# A Novel Logarithmic Transformed Deep Galerkin Approach To Optimal Filtering Problem

Ji Shi†, *member, IEEE*, Xiaopei Jiao†, and Stephen S.-T. Yau*, *Fellow, IEEE*

***Abstract*— The optimal filtering problem for general nonlinear state-observation systems has garnered significant attention in control theory. At its core, optimal filtering involves determining the probability density function of the system state conditioned on historical observations. The Yau-Yau method [25], a pioneering framework, offers a viable approach with comprehensive theoretical guarantees and practical numerical implementation. Specifically, the Yau-Yau framework comprises two key components: offline solution of the forward Kolmogorov equation (FKE) and online data assimilation updates. The primary challenge lies in efficiently and accurately solving the FKE, as it directly impacts the real-time filtering process. To address this fundamental obstacle, we propose a highly efficient filtering algorithm that combines a FKE solver based on deep neural networks and a PDF approximator using generalized Legendre polynomials. By integrating advanced deep learning techniques with Galerkin approximation, we introduce the logarithmic transformed deep Galerkin approach (LTDG). The numerical simulations showcase the effectiveness and accuracy of our newly proposed algorithm. LTDG demonstrates superior performance compared to other methods, such as the extended Kalman filter and particle filter, and it successfully maintains the high accuracy of the Galerkin spectral method while having fewer online computational burdens.**

## I. Introduction

Nonlinear filtering is a kind of real-time signal denoising method in numerous practical application fields including target tracking [17], navigation systems [16], robotics [6], and advanced control systems [21]. In such scenarios, the well-established Duncan-Mortensen-Zakai (DMZ) equation [9], [19], [27] governs the update equation for the unnormalized conditional density of the system state. Generally, the conditional density cannot be described by a finite set of sufficient statistics, rendering it an element of an infinite-dimensional space. While explicit solutions to the DMZ equation exist in special cases such as Kalman filter [14] and Beneš filter [4], solving it straightforwardly proves exceedingly challenging for most general systems.

Given the challenge of solving the DMZ equation directly, researchers seek approximate solutions using various methodologies. Many filtering algorithms are based on the Kalman filter and its variants. Notably, the Extended Kalman Filter (EKF) [1] linearizes nonlinear terms with respect to the current mean before applying the Kalman Filter. Conversely, the Unscented Kalman filter (UKF) [13] utilizes unscented transformation to propagate the mean and covariance of the state, approximating the conditional density up to the second order if Gaussian. In geoscience, weather forecasting poses a significant challenge, addressed by the Ensemble Kalman filter (EnKF) [10], which represents the system state distribution using ensembles and replaces the covariance matrix with sample covariance. Other Kalman filter-based filters are also based on the Gaussian approximation, such as the Cubature Kalman filter (CKF) [2], Invariant Extended Kalman filter (IEKF) [5], and Central Difference Kalman filter (CDKF) [12]. The Particle filter (PF) [11] is employed to handle nonlinear and non-Gaussian systems by utilizing numerous independent random samples to approximate the posterior distribution of the state process. However, PF is hindered by its substantial computational burden and often faces the issue of particle degeneracy. To address this, the resampling techniques have been proposed. In [24], the Feedback Particle filter (FPF) was introduced from the perspective of mean-field theory.

At the beginning of the 21st century, Yau et al. proposed the Yau-Yau filtering algorithm [25], [26], which is an optimal filter for general nonlinear system estimation and arbitrary initial distribution. As a real-time, memoryless algorithm with theoretical guarantee, there have been large interests and developments in the Yau-Yau algorithm. It consists of two steps which are offline prediction and online update. Kolmogorov equation appearing in the Yau-Yau method plays an essential role in the fast state inference. The key aspect of the Yau-Yau filtering framework lies in designing a filter capable of adapting to changing initial conditions for a forward Kolmogorov equation (FKE). Based on the fast solution of a Kolmogorov equation, an amount of PDE-based approaches have been designed [7], [23]. On the one hand, Gaussian decomposition technique was proposed in "Direct method" to deal with finite-dimensional filter in which state estimation can be described by a finite set of statistics. On the other hand, the Galerkin-spectral method is introduced to solve a parabolic Kolmogorov equation which yields the novel Hermite spectral method (HSM) and Legendre spectral method (LSM) [8], [18]. Galerkin-spectral filtering algorithms mainly admit an advantage of possessing theoretical convergence rate analysis and outperforming EKF, PF, etc in low dimensional systems. However, since the requirement of fast inference in engineering, it is still a challenging issue how to design a low-compute, rapid

† Equal contribution; * Corresponding author.

