as follows:

$$\begin{cases} \frac{\partial \rho}{\partial t}(x,t) = (\mathcal{L} - \frac{1}{2}h(x)^\top S^{-1}h(x))\rho(x,t) \\ \rho(x,0) = \phi(x) \in C^\infty(\Omega). \end{cases} \tag{9}$$

1) PDF Approximation Based on Generalized Legendre Polynomials (GLP): We shall decompose the unnormalized density $\rho(x,t)$ through the GLP $\{\phi_k(x)\}$ constructed in [31]. Compared to other basis functions of $L^2(\mathbb{R}^n)$, the GLPs are simple, and more importantly, their values will vanish to zero when the variables approach the boundary of the interval $[-1,1]^n$.

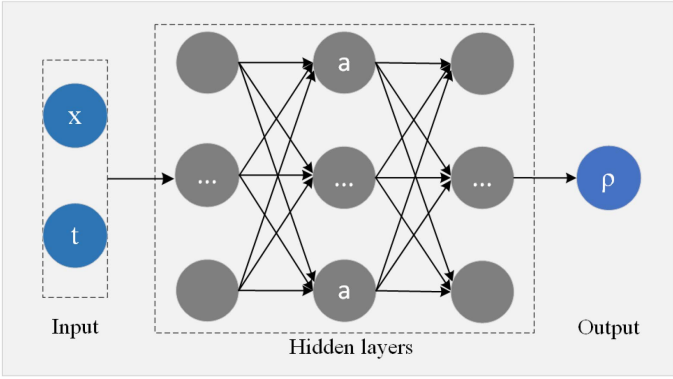


Fig. 1. Network architecture.

For one dimension case, the kth order GLP $\{\phi_k(x)\}$ is defined as

$$\phi_k(x) := c_k\left(\varphi_k(x) - \varphi_{k+2}(x)\right), c_k = \frac{1}{\sqrt{4k+6}}. \tag{10}$$

For high dimension case, i.e., $n \geq 2$, the $\boldsymbol{k}$th order GLP is defined as the tensor product of 1-D GLPs

$$\boldsymbol{\phi}_{\boldsymbol{k}}(x) := \phi_{k_1}(x_1) \cdot \phi_{k_2}(x_2) \cdots \phi_{k_n}(x_n), \mathbf{k} \in \mathbb{N}^n.$$

We define

$$V_M := \operatorname{span}\left\{\boldsymbol{\phi}_{\boldsymbol{k}}(x), \|\boldsymbol{k}\|_\infty \leq M\right\}.$$

Then, for all $\rho(x) \in W_0^{1,2}(\Omega) \subset L^2(\Omega)$, the projection is given by

$$\bar{\rho} := \sum_{\|\boldsymbol{k}\|_\infty \leq M} w_k \boldsymbol{\phi}_{\boldsymbol{k}}(x), M \in \mathbb{N} \tag{11}$$

where the coefficients w_k is determined by

$$\langle \rho - \bar{\rho}, \boldsymbol{\phi} \rangle = 0 \quad \forall \boldsymbol{\phi} \in V_M. \tag{12}$$

Remark 3.2: For our considered domain $\Omega = [a, b]$, by the following transformation:

$$\tilde{\phi}_k(x) = \phi_k\left(\frac{2x - (a+b)}{b-a}\right). \tag{13}$$

Then, $\tilde{\phi}_k(x), k = 0, \ldots, M$ form a group of basis functions on Ω. Hence, for notation simplicity, we only need to consider the case $\Omega = [-1, 1]$.

2) FKE Solver Based on Deep Neural Network: Now we consider the FKE equation (9) with the initial condition $\rho(x, 0)$ being GLPs $\phi_l(x), l = 0, \ldots M-1$ on the closed domain $D = B \times \Omega = [0, T_0] \times [-1, 1]$. Since we do not have any prior numerical solutions in advance, we will adopt the PINN method to solve FKE with $\phi_l(x)$.

The designed network architecture of the FKE solver is shown in Fig. 1. The network input is (x, t), i.e., points sampled from the domain D. The output $\hat{\rho}(x, t; \theta)$ represents the parametric solution of the FKE equation $\rho(x, t)$. We choose the $\tanh$ as the activation function $a(x)$ for all hidden layers. The weights and bias are initialized through Xavier uniform initialization. The Adam optimizer is employed with a dynamically adjusted learning rate $\eta_k = \eta_0 \times \gamma^k$, where η_0 is the initial rate and γ is a regulator factor. The early stop mechanism is employed during training whenever the loss is lower than threshold e_{stop}.

