Journal of Computational Physics 514 (2024) 113207

Contents lists available at ScienceDirect

Journal of Computational Physics

journal homepage: www.elsevier.com/locate/jcp

A score-based filter for nonlinear data assimilation

Feng Bao [a], Zezhong Zhang [b], Guannan Zhang [b,*]

[a] Department of Mathematics, Florida State University, Tallahassee, FL 32306, United States of America
[b] Computer Science and Mathematics Division, Oak Ridge National Laboratory, Oak Ridge, TN 37831, United States of America

ARTICLE INFO

Keywords:
Nonlinear filtering
Diffusion model
Score-based models
Stochastic dynamical systems
Kalman filter
Particle filter

ABSTRACT

We propose a score-based generative sampling method for solving the nonlinear filtering problem with superior accuracy. A major drawback of existing nonlinear filtering methods, e.g., particle filters, is the low accuracy in handling high-dimensional nonlinear problems. To overcome this issue, we incorporate the score-based diffusion model into the recursive Bayesian filter framework to develop a novel score-based filter (SF). The key idea of SF is to store the information of the recursively updated filtering density function in the score function, instead of storing the information in a set of finite Monte Carlo samples (used in particle filters and ensemble Kalman filters). By leveraging the reverse-time diffusion process, SF can generate unlimited samples to characterize the filtering density. An essential aspect of SF is its analytical update step, gradually incorporating data information into the score function. This step is crucial in mitigating the degeneracy issue faced when dealing with very high-dimensional nonlinear filtering problems. Three benchmark problems are used to demonstrate the performance of our method. In particular, SF provides surprisingly impressive performance in reliably capturing/tracking the 100-dimensional stochastic Lorenz system that is a well-known challenging problem for existing filtering methods.

1. Introduction

Nonlinear filtering represents a significant avenue of research in data assimilation, encompassing a wide range of applications in weather forecasting, military operations, material sciences, biology, and finance [1,3,6,8,12,15,18,32,35]. The primary objective of addressing a filtering problem lies in leveraging partially noisy observational data streams to estimate the unobservable state of a stochastic dynamical system of interest. In linear filtering, where both the state and observation dynamics are linear, the Kalman filter provides an optimal estimate for the unobservable state, attainable analytically under the Gaussian assumption.

When dealing with nonlinear dynamical systems, the standard Kalman filter becomes impractical. An extension of the Kalman filter known as the ensemble Kalman filter can be employed to tackle the nonlinearity to a certain extent. The fundamental concept behind the ensemble Kalman filter is to utilize an ensemble of Kalman filter samples to describe the probability distribution of the target state in the form of a Gaussian distribution. As a Kalman type filter, the ensemble Kalman filter stores the information of the state variable as the mean and the covariance of Kalman filter samples. Consequently, the probability density function (PDF) of the target state, which is often referred to as the filtering density, is approximated by a Gaussian distribution. However, in nonlinear

* Corresponding author.
E-mail address: zhangg@ornl.gov (G. Zhang).

https://doi.org/10.1016/j.jcp.2024.113207
Received 1 August 2023; Received in revised form 25 March 2024; Accepted 14 June 2024

Available online 19 June 2024

filtering problems, the filtering density is usually non-Gaussian. Therefore, the ensemble Kalman filter, which still relies on the Gaussian assumption, is not the ideal approach to solve the nonlinear filtering problem [1,38].