This work is supported by National Natural Science Foundation of China (NSFC) grant (12101426,11961141005), Tsinghua University start-up fund, and Tsinghua University Education Foundation fund (042202008).

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 Department of Applied Mathematics, University of Twente, Netherlands (e-mail: jack.jiao@utwente.nl).

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

inference algorithm based on the Yau-Yau framework.

In recent years, with the development of deep learning, a new research area named scientific machine learning has been created which combines deep learning and traditional numerical partial differential equations (PDEs), which sheds light on the optimal filtering problem. However, training deep neural networks often requires large datasets, which may not always be available for scientific computing. In response to this challenge, Karniadakis et al. introduced the physics-informed neural network (PINN) framework in [20] for solving various PDE problems. This framework enforces physics laws as optimization constraints, enabling an unsupervised training strategy. PINN combines mesh-free, data-driven, and physics-based constraints, making it a potentially powerful tool for optimal filtering.

Motivated by the above observations, in this paper, we propose the logarithmic transformed deep Galerkin approach (LTDG) under Yau-Yau framework, leveraging a combination of PINN and Galerkin-spectral approximation based on generalized Legendre polynomials (GLPs). The proposed algorithm is straightforward to numerically implement and efficiently. Additionally, it offers real-time processing and memoryless operation. Numerical experiments validate the effectiveness of our method, demonstrating its ability to successfully track challenging problems such as the notorious cubic sensor problem, where traditional methods such as the EKF and PF often fail most time. This implies LTDG outperforms traditional EKF, PF and LSM in both accuracy and CPU computational time.

The rest of the paper is structured as follows: Section II provides a quick overview of the nonlinear optimal filtering theory and algorithm. Section III provides the derivation of the deep neural network-based FKE solver and associated novel filtering algorithm LTDG. Section IV illustrates details about the numerical implementation of several examples. Section V will give a conclusion of the evaluation of our algorithm.

## II. Basic concepts and Preliminaries

In this section, we shall introduce the nonlinear filtering problem and the optimal filtering briefly, and the goal of this paper. At the beginning, some notations used in the paper will be presented for the convenience of the readers.

**Notations:** The set of real numbers is denoted by $\mathbb{R}$. $\mathbb{R}^n$ refers to $n$ dimensional Euclidean space. Let $C^\infty(\Omega)$ be the set of smooth functions defined on a domain $\Omega$. $L^2(\Omega)$ represents collection of all square integrable functions defined on $\Omega$. $\Delta(*)$ is the Laplacian operator, $\nabla\cdot(*)$ is the divergence operator. In scalar case, $\mathcal{N}(a,b)$ denotes a Gaussian distribution with mean $a$ and standard deviation $b$. In vector case, $\mathcal{N}(\mu,\Sigma)$ denotes a Gaussian distribution with mean vector $\mu$ and covariance matrix $\Sigma$. $U(\Omega)$ denotes the uniform distribution at the region $\Omega$.

In this paper, we consider the type of continuous filtering problem described by the following signal-observation stochastic differential equations (SDE):

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

where $x_t := x(t) \in \mathbb{R}^n$ is the state of the system at time $t$, $y_t := y(t) \in \mathbb{R}^m$ is the observation with $y_0 = 0$, $f(x_t) \in \mathbb{R}^n$, $h(x_t) \in \mathbb{R}^m$ are vector value drift term and observation term, respectively. $w_t, v_t$ are independent standard Brownian motion processes. Both $\{w_t\}_{t\geq 0}$, $\{v_t\}_{t\geq 0}$ and initial state $x_0$ are mutually independent.

The minimum variance estimate of $x_t$ is given by $E[x_t|\mathcal{Y}_t]$, where $\mathcal{Y}_t := \{y_s : 0 \leq s \leq t\}$ is the observation history up to time $t$. Since the conditional probability density function (PDF) $p(x,t)$ embodies all the statistical inforamtion about $x_t$, it is sufficient to obtain the PDF $p(x,t)$. It is well known that $p(x,t)$ satisfies the Kushner-Stratonovich(K-S) equation which is a nonlinear stochastic PDE about $p(x,t)$ and is very difficult to solve. Later on, Duncan, Mortensen, and Zakai proposed independently an unnormalized version of the K-S equation, i.e., the DMZ equation

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

here,

$$\mathcal{L}(*) := \frac{1}{2}\Delta(*) - \nabla\cdot(f*), \tag{3}$$

$\sigma_0(x)$ is an unnormalized version density function of the initial state $x_0$.