The whole training dataset consists of three datasets, i.e.,

$$X_{\text{train}} = X_f \cup X_b \cup X_{ic}. \tag{14}$$

1) We create the interior dataset X_f with N_f points sampled within domain D using Latin hypercube sampling (LHS). LHS provides better coverage and less redundancy compared to other random sampling methods.
2) We construct dataset X_b near the spatial boundary $\partial\Omega$, i.e., $B \times ([a, a + c_b] \cup [b - c_b, b])$, where c_b is a small positive number, by randomly sampling N_b points uniformly. Finer sampling is expected to improve model accuracy near the boundary.
3) On the initial boundary $\{t = 0\} \times \Omega$, we sample N_{ic} points using LHS to create dataset X_{ic}. We place denser sampling points near $\{t = 0\} \times \partial\Omega$ as these regions are harder to train.

Besides, we adopt an adaptive residual resampling strategy to accelerate model convergence. After every fixed epoch n_{rar}, we update the dataset X_f by adding resampled points with larger residual errors.

The loss function consists of three terms. For the kth training epoch, we denote the residual of the network as $\mathcal{R}(k)$. Specifically, for collocation points inside the computational region D, the network output should satisfy the equation constraint, i.e., for $(x^{(i)}, t^{(i)}) \in X_f$

$$\mathcal{R}_f^{(i)}(k) := |\mathcal{F}(\hat{\rho}(\theta^{(k)}; x^{(i)}, t^{(i)}))|^2.$$

For the near spatial boundary dataset X_b, they also satisfy the equation constraint as well, so we define the loss for them as follows, for $(x^{(i)}, t^{(i)}) \in X_b$

$$\mathcal{R}_b^{(i)}(k) := |\mathcal{F}(\hat{\rho}(\theta^{(k)}; x^{(i)}, t^{(i)}))|^2.$$

For the initial time boundary, they should be consistent with the initial condition $\rho(x, 0) = \phi_l(x)$, hence for $(x^{(i)}, t^{(i)}) \in X_{ic}$

$$\mathcal{R}_{ic}^{(i)}(k) := |\mathcal{F}(\hat{\rho}(\theta^{(k)}; x^{(i)}, t^{(i)})) - \phi_l(x^{(i)})|^2.$$

Furthermore, to make the network optimization procedure pay more attention to those points with larger residual errors as training goes by, we require the training points to be adaptively weighted. For the $(k+1)$th epoch, the weights for datasets X_f, X_b, X_{ic} are defined, respectively, as

$$\begin{aligned}
\omega_f^{(i)}(k+1) &= \mathcal{R}_f^{(i)}(k) / \sum_{i=1}^{N_f} \mathcal{R}_f^{(i)}(k) \\
\omega_b^{(i)}(k+1) &= \mathcal{R}_b^{(i)}(k) / \sum_{i=1}^{N_b} \mathcal{R}_b^{(i)}(k) \\
\omega_{ic}^{(i)}(k+1) &= \mathcal{R}_{ic}^{(i)}(k) / \sum_{i=1}^{N_{ic}} \mathcal{R}_{ic}^{(i)}(k).
\end{aligned} \tag{15}$$

The weights of different terms are initialized as

$$\omega_f^{(i)}(1) = \frac{1}{N_f}, \omega_b^{(i)}(1) = \frac{1}{N_b}, \omega_{ic}^{(i)}(1) = \frac{1}{N_{ic}}.$$

Finally, we have the following weighted loss for $(k+1)$th epoch:

$$\begin{aligned}
\mathcal{L}(\theta)|_{\theta=\theta^{(k+1)}} = \omega_f(k)\mathcal{R}_f(k+1) + \omega_b(k)\mathcal{R}_b(k+1) \\
+ \omega_{ic}(k)\mathcal{R}_{ic}(k+1).
\end{aligned} \tag{16}$$

3) Algorithms: The developed filtering method (named as DGLG) consists of two parts, the offline deep FKE solver is listed in Algorithm 1, and the online GLP estimator is listed in Algorithm 2.

C. Convergence Analysis

Before the convergence analysis of our method, let us recall that the Assumption (A.2) in our Assumption 1 essentially says that the growth of $|h|$ is greater than the growth of $|f|$. Under Assumption 1,

Algorithm 1: Offline Deep FKE solver.

1: **Initialization:** given the number of GLP basis function M, the off-line computation time T_0.
2: **generate** the training dataset X_{train}.
3: **for** $l = 0 : M-1$ **do**
4: **train** the FKE solver with $\rho(x,0) = \phi_l(x)$.
5: **predict** on the grid points of the considered domain D, and **store** the solution at T_0, i.e. $\Phi_l(x, T_0)$ up for the preparation of the on-line computation.
6: **end for**

Algorithm 2: Online GLP Estimator.