In addition to the ensemble Kalman filter, several effective nonlinear filtering methods have been developed to tackle nonlinearity in various applications. These methods include the particle filter [4,19], the Zakai filter [7,45] and so on. Among these approaches, the particle filter stands out as the most widely applied solution for addressing the nonlinear filtering problem. Also known as sequential Monte Carlo, the particle filter employs a collection of Monte Carlo samples, referred to as particles, to construct an empirical distribution that characterizes the conditional probability distribution for the target state. In this work, we slightly misuse the terminology and continue to use "filtering density" to denote the conditional distribution for the state variable. Upon receiving observational data, the particle filter employs a Bayesian inference procedure to assign likelihood weights to the particles. Subsequently, a resampling process is iteratively performed, generating duplicates of particles with large weights while discarding particles with small weights. This resampling technique ensures that the particle filter adapts well to the nonlinearity present in the optimal filtering problem. By utilizing particle simulations, the filter incorporates the nonlinear state dynamics into the filtering density. Simultaneously, the Bayesian inference serves as a standard approach to handle nonlinear observations. In contrast to Kalman type filters, which capture information about the filtering density through mean and covariance, the particle filter stores data about the filtering density in the positions of its numerous particles. As a result, the particle filter excels at characterizing more complex non-Gaussian filtering densities through the empirical distributions constructed from the particles. This unique feature of the particle filter allows it to handle a broader range of challenging filtering scenarios compared to traditional Kalman filters.

The main drawback of the particle filter is the low accuracy in handling high-dimensional nonlinear problems. In the particle filter, once the total number of particles is fixed, the capability to characterize the filtering density is fixed, and one may only use those finite particles to approximate the filtering density. However, the nonlinear state dynamics and nonlinear observations could result in unpredictable features of the filtering densities. Therefore, it's hard to use finite particles to characterize unlimited possibilities of filtering densities. This often causes the so-called degeneracy issue, i.e., the amount of particles in high probability regions are not sufficient to characterize highly probable features in filtering densities. Such a degeneracy issue is even more prohibitive when the dimension of the problem is high due to the curse of dimensionality. Although advanced resampling methods are proposed to address the degeneracy issue by relocating particles to high probability regions [4,16,24,34,38], the nature of finite particle representation for the filtering density cannot be changed under the sequential Monte Carlo framework, and the information that any re-sampling method may use cannot exceed the information carried by those finite particles.

In this work, we introduce a novel score-based filter (SF) that allows to use a score function to generate samples from the filtering density through a diffusion process. The score function is a key component in diffusion model, which is a well-known generative machine learning model for generating samples from a target PDF. Diffusion models are generative models that utilize noise injection to progressively distort data and then learn to reverse this process for sample generation. As a category of deep generative models, diffusion models are widely used in image processing applications, such as image synthesis [17,21,40,41,13,22,33], image denoising [25,30,21,39], image enhancement [26,27,36,43], image segmentation [2,10,11,20], and natural language processing [5,23,29,37,44]. Diffusion models are also capable of density estimation (i.e., learning how to draw samples from the probability distribution). Specifically, a diffusion model can transport a prior distribution, which is often chosen as the standard Gaussian distribution, to a complex target data distribution through a reverse-time diffusion process in the form of a stochastic differential equation, and the score function is the forcing term that guides the reverse-time diffusion process towards the data distribution. Since the prior distribution is independent of the target data distribution, the information of the data distribution is stored in the score function.

The key idea of SF is to store the information of the recursively updated filtering density function in the score model, instead of storing the information in a set of finite Monte Carlo samples used in particle filters and ensemble Kalman filters. Specifically, we propagate Monte Carlo samples through the state dynamics to generate data samples that follow the filtering density, and use the data samples to train a score function defined by a deep neural network. Although samples that characterize the filtering density are still needed in our method, SF is essentially different from existing Monte Carlo based filtering methods. Using the reverse-time diffusion sampler, we can generate unlimited samples to characterize the filtering density. Moreover, when the score function is approximated by a deep neural network [41], SF has the potential to handle very high-dimensional nonlinear filtering problems.

The rest of this paper is organized as follows. In Section 2, we briefly introduce the nonlinear filtering problem and its state-of-the-art solver, i.e., the particle filter method. In Section 3, we provide a comprehensive discussion to develop our score-based filter method. In Section 4, we carry out numerical experiments for three benchmark nonlinear filtering problems, which include a 100-dimensional stochastic Lorenz system, to demonstrate the superior performance of the score-based filter.