Note in the DMZ equation, there appears the term $dy_t$. In real applications, the usually discontinuous trajectories of discretized observations $y_t$ render underlying filtering algorithms lack robustness. Therefore, people are interested in considering robust state estimators from observed sample paths with some properties of robustness. By making the following gauge transformation:

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

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

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

The equation (5) is called the "pathwise-robust" DMZ equation. Generally speaking, the equation (5) does not have a closed-form solution. To solve this equation, Yau et al. developed a theoretical filtering framework and proved the convergence of the approximate solution in [25],

Let us assume the total filtering time is $T$. The observations occur at time sequence $P_N = \{0 = \tau_0 < \tau_1 < \cdots < \tau_N = T\}$. Without loss of generality, we assume that the observations arrive at an equal time span, i.e., $\forall 1 \leq i \leq N, \tau_i - \tau_{i-1} = \Delta\tau > 0$.

Let $u_i$ be the solution of the robust DMZ equation (5) with observation process fixed on the interval $[\tau_{i-1}, \tau_i)$ by $y_t = y_{\tau_{i-1}}$, i.e.

$$\begin{cases} \frac{\partial u_i}{\partial t}(x,t) = \exp(-h^\top y_{\tau_{i-1}}) \left(\mathcal{L} - \frac{1}{2}h^\top h\right) \\ \qquad\qquad \cdot \left[\exp(h^\top y_{\tau_{i-1}}) u_i(x,t)\right], \\ u_1(x,0) = \sigma_0(x), \\ u_i(x,\tau_{i-1}) = u_{i-1}(x,\tau_{i-1}), \text{for } i = 2,3,\cdots,N. \end{cases} \tag{6}$$

Note the observations $y_{\tau_i}$ are contained in the coefficients of (6), by the following proposition [25], the equation (6) can be transformed into a linear, observation-independent equation.

*Proposition 2.1:* For each $\tau_{i-1} \le t < \tau_i, i = 1,2,\cdots,N$, $u_i(x,t)$ satisfies (6) if and only if

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

satisfies the following forward Kolmogorov equation(FKE)

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

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

This property of observation-independent enables us to solve the FKE (8) in advance. By solving the equation (8) with a group of basis functions of $L^2(\mathbb{R}^n)$ as initial conditions offline, the online state estimation can be done via PDF projection in real-time.

## III. Deep neural network based FKE solver and filtering algorithm

The key part of Yau-Yau filtering framework is how to solve the FKE efficiently. In this section, we shall develop a new efficient algorithm to compute the FKE equation (8) with specific initial conditions by a deep neural network. For notation convenience, we omit the subscript in (8) as follows:

$$\begin{cases} \frac{\partial \rho}{\partial t}(x,t) = \left(\mathcal{L} - \frac{1}{2}h^\top h\right)\rho(x,t), \\ \rho(x,0) = \phi(x), \end{cases} \tag{9}$$

where the initial condition $\phi(x) \in C^\infty(\mathbb{R}^n)$.

In the following, we shall solve the above FKE with GLPs as initial conditions under the PINN framework, since the PINN can solve various PDE problems in an unsupervised learning way. However, it is quite difficult to train the PINN network, especially when the initial condition becomes complex. Therefore, we first transform the original FKE with GLP initial conditions into a new parabolic PDE with simple initial conditions. Then we solve the log-transformed PDE via a specially designed PINN method.

### A. Log Transformed Galerkin Approach To FKE

We shall numerically solve the above FKE by Galerkin method, i.e., the solution $\rho(x,t)$ will be approximated through a group of selected basis functions. In this paper, we shall use the GLPs $\{\phi_k(x)\}$ as constructed in [22] since the GLP vanishes to zero at the boundary of $[-1,1]$. In practice, we always do filtering estimation in a finite time $T$, hence the states are in a bounded domain $\Omega = [-a,a]^n$ over the entire filtering time interval. Therefore, it is reasonable to assume the state density $\rho(x,t)$ is supported on the considered bounded domain $\Omega$.

The one dimensional GLP $\{\phi_k(x)\}$ is defined as

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

where $\varphi_k(x)$ is Legendre polynomial of degree $k$. For high dimensional case, let us define the tensor product of the one dimension GLPs:

$$\phi_k(x) := \phi_{k_1}(x_1)\cdot\phi_{k_2}(x_2)\cdots\phi_{k_n}(x_n),$$

$k = (k_1,\cdots,k_n) \in \mathbb{N}^n$. By scaling and translating transformations, $\phi_k(x)$ can be easily generalized to interval $[-a,a]$.