1: **Initialization:** Given the off-line data $\{\Phi_l(x,T_0)\}_{l=0}^{M-1}$.
2: **for** $i = 1, \ldots, N$ **do**
3: **project** $\hat{\rho}_i(x,\tau_{i-1})$ onto the GLP basis functions, $\hat{\rho}_i(x,\tau_{i-1}) \approx \sum_{l=0}^{M-1} c_{i,l}\phi_l(x)$.
4: **assemble** the solution $\hat{\rho}_i(x,\tau_i)$ of FKE by $\hat{\rho}_i(x,\tau_i) \approx \sum_{l=0}^{M-1} c_{i,l}\Phi_l(x,\Delta\tau)$.
5: **estimate** the current state by $\hat{x}(\tau_i) = \int_{\mathbb{R}^n} x \cdot \hat{u}_i(x,\tau_i)dx / \int_{\mathbb{R}^n} \hat{u}_i(x,\tau_i)dx$, here the solution $\hat{u}_i(x,\tau_i)$ of (5) is calculated by (7) and $\hat{\rho}_i(x,\tau_i)$.
6: **update** the initial pdf of the next time interval by

$$\hat{\rho}_{i+1}(x,\tau_i) = \exp(h(x)^\top S^{-1}(y_{\tau_i} - y_{\tau_{i-1}})) \cdot \hat{\rho}_i(x,\tau_i).$$

7: **end for**

the existence and uniqueness of a weak solution for the robust DMZ equation are guaranteed. Combined with Assumptions 3, the robust DMZ equation admits a smooth solution. Due to the page limit, we refer the readers to [36] and [38] for detailed discussions.

Now, let us define some notations. We define the the FKE equation associated operator

$$\mathcal{A} : L^2(\Omega) \to L^2(\Omega)$$

by $\varphi(x,T_0) = \mathcal{A}\varphi(x,0), \varphi(x,0) \in L^2(\Omega)$. The operator $\mathcal{A}$ depends on T_0 and $\mathcal{L}, h(x)$ and S.

Similarly, we define the deep FKE-solver associated operator

$$\mathcal{A}^{nn} : L^2(\Omega) \to L^2(\Omega)$$

by $\varphi(x,T_0) = \mathcal{A}^{nn}\varphi(x,0)$.

Note on the time interval $[\tau_{i-1},\tau_i)$, the approximation solution by our method is

$$\hat{u}_i(x,t) = \exp\left(-h(x)^\top S^{-1} y_{\tau_{i-1}}\right)\hat{\rho}_i(x,t). \tag{17}$$

Thus, our approximation solution is

$$\hat{u}(x,t) = \sum_{i=1}^{N} \hat{u}_i(x,t)\mathbb{I}_{[\tau_{i-1},\tau_i)}(t). \tag{18}$$

Theorem 3.1: Suppose Assumptions (1)–(3) hold, the approximation solution $\hat{u}(x,t)$ converges to the true solution $u(x,t)$ of (5) in L^1 sense with probability 1 over independently and identically samples (w. p. 1 i.i.d.), i.e.,

$$\hat{u}(x,t) \xrightarrow{\text{w. p. 1, i.i.d.}} u(x,t) \tag{19}$$

in $L^1(\Omega)$ as N, M, ξ(defined below) goes to $+\infty$.

Proof: For clarity, the proof is divided into three parts.

a) *Convergence of robust-DMZ equation solution:* First, note the pathwise approximate solution of (6) is

$$\tilde{u}(x,t) = \sum_{i=1}^{N} u_i(x,t)\mathbb{I}_{[\tau_{i-1},\tau_i)}(t). \tag{20}$$

Under the Assumption 1, it has been proven in [36] that in L^1 sense converges to $u(x,t)$, i.e.,

$$u(x,t) \overset{L^1}{=} \lim_{N\to\infty} \tilde{u}(x,t), \quad 0 \le t \le T. \tag{21}$$

Since $\tilde{u}(x,t)$ and $\hat{u}(x,t)$ are pathwise functions, and by Proposition 3.1, there is an one-to-one correspondence between $\rho_i(x,t)$ and $u_i(x,t)$, we only need to prove $\hat{\rho}_i(x,t)$ converges to $\rho_i(x,t)$.

b) *Convergence of GLP approximation:* Second, recall that on the time interval $[\tau_{i-1},\tau_i)$, the initial PDF of (8) is $\rho_i(x,\tau_{i-1})$, and its GLP projection $\bar{\rho}_i(x,\tau_{i-1})$. Therefore, we have

$$\begin{aligned}\rho_i(x,\tau_i) &= \mathcal{A}\rho_i(x,\tau_{i-1})\\ \bar{\rho}_i(x,\tau_i) &= \mathcal{A}\bar{\rho}_i(x,\tau_{i-1}).\end{aligned} \tag{22}$$

It has been proven in [6] that the GLP approximation error goes to zero as M goes to ∞, i.e.,

$$\lim_{M\to+\infty} \bar{\rho}_i(x,\tau_{i-1}) = \rho_i(x,\tau_{i-1}). \tag{23}$$

Then, under our assumptions, by the classical Galerkin approximation approach, we have

$$\lim_{M\to+\infty} \bar{\rho}_i(x,\tau_i) \overset{L^2}{=} \rho_i(x,\tau_i). \tag{24}$$

Since, the domain Ω is bounded, the convergence is also in L^1 sense naturally.