2. Problem setting

Nonlinear filters are important tools for dynamical data assimilation with a variety of scientific and engineering applications. The definition of a nonlinear filtering problem can be viewed as an extension of Bayesian inference to the estimation and prediction of a nonlinear stochastic dynamical system. In this effort, we consider the following state-space nonlinear filtering model:

$$\begin{aligned} \textbf{State:} \quad & X_{t+1} = f(X_t, \omega_t), \\ \textbf{Observation:} \quad & Y_{t+1} = g(X_{t+1}) + \epsilon_{t+1}, \end{aligned} \tag{1}$$

where $t \in \mathbb{Z}^+$ represents the discrete time, $X_t \in \mathbb{R}^d$ is a d-dimensional unobservable dynamical state governed by the nonlinear function $f : \mathbb{R}^d \times \mathbb{R}^k \mapsto \mathbb{R}^d$, $\omega_t \in \mathbb{R}^k$ is a random variable that follows a given probability law representing the uncertainty in f,

and the random variable $Y_{t+1} \in \mathbb{R}^r$ provides nonlinear partial observation on X_{t+1}, i.e., $g(X_{t+1})$, perturbed by a Gaussian noise $\varepsilon_{t+1} \sim \mathcal{N}(0, \Sigma)$.

The overarching goal is to find the best estimate, denoted by $\hat{X}_{t+1}$, of the unobservable state X_{t+1}, given the observation data $\mathcal{Y}_{t+1} := \sigma(Y_{1:t+1})$ that is the σ-algebra generated by all the observations up to the time instant $t+1$. Mathematically, such optimal estimate for X_{t+1} is usually defined by a conditional expectation, i.e.,

$$\hat{X}_{t+1} := \mathbb{E}[X_{t+1}|\mathcal{Y}_{t+1}], \tag{2}$$

where the expectation is taken with respect to the random variables $\omega_{0:t}$ and $\varepsilon_{1:t+1}$ in Eq. (1). When solving nonlinear filtering problems, where the conditional distribution for the state variable is often non-Gaussian, our objective is to approximate the conditional probability density function (PDF) of the state, denoted as $P(X_{t+1}|\mathcal{Y}_{t+1})$. In this context, we refer to this distribution as the "filtering density." The Bayesian filter framework is to recursively incorporate observation data to describe the evolution of the filtering density. There are two steps from time t to $t+1$, i.e., the prediction step and the update step:

- *The prediction step* is to use the Chapman-Kolmogorov formula to propagate the state equation in Eq. (1) from t to $t+1$ and obtain the prior filtering density, i.e.,

$$\textbf{Prior filtering density:} \quad P(X_{t+1}|\mathcal{Y}_t) = \int P(X_{t+1}|X_t)P(X_t|\mathcal{Y}_t)dX_t, \tag{3}$$

  where $P(X_t|\mathcal{Y}_t)$ is the posterior filtering density obtained at the time instant t, $P(X_{t+1}|X_t)$ is the transition probability derived from the state dynamics in Eq. (1), and $P(X_{t+1}|\mathcal{Y}_t)$ is the prior filtering density for the time instant $t+1$.
- *The update step* is to combine the likelihood function, defined by the new observation data Y_{t+1}, with the prior filtering density to obtain the posterior filtering density, i.e.,

$$\textbf{Posterior filtering density:} \quad P(X_{t+1}|\mathcal{Y}_{t+1}) \propto P(X_{t+1}|\mathcal{Y}_t)\, P(Y_{t+1}|X_{t+1}), \tag{4}$$

  where the likelihood function $P(Y_{t+1}|X_{t+1})$ is defined by

$$P(Y_{t+1}|X_{t+1}) \propto \exp\left[-\frac{1}{2}\big(g(X_{t+1}) - Y_{t+1}\big)^\top \Sigma^{-1}\big(g(X_{t+1}) - Y_{t+1}\big)\right], \tag{5}$$

  with Σ being the covariance matrix of the random noise ε in Eq. (1).