Now, the FKE (9) becomes

$$\begin{cases} \frac{\partial \rho}{\partial t}(x,t) = \frac{1}{2}\Delta\rho - f\cdot\nabla\rho - (\nabla\cdot f + \frac{1}{2}h^\top h)\rho(x,t) \\ \rho(x,0) = \phi_k(x) \end{cases} \tag{11}$$

Suppose the solution of equation (11) with initial condition $\phi_k(x)$ is $\rho^{(k)}(x,t)$.

Let $z = \frac{x}{a} + c, c > 1$, then $z \in D := [c-1, c+1]^n$. We define $\tilde{\rho}(z,t) := \rho(x,t)$, then we have

$$\begin{cases} \frac{\partial \tilde{\rho}}{\partial t}(z,t) = \frac{1}{2}a^2\Delta\tilde{\rho} - af\cdot\nabla\tilde{\rho} - (a\nabla\cdot f + \frac{1}{2}h^\top h)\tilde{\rho}(z,t), \\ \tilde{\rho}(z,0) = \phi_k(a(z-c)), \end{cases} \tag{12}$$

Note the initial condition $\phi_k(z)$ is a multivariate polynomial. Therefore it is a linear combination of

$$z^\alpha := z_1^{\alpha_1} z_2^{\alpha_2}\cdots z_n^{\alpha_n}, \alpha = (\alpha_1,\cdots,\alpha_n).$$

Suppose the 1-d polynomial $\phi_{k_i}(z_i)$ has the form of

$$\phi_{k_i}(z_i) = \sum_{j=0}^{k_i+2} c_j^{(i)} z_i^j, \tag{13}$$

then the coefficient of the term $z^\alpha$ is $\prod_{i=1}^n c_{\alpha_i}^{(i)}$. Without loss of generality, we only need to consider one dimension case with $\phi_k(z) = z^k, k \in \mathbb{N}$. Suppose the solution of (12) with initial condition $\tilde{\rho}(z,0) = z^k$ is $\tilde{\rho}^{(k)}(z,t)$.

Next, we define $v(z,t) := \ln\tilde{\rho}(z,t)$, then the equation (12) becomes following log-FKE,

$$\begin{cases} \frac{\partial v}{\partial t}(z,t) = \mathcal{L}_{log}[v](z,t), \quad (z,t) \in D\times[0,T], \\ v(z,0) = \varphi(z), \end{cases} \tag{14}$$

where $\mathcal{L}_{log}[v] := \frac{1}{2}a^2(\Delta v + (\nabla\cdot v)^2) - af\cdot\nabla v - (a\nabla\cdot f + \frac{1}{2}h^\top h)$ and $\varphi(z) := k\ln z$. Suppose the solution of (14) with initial condition $k\ln z$ is $v^{(k)}(z,t)$. This equation is a nonlinear PDE with an almost linear initial condition. Hence, it is much easier for the network to learn the solution.

Finally, the solution to the robust DMZ equation (5) can be obtained by the following computation flow:

$$v^{(k)} \xrightarrow[\text{transform}]{\exp} \tilde{\rho}^{(k)} \xrightarrow[\text{combination}]{\text{linear}} \rho^{(k)} \xrightarrow[\text{transform (7)}]{\text{gauge}} u(x,t) \tag{15}$$

### B. Deep log-FKE Solver

To begin with the work of Raissi et. al. [20], the Physics-Informed neural network will be utilized to approximate the solution of log-FKE (14). The architecture of the neural network consists of a multilayer perceptron with one head layer, four hidden layers, and an output layer. The input of the network is $(x,t)$, i.e. points sampled from the domain $D$. The output $v(x,t;\theta)$ represents the parametric solution of the log FKE equation (14). For simplicity, the number of neurons of each hidden layer is set to be the same. We choose the *tanh* function as the activation function for all hidden layers. Besides, we initialized the weights of each layer through Xavier uniform initialization, and all biases are initialized as 0.

A high-quality training dataset is crucial for the successful training of a deep learning model. We construct the interior dataset $X_r$ by sampling $N_r$ points on region $D\times(0,T]$ by the Latin hypercube sampling (LHS) method. On the initial boundary, i.e. $\{t=0\}\times D$, we sample $N_{ic}$ points by LHS method to construct dataset $X_{ic}$. More precisely, we shall distribute more dense sampling points near the region $\{t=0\}\times\partial\Omega$ because numerical experiments show these areas are more challenging to train well. We construst dataset $X_b$ by sampling $N_b$ points on the spatial boundary $\partial\Omega\times[0,T]$.

$$X_{train}=X_r\cup X_b\cup X_{ic}. \tag{16}$$

The PDE residual is defined on the region $D\times[0,T]$, i.e.