c) *Convergence of deep FKE-solver:* Third, the solutions of the deep FKE-solver with initial condition $\hat{\rho}_i(x,\tau_{i-1})$ is given by

$$\hat{\rho}_i(x,\tau_i) = \mathcal{A}^{nn}\hat{\rho}_i(x,\tau_{i-1}). \tag{25}$$

From Proposition 3.1, it is easy to derive that

$$\bar{\rho}_i(x,\tau_{i-1}) = \begin{cases}\sigma_0(x), & i = 1\\ Z_i(x)\bar{\rho}_{i-1}(x,\tau_{i-1}), & i \ge 2.\end{cases} \tag{26}$$

Here, $Z_i(x) := \exp\{h(x)^\top(y(\tau_{i-1}) - y(\tau_{i-2}))\}, i \le 2$. The same recursion holds for $\hat{\rho}_i(x,\tau_{i-1})$.

Next, we define the error between $\bar{\rho}_i(x,t)$ and $\hat{\rho}_i(x,t)$ by

$$e_i(x,t) := \bar{\rho}_i(x,t) - \hat{\rho}_i(x,t), \quad \tau_{i-1} \le t < \tau_i. \tag{27}$$

Then, by (25) we have

$$\begin{aligned}e_1(x,\tau_1) &= \mathcal{A}\sigma_0(x) - \mathcal{A}^{nn}\sigma_0(x)\\ e_i(x,\tau_i) &= \mathcal{A}\bar{\rho}_i(x,\tau_{i-1}) - \mathcal{A}^{nn}\hat{\rho}_i(x,\tau_{i-1})\\ &= B_{i,1} + B_{i,2}, \quad 2 \le i \le N\end{aligned} \tag{28}$$

where we denote

$$\begin{aligned}B_{i,1} &:= \mathcal{A}\bar{\rho}_i(x,\tau_{i-1}) - \mathcal{A}\hat{\rho}_i(x,\tau_{i-1})\\ B_{i,2} &:= \mathcal{A}\hat{\rho}_i(x,\tau_{i-1}) - \mathcal{A}^{nn}\hat{\rho}_i(x,\tau_{i-1}).\end{aligned}$$

Under the Assumption 2, by [33, Thm. 3.6] we have

$$\begin{aligned}e_1(x,\tau_1) &\xrightarrow{\text{w. p. 1 i.i.d.}} 0\\ B_{i,2} &\xrightarrow{\text{w. p. 1 i.i.d.}} 0, i \ge 2\end{aligned} \tag{29}$$

in $W_0^{1,2}(\Omega)$, thus $L^1(\Omega)$, as $\xi \to +\infty$, here in our case, $\xi := (N_f + N_b, N_{ic})$ denotes the numbers of training samples.

Now, for $B_{i,1}$ term, using the recursion (26), we have

$$B_{i,1} = \mathcal{A}(Z_i(x)e_{i-1}(x,\tau_{i-1})). \tag{30}$$

Under our assumption, we know that the operator $\mathcal{A}$ and the function $Z_i(x)$ are all bounded on Ω, there exist a constant $C_i > 0$, such that for $i \geq 2$

$$\|B_{i,1}\|_{L^2(\Omega)} \leq C_i \|e_{i-1}(x,\tau_{i-1})\|_{L^2(\Omega)}. \tag{31}$$

Since from (29), $e_1(x,\tau_1) \xrightarrow{\text{w. p. 1 i.i.d.}} 0$, we have

$$B_{2,1} \xrightarrow{\text{w. p. 1 i.i.d.}} 0$$

and then by (29,30), we know

$$e_2(x,\tau_2) \xrightarrow{\text{w. p. 1 i.i.d.}} 0.$$

By applying induction to the recursion (28), we have

$$e_i(x,\tau_i) \xrightarrow{\text{w. p. 1 i.i.d.}} 0, 1 \leq i \leq N. \tag{32}$$

In other words, we have

$$\hat{\rho}_i(x,\tau_i) \xrightarrow{\text{w. p. 1 i.i.d.}} \bar{\rho}_i(x,\tau_i), 1 \leq i \leq N \tag{33}$$

in $L^1(\Omega)$ as $\xi \to +\infty$.

Finally, the theorem's conclusion is followed by (21), (24), and (33). ■

IV. Numerical Results

In this section, three examples are tested to verify the availability and effectiveness of the newly proposed DGLG algorithm. In the first example, a 1-D highly nonlinear filter is provided, which contains oscillatory drift term, nontrivial diffusion coefficients, and cubic observation. The second example is a 2-D case with a highly oscillatory observation term. The third example exhibits a 2-D cubic sensor problem in which the drift term is an affine function. All three examples above are typical and challenging to be solved by traditional methods, such as EKF and PF.