In this way, the filtering density is predicted and updated through formulas Eq. (3) to Eq. (4) recursively in time. Note that both the prior and the posterior filtering densities in Eq. (3) and Eq. (4) are defined as the continuum level, which is not practical. Thus, one important research direction in nonlinear filtering is to study how to accurately approximate the prior and the posterior filtering densities.

2.1. The state of the art: particle filters

Particle filters (PF), which is the state of the art in nonlinear filtering, approximate the filtering densities in Eq. (3) and Eq. (4) using empirical distributions defined by a set of random samples, referred to as "particles". To compare with the proposed score-based filter in Section 3, we briefly recall how particle filters use random samples to approximate the filtering densities. At the time instant t, we assume that we have a set of M particles, denoted by $\{x_{t,m}\}_{m=1}^M$, that follows the posterior filtering density $P(X_t|\mathcal{Y}_t)$. The empirical distribution for approximating $P(X_t|\mathcal{Y}_t)$ is given by

$$P(X_t|\mathcal{Y}_t) \approx P^{\mathrm{PF}}_{t|t}(X_t) := \frac{1}{M}\sum_{m=1}^M \delta_{x_{t,m}}(X_t), \tag{6}$$

where $\delta_{x_{t,m}}$ is the Dirac delta function at the m-th particle $x_{t,m}$. In practice, PF is implemented through the following 3-step procedure to propagate from time t to $t+1$:

- *The prediction step.* It is to propagate the particle cloud through the state dynamics and generate a set of predicted particles. For each particle $x_{t,m}$ that represents state X_t, we run the state equation in Eq. (1) to obtain a predicted particle $\tilde{x}_{t+1,m} = f(x_{t,m}, \omega_{t,m})$, where $\omega_{t,m}$ is a sample of the random variable ω_t. As a result, we obtain a set of particles $\{\tilde{x}_{t+1,m}\}_{m=1}^M$ and the corresponding empirical distribution

$$P(X_{t+1}|\mathcal{Y}_t) \approx P^{\mathrm{PF}}_{t+1|t}(X_{t+1}) := \frac{1}{M}\sum_{m=1}^M \delta_{\tilde{x}_{t+1,m}}(X_{t+1}), \tag{7}$$

  which approximates the prior filtering density $P(X_{t+1}|\mathcal{Y}_t)$ in Eq. (3).
- *The update step.* It is to incorporates the new observational data Y_{t+1} through Bayesian inference to update the prior filtering density $P^{\mathrm{PF}}_{t+1|t}(X_{t+1})$ to the posterior filtering density $\tilde{P}^{\mathrm{PF}}_{t+1|t+1}(X_{t+1})$, i.e.,

$$P(X_{t+1}|\mathcal{Y}_{t+1}) \approx \tilde{P}^{\mathrm{PF}}_{t+1|t+1}(X_{t+1}) := \sum_{l=1}^{L} w_{t+1,m}\delta_{\tilde{x}_{t+1,m}}(X_{t+1}), \tag{8}$$

where $\delta_{\tilde{x}_{t+1,m}}$ is the Dirac delta function at the m-th predicted particle $\tilde{x}_{t+1,m}$, and the weight $w_{t+1,m} \propto P(M_{t+1}|\tilde{x}_{t+1,m})$ is defined by the likelihood function $P(M_{t+1}|X_{t+1})$ at the m-th predicted particle $\tilde{x}_{t+1,m}$.

- *The resampling step.* It is to alleviate the degeneracy issue, in which only a few particles have significant weights while the weights on other particles maybe neglectable. The resampling is often implemented to re-generate a set of equally weighted particles that follows the weighted empirical distribution $\tilde{P}^{\mathrm{PF}}_{t+1|t+1}(X_{t+1})$. We denote the resampled equally weighted particles by $\{x_{t+1,m}\}_{m=1}^{M}$, which formulate the empirical distribution

$$P(X_{t+1}|\mathcal{Y}_{t+1}) \approx P^{\mathrm{PF}}_{t+1|t+1}(X_{t+1}) := \frac{1}{M}\sum_{m=1}^{M} \delta_{x_{t+1,m}}(X_{t+1}), \tag{9}$$

which is the final approximation of the posterior filtering density at the time instant $t+1$.