$$r(t,y;\theta):=\frac{\partial v}{\partial t}(t,y;\theta)-\mathcal{L}_{log}[v(t,y;\theta)]. \tag{17}$$

We define the weighted loss function at $k-$th training epoch

$$\mathcal{L}(\theta^k)=\sum_{i=1}^{N_r}\omega_r^{(i)}\mathcal{L}_r^{(i)}(k)+\sum_{i=1}^{N_b}\omega_b^{(i)}\mathcal{L}_b^{(i)}(k)+\sum_{i=1}^{N_{ic}}\omega_{ic}^{(i)}\mathcal{L}_{ic}^{(i)}(k), \tag{18}$$

where

$$\begin{aligned}
\mathcal{L}_r^{(i)}(k)&=|r(t^{(i)},y^{(i)};\theta^k)|^2,(y^{(i)},t^{(i)})\in X_r,\\
\mathcal{L}_{ic}^{(i)}(k)&=|v(y^{(i)},0;\theta^k)-\varphi(y^{(i)})|^2,(y^{(i)},0)\in X_{ic},\\
\mathcal{L}_b^{(i)}(k)&=|v(y^{(i)},t^{(i)};\theta^k)|^2,(y^{(i)},t^{(i)})\in X_b,
\end{aligned} \tag{19}$$

then at next epoch, the weights for datasets $X_f,X_b,X_{ic}$ are defined respectively by

$$\begin{aligned}
\omega_r^{(i)}(k+1)&=\mathcal{L}_r^{(i)}(k)/\sum_{i=1}^{N_r}\mathcal{L}_r^{(i)}(k),\\
\omega_b^{(i)}(k+1)&=\mathcal{L}_b^{(i)}(k)/\sum_{i=1}^{N_b}\mathcal{L}_b^{(i)}(k),\\
\omega_{ic}^{(i)}(k+1)&=\mathcal{L}_{ic}^{(i)}(k)/\sum_{i=1}^{N_{ic}}\mathcal{L}_{ic}^{(i)}(k),
\end{aligned} \tag{20}$$

The weights of different terms are initialized as $\omega_r^{(i)}(1)=\frac{1}{N_r},\omega_b^{(i)}(1)=\frac{1}{N_b},\omega_{ic}^{(i)}(1)=\frac{1}{N_{ic}}$.

To ensure stable training and improve accuracy, we employ practical numerical strategies in this study. We utilize the Adam optimizer [15] for training the neural network, adjusting the learning rate dynamically for faster convergence. The learning rate $\eta_k$ at each iteration $k$ is determined by $\eta_k=\eta_0\times\gamma^k$, where $\eta_0$ is the initial learning rate, $\gamma$ is a regularization factor, and $\eta_k$ is the learning rate used at iteration $k$. Typically, each iteration consists of 50 gradient descent loops, although the number of iterations can be adjusted as needed. Additionally, we set a predefined stop criterion $e_{stop}$ for the training loss, once the loss reaches this threshold, training halts to save time.

### C. Filtering Algorithms

We give a detailed description of the developed filtering method in algorithm 1.

**Algorithm 1** LTDG filtering algorithm

1: **Initialization:** Given GLPs basis $\{\phi_k(x)\}_{k=0}^M$, observation gap time $T_0$, the number of the temporal partition $N$. Config the neural network, and generate training dataset $X_{train}$.
2: **for** $k=0:M$ **do**
3: Solve the equation (14) with $v(z,0)=k\ln z, k=0,\cdots,M+2$ by deep log-FKE solver on the domain $D$.
4: Obtain the solution $\rho^{(k)}(x,T_0)$ of equation (9) with GLP initial condition $\phi_k(x)$ by exponential transformation and linear combination and store it up.
5: **end for**
6: **for** $i=1,\cdots,N$ **do**
7: **Project** $\rho_i(x,\tau_{i-1})$ onto the subspace spanned by GLPs,

$$\rho_i(x,\tau_{i-1})\approx\sum_{k=0}^{N_{glp}}c_{i,k}\phi_k(x), \tag{21}$$

here, $\{c_{i,k}\}_{k=0}^M$ are the coefficients of projection.
8: **Calculate** terminal FKE solution $\rho_i(x,\tau_i)$ by $\{c_{i,k}\}_{k=0}^M$ and the stored solutions $\{\rho^{(k)}(x,T_0)\}_{k=0}^M$, i.e.,

$$\rho_i(x,\tau_i)\approx\sum_{k=0}^{M}c_{i,k}\rho^{(k)}(x,T_0). \tag{22}$$

9: **Estimate** the current state by $u_i(x,\tau_i)$, i.e.