The mean-square-error (MSE) metric is used to measure the accuracy of filtering algorithms at each instant and the mean of MSE (MMSE) for the whole time T

$$\begin{aligned} \text{MSE}(t_k) &:= \tfrac{1}{N_{tr}} \textstyle\sum_{i=1}^{N_{tr}} (\hat{X}^i_{t_k} - X^i_{t_k})^2 \\ \text{MMSE} &:= \tfrac{1}{N} \textstyle\sum_{k=1}^{N} \text{MSE}(t_k) \end{aligned} \tag{34}$$

where $\hat{X}^i_t$ denotes state estimate in ith trial at instant t. X^i_t represents the real state trajectory. N_{tr} denotes number of simulation trials. Mean time (MT) is defined as

$$\text{MT} = \frac{1}{N_{tr}} \sum_{i=1}^{N_{tr}} T_i$$

where T_i is the computational time for ith trial. In all three examples, the number of independent trials N_{tr} is set to 20.

Example 4.1 (1-D highly nonlinear system):

$$\begin{cases} dX_t = a\sin(X_t)dt + \sigma_B dW_t, \quad \mathbb{E}[(dW_t)^2] = dt \\ dZ_t = 0.5X_t^3 dt + dV_t, \quad \mathbb{E}[(dV_t)^2] = Sdt \\ X_0 \sim \sigma_0 := \mathcal{N}(\mu_0 = 0.1, \sigma_0 = 0.05) \end{cases} \tag{35}$$

where $a = 0.2$, diffusion coefficient $\sigma_B = 1.2$, and observation variance $S = 0.03$. Let the Total simulation time be $T = 4$ s. The time increment of the evolution of the filtering system is $dt = 0.001$. The number of GLP basis functions M is chosen to be 7. Observation time increment $\Delta t = 0.01$. For the solution of the SM, the time increment is set to 0.001. For the PF algorithm, we choose 20 particles to evolve. For DGLG, during the training stage, a fully connected network is chosen with layers $[2, 80, 80, 80, 1]$ in the feasible region $(t, x) \in [0, 0.3] \times [-1, 1]$. The maximal epoch is set to 10 000 with the Adam optimizer. For each of the three parts in the loss function, i.e., interior, initial, and boundary parts, we shall sample 1000 collocation points. The results of state tracking are shown in Table I and Fig. 2. In terms of MSE, it can be found that DGLG and SM algorithm both attain the most accurate result, which is lower than EKF by 42% and than PF by 34%. According to the MT result, we shall find that DGLG has the fastest simulation speed in which computational time is lower than SM by 66% than PF by 51%. The conditional density evolution is shown in Fig. 3(left). Due to the nonlinear structure contained in drift and observation, it is a moderately challenging problem to recover the real state from the corresponding observation data. The DGLG method performs best both in computational time and MMSE for such a problem.

TABLE I
Performance of All Simulated Algorithms in Example 4.1.

Algorithms	DGLG	SM	EKF	PF
MT	0.1407	0.428	0.004	0.2875
MMSE	0.1823	0.1798	0.3151	0.2762

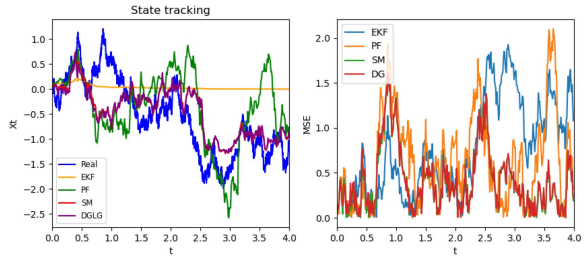


Fig. 2. State tracking (left) and MSE (right) in Example 4.1.

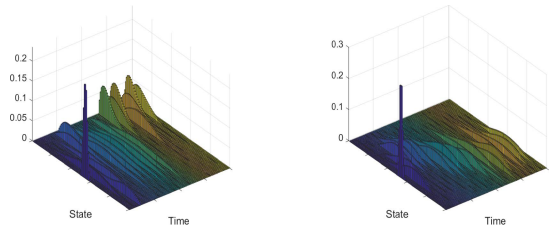


Fig. 3. Conditional density in Example 4.1(left) and Example 4.2(right).

Example 4.2 (2-D nonlinear observation system):

$$\begin{cases} dX_t = (a_1 + a_2 X_t)dt + dW_t, \quad \mathbb{E}[(dW_t)^2] = dt \\ dZ_t = \begin{bmatrix} X_t \sin(X_t) \\ X_t \cos(X_t) \end{bmatrix} dt + dV_t, \quad \mathbb{E}[dV_t dV_t^\top] = Sdt \\ X_0 \sim \mathcal{N}(\mu_0 = 0.1, \sigma_0 = 0.05) \end{cases} \tag{36}$$

where covariance matrix is set $S = sI$ for simplicity.