The challenge of particle filters is the so-called degeneracy issue, especially for high-dimensional nonlinear filtering problems or long-term tracking problems. In these scenarios, the likelihood weights $w_{t+1,m}$ in Eq. (8) tend to concentrate on a very small number of particles. As a result, only a few number of particles are taken into account to construct the approximate posterior filtering density in Eq. (9), and the information of the prior filtering density stored in the particles with small weights is gradually ignored. The main reason causing the degeneracy issue is the use of a discrete approximation (i.e., the empirical distributions) based on a finite number of pre-chosen samples to characterize the continuous filtering densities in Eq. (3) and Eq. (4). Due to the "curse of dimensionality", a distribution in a high-dimensional space contains enormous information, and it's very difficult for finite amount of particles to capture the continuous characterization of the filtering densities in high-dimensional spaces. This challenge motivated us to exploit recent advances in diffusion models to develop a score-based nonlinear filter that uses a continuous score function to indirectly store the information of the filtering densities.

3. Our method: the score-based filter (SF)

This section contains the key components of the proposed method. The score-based diffusion model is briefly introduced in Section 3.1. We introduce the details of the score-based filter in Section 3.2 with the implementation details provided in Appendix A.

3.1. The score-based diffusion model

The diffusion model is a type of generative machine learning models for generating samples from a target probability density function, denoted by

$$Q(Z) \quad \text{for} \quad Z \in \mathbb{R}^d, \tag{10}$$

where Z is a d-dimensional random variable. The key idea is to transform the unknown density $Q(Z)$ to a standard probability distribution, e.g., the standard Gaussian $\mathcal{N}(0, \mathbf{I}_d)$. To this end, a diffusion model first defines a forward stochastic differential equation (SDE), i.e.,

$$\textbf{Forward SDE:} \quad Z_\tau = b(\tau)Z_\tau d\tau + \sigma(\tau)dW_\tau \quad \text{for } \tau \in \mathcal{T} = [0,1], \tag{11}$$

where $\mathcal{T} = [0,1]$ is a pseudo-temporal domain that is different from the real temporal domain in which the nonlinear filtering problem is defined, W_τ is a standard d-dimensional Brownian motion, $b : \mathcal{T} \mapsto \mathbb{R}$ is the drift coefficient, $\sigma : \mathcal{T} \mapsto \mathbb{R}$ the diffusion coefficient, and the solution $\{Z_\tau\}_{0\le\tau\le1}$ is a diffusion process that takes values in $\mathbb{R}^d$.

The probability density function of the forward process Z_τ is denoted by

$$Q_\tau(Z_\tau) \quad \text{for} \quad \tau \in [0,1]. \tag{12}$$

With a proper definition of $b(\tau)$ and $\sigma(\tau)$ (e.g., see Appendix A), the forward SDE in Eq. (11) can transform any initial distribution of $Q_0(Z_0)$ to a standard Gaussian variable $Q_1(Z_1) = \mathcal{N}(0, \mathbf{I}_d)$. Therefore, when we set the initial state to be the target random variable, i.e., $Z_0 = Z$ in Eq. (11), the forward SDE can transform the target distribution $Q(Z)$ to the standard Gaussian distribution $\mathcal{N}(0, \mathbf{I}_d)$.