$$\hat{x}(\tau_i)=\frac{\int x\cdot u_i(x,\tau_i)dx}{\int u_i(x,\tau_i)dx}. \tag{23}$$

where the solution to robust-DMZ equation (5) $u_i(x,\tau_i)$ is given by $\rho_i(x,\tau_i)$ and (7), i.e.,

$$u_i(x,\tau_i)=\exp(-h^\top y_{\tau_{i-1}})\rho_i(x,\tau_i) \tag{24}$$

10: **Update** the initial distribution $\rho_{i+1}(x,\tau_i)$ in the next time interval $[\tau_i,\tau_{i+1})$ by

$$\rho_{i+1}(x,\tau_i)=\exp(h^\top(y_{\tau_i}-y_{\tau_{i-1}}))\cdot\rho_i(x,\tau_i). \tag{25}$$

11: **end for**

## IV. Numerical results

In this section, numerical simulation based on the proposed logarithmic transformed deep Galerkin approach (LTDG) algorithm will be tested in several typical filtering systems. LTDG will be compared with the traditional Kalman filter (KF), Extended Kalman filter (EKF), particle filter (PF) [3], and Legendre Spectral method (SM) [8]. The first example is a 1D cubic example with mixed Gaussian initial distribution. The second example is a 2D nonlinear system which has been proven strong nonlinearity in estimation behavior. All simulations are implemented in **Tensorflow** and run in laptop with 12th Gen Intel(R) Core(TM) i9-12900H 2.50 GHz.

So as to measure computational load and accuracy, the following measurement index will be defined which includes MSE (Mean Square Error), MMSE (Mean MSE), and MT (Mean Time).

$$\begin{aligned} \text{MSE}_i :=& \frac{1}{N_{tr}} \sum_{i=1}^{N_{tr}} (\hat{X}_{t_k}^i - X_{t_k}^i)^2 \\ \text{MMSE} :=& \frac{1}{N} \sum_{i=1}^{N} \text{MSE}_i, \\ \text{MT} :=& \frac{1}{N_{tr}} \sum_{i=1}^{N_{tr}} T_i \end{aligned} \tag{26}$$

where $\hat{X}_t^i$ denotes state estimate for a filtering algorithm in $i$-th trial at instant $t$. $X_t^i$ represents real state value in $i$-th trial at instant $t$. $N_{trial}$ denotes number of simulation trials $T_i$ is running time of programming code in $i$-th trial.

### A. 1D Cubic system

$$\begin{cases} dX_t = dW_t, \quad \mathbb{E}[(dW_t)^2] = dt \\ dZ_t = X_t^3 dt + dV_t, \quad \mathbb{E}[(dV_t)^2] = S dt \end{cases} \tag{27}$$

where initial distribution is taken mixed Gaussian density.

$$\sigma_0 \sim 0.5\mathcal{N}(-0.1, 0.2) + 0.5\mathcal{N}(-0.1, 0.3)$$

We take variance $S = 0.03$. Number of polynomial basis functions is chosen 7. The number of training epochs is 10000 in a multilayer perceptron (MLP). Initial learning rate $\eta_0 = 0.01$. Batch sizes $N_r = 1000, N_b = 900, N_{ic} = 800$. The nonlinear activation function is chosen as $\tanh$. Network architecture is chosen as $[2, 100, 100, 100, 1]$.

From a state tracking perspective, Fig. 1 (left) shows that LTDG outperforms other algorithms. PF and EKF exhibit high oscillations, making them inefficient for estimation, with EKF performing the worst. Based on the Yau-Yau framework, the Spectral Method (SM) can achieve optimal estimation with enough basis functions but often incurs large computational loads due to the need to solve the Yau-Yau equation at each time step via the Galerkin method. Despite this, SM remains a real-time algorithm in numerous examples. LTDG, on the other hand, demonstrates the fastest CPU processing time, reducing computational time by over 50% compared to SM while maintaining the same accuracy.

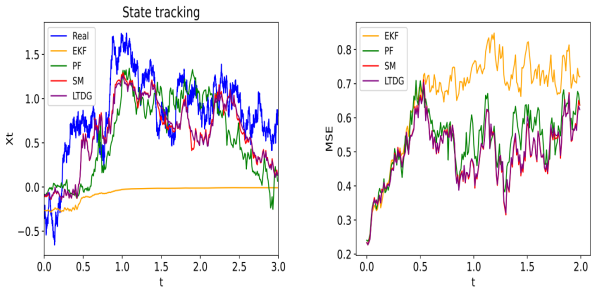


Fig. 1. **1D Cubic filter.** (Left) State tracking. (Right) Mean square error.