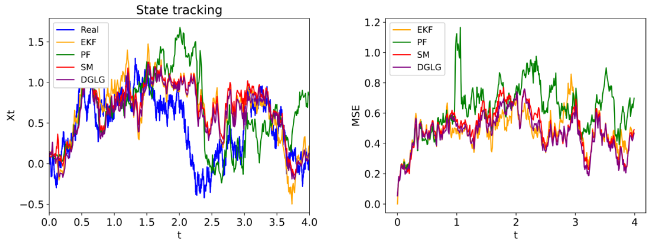
The corresponding FKE equation is

$$\begin{aligned} \frac{\partial \rho(t,x)}{\partial t} = \frac{1}{2}\frac{\partial^2 \rho(t,x)}{\partial x^2} - (a_1 + a_2 x)\frac{\partial \rho(t,x)}{\partial x} \\ - (a_2 + \frac{1}{2}x^2 s^{-1})\rho(t,x) \end{aligned} \tag{37}$$

TABLE II
PERFORMANCE OF ALL SIMULATED ALGORITHMS IN EXAMPLE 4.2

Algorithms	DGLG	SM	EKF	PF
MT	0.283	0.492	0.006	1.623
MMSE	**0.499**	0.506	0.648	0.577

The bold value indicate the smallest MMSE value.



(a) (b)

Fig. 4. State tracking (a) and MSE (b) in Example 4.1.

where drift coefficients $a_1 = 0.3, a_2 = -0.1$. Total simulation time $T = 4$. Number of GLPs $M = 8$. The number of particles for the PF algorithm is $N_{\text{pf}} = 50$. For DGLG, the setting of hyperparameters is the same as the Example 4.1.

Accordingly, results of state tracking and MSE have been shown in Table II and Fig. 4. The conditional density evolution is shown in Fig. 3(right). Nonlinear property in this example appears in the observation function, which exhibits strong oscillation behavior itself. In terms of MSE, DGLG has the best performance, which is lower than EKF by 23% and PF 13.5%. According to running time, it can be noticed that DGLG significantly speeds up the SM by 42.4% and PF by 82.5%.

Example 4.3 (2-D cubic system):

$$\begin{cases} dX_t = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} X_1 \\ X_2 \end{bmatrix} dt + dW_t, \quad \mathbb{E}[dW_t dW_t{}^\top] = Qdt \\ dZ_t = \begin{bmatrix} X_1^3 \\ X_2^3 \end{bmatrix} dt + dV_t, \quad \mathbb{E}[dV_t dV_t{}^\top] = Sdt \\ X_0 \sim \mathcal{N}([0.1, 0.1]^\top, 0.05 I_2) \end{cases} \tag{38}$$

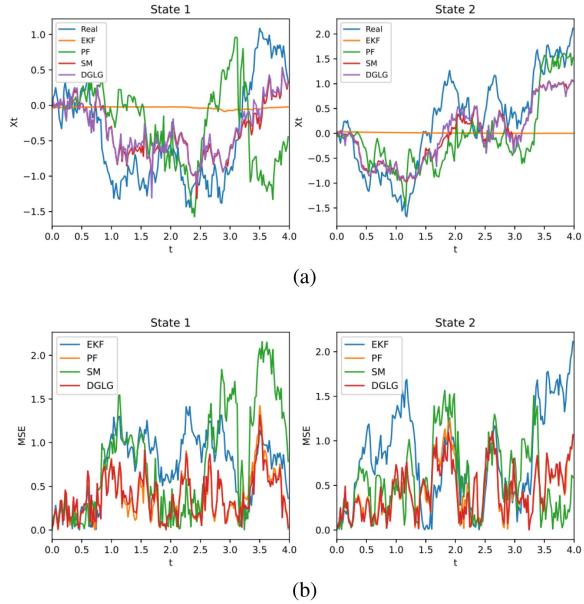
where covariance matrix is set $S = sI$ for simplicity.

The corresponding FKE equation in this case is

$$\begin{aligned} \frac{\partial v}{\partial t}(t, x_1, x_2) = \frac{1}{2}(v_{x_1x_1} + v_{x_2x_2}) - (a_{11}x_1 + a_{12}x_2)v_{x_1} \\ - (a_{21}x_1 + a_{21}x_2)v_{x_2} \\ - (a_{11} + a_{22})v - \frac{1}{2}(x_1^6 + x_2^6)s^{-1}v \end{aligned} \tag{39}$$

where drift coefficients $a_{11} = -0.4, a_{12} = 0.1, a_{21} = 0, and a_{22} = -0.6$. Spatial scaling factor $C = 1.2$. Covariance coefficient $s = 0.1$. The number of GLPs $M = 15$. Particle number is $N_{\text{pf}} = 50$. For DGLG, the architecture of the neural network is $[3, 100, 100, 100, 1]$ and we shall uniformly select collocation points for independent variables (t, x_1, x_2) in terms of initial, boundary, and residual regions. Training time interval is chosen as $t \in [0, 0.4]$.

As Fig. 5 shows, for this cubic sensor system, traditional EKF exhibits an invalid estimation for both states. In terms of state tracking, PF can only give the rough trend of state evolution, especially for state x_1. In terms of MSE, DGLG has the best and same performance, which is lower than EKF by 45% and PF 43%. According to running



(a)

(b)

Fig. 5. State tracking (a) and MSE (b) in Example 4.3.