TABLE I
Performance of all simulated algorithms in 1D cubic system.

| Algorithms | LTDG | SM | EKF | PF |
|---|---|---|---|---|
| MMSE | 0.5050 | 0.5071 | 0.6188 | 0.5176 |
| CPU time(s) | 0.0668 | 0.1962 | 0.0094 | 0.2837 |

### B. 2D cubic system

$$\begin{cases} dX_t = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} X_1 \\ X_2 \end{bmatrix} dt + dW_t, \\ dZ_t = \begin{bmatrix} X_1^3 \\ X_2^3 \end{bmatrix} dt + dV_t, \quad \mathbb{E}[dV_t dV_t^\top] = S dt, \\ X_0 \sim \mathcal{N}([0.1, 0.1]^\top, 0.05 I_2) \end{cases} \tag{28}$$

where the drift coefficients $a_{11} = 0.2, a_{12} = 0.3, a_{21} = -0.3, a_{22} = 0.1$, $W_t$ is a standard Brownian motion process, the covariance coefficient matrix is set $S = 0.1I$ for simplicity. The number of GLPs $M = 15$. For LTDG, the architecture of the neural network is $[3, 100, 100, 100, 1]$. Other parameter setting is the same as in the previous example.

TABLE II
Performance of all simulated algorithms in 2D cubic system.

| Algorithms | LTDG | SM | EKF | PF |
|---|---|---|---|---|
| MMSE | 0.5650 | 0.5387 | 1.0069 | 0.8662 |
| CPU time(s) | 0.3614 | 0.7408 | 0.0102 | 0.8268 |

Fig. 2 illustrates the performance of various algorithms in comparison to the actual state. The upper two subfigures in Fig. 2 depict the accuracy of state tracking, while the lower two subfigures present the mean squared error (MSE) over time. It is evident that the EKF fails to accurately track the state for both variables $x_1$ and $x_2$. Although the PF performs better than the EKF, it still falls short of achieving effective state tracking. The SM provides the optimal estimate for this nonlinear system; however, its computational load is nearly twice that of our newly proposed method, LTDG.

## V. Conclusion

In this paper, we propose a novel filtering algorithm combining deep neural networks, the Galerkin-spectral approach, and gauge logarithmic transformation. Specifically,

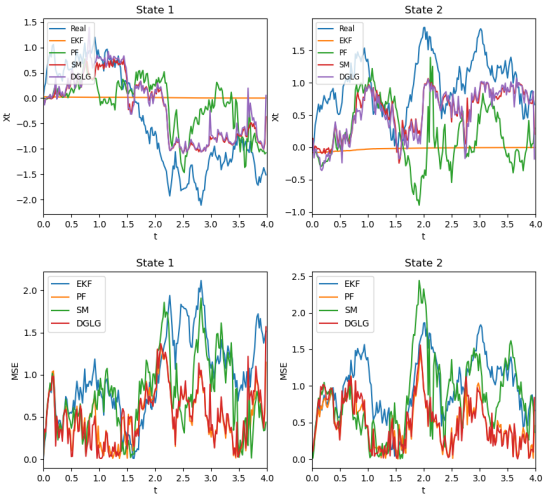


Fig. 2. **2D Cubic filter.** State tracking (upper two subfigures) and MSE (lower two subfigures)in the 2D cubic system.

we efficiently solve the observation-independent Forward Kolmogorov Equation (FKE) using deep neural networks within the PINN paradigm. To address fast inference under various initial conditions, we approximate the unnormalized density function with GLPs. The method inherits the benefits of the Yau-Yau filtering algorithm and requires minimal assumptions on the filtering system, making it widely applicable. We validate the algorithm's effectiveness through numerical experiments, showing improved stability and accuracy compared to traditional methods like EKF, LSM, and PF. However, the algorithm is limited by the "curse of dimensionality" particularly in higher-dimensional systems. Future work will focus on designing efficient forward Kolmogorov equation operator networks to reduce offline training costs.

## REFERENCES


[1] Brian D. O. Anderson and John B. Moore. *Optimal filtering*. Courier Corporation, 2012.

[2] Ienkaran Arasaratnam and Simon Haykin. Cubature kalman filters. *IEEE Transactions on automatic control*, 54(6):1254–1269, 2009.

[3] M Sanjeev Arulampalam, Simon Maskell, Neil Gordon, and Tim Clapp. A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking. *IEEE Transactions on signal processing*, 50(2):174–188, 2002.