TABLE III
PERFORMANCE OF ALL SIMULATED ALGORITHMS IN EXAMPLE 4.3.

Algorithms	DGLG	SM	EKF	PF
MT	1.234	2.192	0.019	2.967
MMSE	**0.549**	0.562	0.993	0.958

The bold value indicate the smallest MMSE value.

time, as shown in Table III, DGLG significantly speeds up the SM by 44% and PF by 58%. DGLG exhibits the best performance for both state estimations while largely enhancing the computational efficiency compared with SM.

V. CONCLUDING REMARKS

We propose an efficient filtering algorithm using deep neural networks and the classical Galerkin approach. The observation-independent FKE is solved with a deep neural network under the PINN paradigm, approximating the unnormalized density function with GLP basis functions to handle time-varying initial conditions. This method requires only mild assumptions, supports arbitrary initial distributions, and converges theoretically to the true DMZ equation solution. It is implemented in a real-time, memoryless manner, demonstrating superior efficiency and stability in three numerical experiments compared to EKF, SM, and PF.

The method is limited by the "curse of dimensionality," making it suitable for moderate-high dimension systems. Future work will focus on designing an efficient FKE operator network to reduce offline training costs, inspired by recent advancements in deep neural network-based operator learning.

REFERENCES

[1] D. O. B. Anderson and J. B. Moore. *Optimal Filtering*. North Chelmsford, MA, USA: Courier Corporation, 2012.
[2] I. Arasaratnam and S. Haykin, "Cubature Kalman filters," *IEEE Trans. Autom. Control*, vol. 54, no. 6, pp. 1254–1269, Jun. 2009.
[3] VE Beneš, "Exact finite-dimensional filters for certain diffusions with nonlinear drift," *Stochastics: An Int. J. Probability Stochastic Processes*, vol. 5, no. 1-2, pp. 65–92, 1981.

[4] S. Bonnable, P. Martin, and E. Salaü"n, "Invariant extended Kalman filter: Theory and application to a velocity-aided attitude estimation problem," in *Proc. 48th IEEE Conf. Decis. Control (CDC) held jointly with 2009 28th Chin. Control Conf.*, 2009, pp. 1297–1304.

[5] G. Calafiore, "Reliable localization using set-valued nonlinear filters," *IEEE Trans. Syst., Man, Cybern.-Part A: Syst. Humans*, vol. 35, no. 2, pp. 189–197, Mar. 2005.

[6] C. Canuto, M. Y. Hussaini, A. Quarteroni, and A. Z. Thomas, *Spectral Methods in Fluid Dynamics*. Berlin, Germany: Springer Science & Business Media, 2012.

[7] J. Chen, "On ubiquity of Yau filters," in *Proc. 1994 Amer. Control Conf.*, 1994, pp. 252–254.

[8] N. Cui, L. Hong, and Jeffery R. Layne, "A comparison of nonlinear filtering approaches with an application to ground target tracking," *Signal Process.*, vol. 85, no. 8, pp. 1469–1492, 2005.

[9] Q. Cui, R. Ding, B. Zhou, and X. Wu, "Path-tracking of an autonomous vehicle via model predictive control and nonlinear filtering," *Proc. Inst. Mech. Engineers, Part D: J. Automobile Eng.*, vol. 232, no. 9, pp. 1237–1252, 2018.

[10] W. Dong, X. Luo, and S. S.-T. Yau, "Solving nonlinear filtering problems in real time by Legendre Galerkin spectral method," *IEEE Trans. Autom. Control*, vol. 66, no. 4, pp. 1559–1572, Apr. 2021.

[11] E. Tyrone Duncan, "Probability densities for diffusion processes with applications to nonlinear filtering theory and detection theory", Ph.D. dissertation, Syst. Theory Lab., Standford Univ., Stanford, CA, USA, 1967.

[12] A. L. Escoriza, G. Revach, N. Shlezinger, and R. J. G. van Sloun, "Data-driven Kalman-based velocity estimation for autonomous racing," in *Proc. 2021 IEEE Int. Conf. Auton. Syst.*, 2021, pp. 1–5.

[13] G. Evensen, "Sequential data assimilation with a nonlinear quasi-geostrophic model using Monte Carlo methods to forecast error statistics," *J. Geophysical Res.: Oceans*, vol. 99, no. C5, pp. 10143–10162, 1994.

[14] Neil J. Gordon, David J. Salmond, and Adrian FM Smith, "Novel approach to nonlinear/non-Gaussian Bayesian state estimation," *IEE Proc. F. (Radar Signal Process.)*, vol. 140, no. 2, pp. 107–113, 1993.

[15] L. Hostetler and R. Andreas, "Nonlinear Kalman filtering techniques for terrain-aided navigation," *IEEE Trans. Autom. Control*, vol. 28, no. 3, pp. 315–323, Mar. 1983.

[16] K. Ito and K. Xiong, "Gaussian filters for nonlinear filtering problems," *IEEE Trans. Autom. control*, vol. 45, no. 5, pp. 910–927, May 2000.

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

[18] R. E. Kalman, "A new approach to linear filtering and prediction problems," *J. Basic Eng.*, vol. 82, no. 1, pp. 35–45, 1960.

[19] R. E. Kalman and R. S. Bucy, "New results in linear filtering and prediction theory," *J. Basic Eng.*, vol. 83, no. 1, pp. 95–108, 1961.

[20] G. Em Karniadakis et al., "Physics-informed machine learning," *Nature Rev. Phys.*, vol. 3, no. 6, pp. 422–440, 2021.

[21] R. G. Krishnan, U. Shalit, and D. Sontag, "Deep Kalman filters," 2015, *arXiv:1511.05121v2*.

[22] H. J. Kushner, "On the differential equations satisfied by conditional probablitity densities of Markov processes, with applications," *J. Soc. Ind. Appl. Math., Ser. A: Control*, vol. 2, no. 1, pp. 106–119, 1964.

[23] D.-J. Lee, "Nonlinear Bayesian filtering with applications to estimation and navigation," Ph.D. dissertation, Texas A&M Univ., 2005. [Online]. Available: https://hdl.handle.net/1969.1/2269

[24] X.-R. Li and V. P. Jilkov, "A survey of maneuvering target tracking: Approximation techniques for nonlinear filtering," in *Signal Data Process. Small Targets 2004*, 2004, vol. 5428, pp. 537–550.

[25] X. Luo and S.S.-T. Yau, "Hermite spectral method to 1-D forward Kolmogorov equation and its application to nonlinear filtering problems," *IEEE Trans. Autom. Control*, vol. 58, no. 10, pp. 2495–2507, Oct. 2013.

[26] R. E. Mortensen, "Optimal control of continuous time stochastic systems," Ph.D. thesis, University of California, Berkley, CA, USA, Aug. 1966.

[27] M. Raissi, P. Perdikaris, and G. E. Karniadakis, "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations," *J. . Comput. Phys.*, vol. 378, pp. 686–707, 2019.

[28] G. Revach et al., "Kalmannet: Neural network aided Kalman filtering for partially known dynamics," *IEEE Trans. Signal Process.*, vol. 70, pp. 1532–1547, 2022.

[29] G. G. Rigatos, "Particle filtering for state estimation in nonlinear industrial systems," *IEEE Trans. Instrum. Meas.*, vol. 58, no. 11, pp. 3885–3900, Nov. 2009.

[30] G. G. Rigatos, *Nonlinear Control and Filtering Using Differential Flatness Approaches: Applications to Electromechanical Systems*. Berlin, Geramny: Springer, 2015.

[31] J. Shen, "Efficient spectral-Galerkin method I: Direct solvers of second- and fourth-order equations using Legendre polynomials," *SIAM J. Sci. Comput.*, vol. 15, no. 6, pp. 1489–1505, 1994.

[32] J. Shi, X. Chen, and S. S. -T. Yau, "A novel real-time filtering method to general nonlinear filtering problem without memory," *IEEE Access*, vol. 9, pp. 119343–119352, 2021.

[33] Y. Shin, J. Darbon, and G. Em Karniadakis, "On the convergence of physics informed neural networks for linear second-order elliptic and parabolic type pdes," *Commun. Comput. Phys.*, vol. 28, no. 5, pp. 2042–2074, 2020.

[34] R. L. Stratonovich, "On the theory of optimal non-linear filtering of random functions," *Theory Probability Appl.*, vol. 4, pp. 223–225, 1959.

[35] T. Yang, P. G. Mehta, and S. P. Meyn, "Feedback particle filter," *IEEE Trans. Autom. Control*, vol. 58, no. 10, pp. 2465–2480, Oct. 2013.

[36] S.-T. Yau and S.S.-T. Yau, "Real time solution of the nonlinear filtering problem without memory II," *SIAM J. Control Optim.*, vol. 47, no. 1, pp. 163–195, 2008.

[37] S.-T. Yau and S. S.-T. Yau, "Real time solution of nonlinear filtering problem without memory II," *Math. Res. Lett.*, vol. 7, no. 6, pp. 671–693, 2000.

[38] T. S. Yau and S. S. T. Yau, "Existence and uniqueness of solutions for Duncan-Mortensen-Zakai equations," in *Proc. 44th IEEE Conf. Decis. Control*, 2006, pp. 536–541.

[39] M. Zakai, "On the optimal filtering of diffusion process," *Z. Wahrsch. Verw. Gebiete*, vol. 11, no. 3, pp. 230–243, 1969.

[40] Z. Zhao, T. X. R. Li, and V. P. Jilkov, "Best linear unbiased filtering with nonlinear measurements for target tracking," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 40, no. 4, pp. 1324–1336, Oct. 2004.