[4] VE Beneš. Exact finite-dimensional filters for certain diffusions with nonlinear drift. *Stochastics: An International Journal of Probability and Stochastic Processes*, 5(1-2):65–92, 1981.

[5] Silvere Bonnable, Philippe Martin, and Erwan Salaün. Invariant extended kalman filter: theory and application to a velocity-aided attitude estimation problem. In *Proceedings of the 48h IEEE Conference on Decision and Control (CDC) held jointly with 2009 28th Chinese Control Conference*, pages 1297–1304. IEEE, 2009.

[6] Giuseppe Calafiore. Reliable localization using set-valued nonlinear filters. *IEEE Transactions on systems, man, and cybernetics-part A: systems and humans*, 35(2):189–197, 2005.

[7] Xiuqiong Chen, Ji Shi, and Stephen S-T Yau. Real-time solution of time-varying Yau filtering problems via direct method and Gaussian approximation. *IEEE Transactions on Automatic Control*, 64(4):1648–1654, 2019.

[8] Wenhui Dong, Xue Luo, and Stephen S.-T. Yau. Solving nonlinear filtering problems in real time by legendre galerkin spectral method. *IEEE Transactions on Automatic Control*, 66(4):1559–1572, 2021.

[9] Tyrone Edward Duncan. *Probability densities for diffusion processes with applications to nonlinear filtering theory and detection theory*. Stanford University, 1967.

[10] Geir Evensen. Sequential data assimilation with a nonlinear quasi-geostrophic model using monte carlo methods to forecast error statistics. *Journal of Geophysical Research: Oceans*, 99(C5):10143–10162, 1994.

[11] Neil J Gordon, David J Salmond, and Adrian FM Smith. Novel approach to nonlinear/non-gaussian bayesian state estimation. *IEE proceedings F (radar and signal processing)*, 140(2):107–113, 1993.

[12] Kazufumi Ito and Kaiqi Xiong. Gaussian filters for nonlinear filtering problems. *IEEE transactions on automatic control*, 45(5):910–927, 2000.

[13] Simon Julier, Jeffrey Uhlmann, and Hugh F Durrant-Whyte. A new method for the nonlinear transformation of means and covariances in filters and estimators. *IEEE Transactions on automatic control*, 45(3):477–482, 2000.

[14] Rudolph Emil Kalman. A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, 82(1):35–45, 1960.

[15] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[16] Deok-Jin Lee. *Nonlinear Bayesian filtering with applications to estimation and navigation*. Texas A&M University, 2005.

[17] Xiao-Rong Li and Vesselin P Jilkov. A survey of maneuvering target tracking: approximation techniques for nonlinear filtering. In *Signal and Data Processing of Small Targets 2004*, volume 5428, pages 537–550. SPIE, 2004.

[18] X. Luo and S. S.-T. Yau. Hermite spectral method to 1-D forward Kolmogorov equation and its application to nonlinear filtering problems. *IEEE Transactions on Automatic Control*, 58:2495–2507, 2013.

[19] R. E. Mortensen. *Optimal control of continuous time stochastic systems*. PhD thesis, University of California, Berkley, California, Aug. 1966.

[20] Maziar Raissi, Paris Perdikaris, and George E Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational physics*, 378:686–707, 2019.

[21] Gerasimos G Rigatos. *Nonlinear control and filtering using differential flatness approaches: applications to electromechanical systems*, volume 25. Springer, 2015.

[22] Jie Shen. Efficient spectral-galerkin method i. direct solvers of second- and fourth-order equations using legendre polynomials. *SIAM Journal on Scientific Computing*, 15(6):1489–1505, 1994.

[23] Ji Shi, Zhiyu Yang, and Stephen S-T Yau. Direct method for Yau filtering system with nonlinear observations. *International Journal of Control*, 91(3):678–687, 2018.

[24] Tao Yang, Prashant G Mehta, and Sean P Meyn. Feedback particle filter. *IEEE transactions on Automatic control*, 58(10):2465–2480, 2013.

[25] S.-T. Yau and S. S.-T. Yau. Real time solution of the nonlinear filtering problem without memory II. *SIAM Journal on Control and Optimization*, 47(1):163–195, 2008.

[26] Shing-Tung Yau and Stephen S-T Yau. Real time solution of nonlinear filtering problem without memory I. *Mathematical Research Letters*, 7(6):671–693, 2000.

[27] M. Zakai. On the optimal filtering of diffusion process. *Z. Wahrsch. verw. Gebiete*, 11(3):230–243, 1969.