It is easy to see that the forward SDE cannot be used to generate samples of the target distribution $Q(Z)$. To do this, the score-based diffusion model shows that there is an equivalent reverse-time SDE to transform the terminal distribution $Q_1(Z_1) = \mathcal{N}(0, \mathbf{I}_d)$ to the initial distribution $Q_0(Z_0)$, i.e.,

$$\textbf{Reverse-time SDE:} \quad dZ_\tau = \left[b(\tau)Z_\tau - \sigma^2(\tau)S(Z_\tau,\tau)\right]d\tau + \sigma(\tau)d\bar{W}_\tau, \tag{13}$$

where $\bar{W}_\tau$ is the backward Brownian motion and $S(Z_\tau,\tau)$ is referred to as the score function

$$\textbf{Score function:} \quad S(Z_\tau,\tau) := \nabla_z \log Q_\tau(Z_\tau), \tag{14}$$

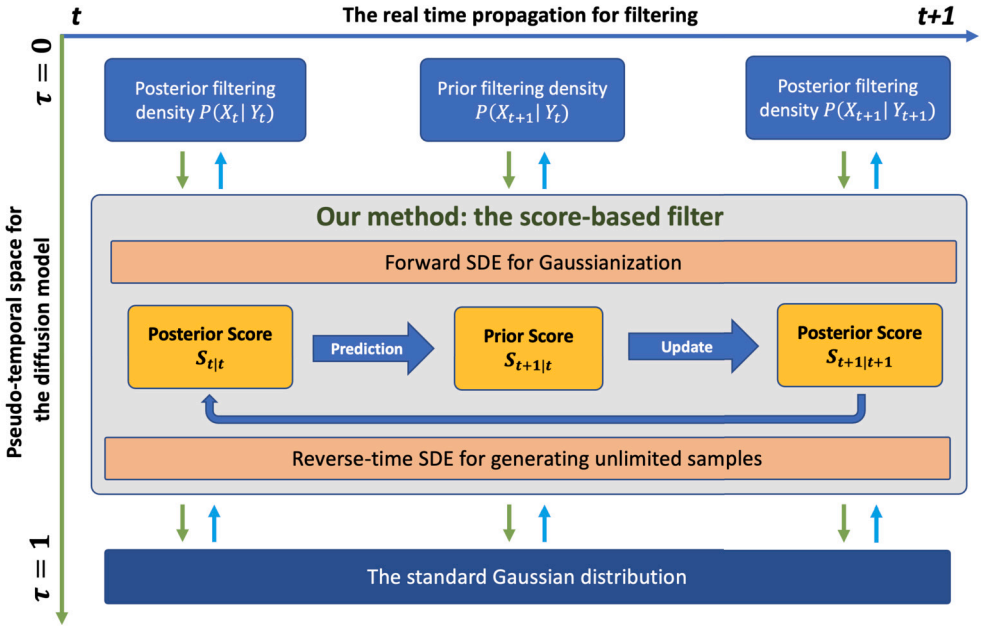


Fig. 1. The schematic overview of the proposed SF method. The key idea is to store the information of the recursively updated filtering density, i.e., $S_{t|t}, S_{t+1|t}$ and $S_{t+1|t+1}$, in the score function, instead of storing the information in a set of finite Monte Carlo samples (used in particle filters and ensemble Kalman filters). The reverse-time SDE uses the score function to generate unlimited number of samples of the current filtering density by simulating Eq. (13) in the pseudo-temporal space. Another key ingredient of SF the analytical update step $S_{t+1|t}$ to $S_{t+1|t+1}$ (see Eq. (20)) to mitigate the degeneracy issue.

that is uniquely determined by the initial distribution $Q_0(Z_0)$ and the coefficients $b(\tau)$, $\sigma(\tau)$. Note that the SDE in Eq. (13) is solved from $\tau = 1$ to $\tau = 0$. In this way, if the score function is given, we can easily generate samples from the target distribution $Q(Z)$ by generating samples from $Q_1(Z_1) = \mathcal{N}(0, \mathbf{I}_d)$ and then solving the reverse-time SDE.

As such, the problem of generating samples becomes a problem of how to approximate the score function. In practice, the score function can be estimated by training a score-based model on samples with score matching (Hyvarinen, 2005; Song et al., 2019a). To this end, we train a time-dependent parameterized score-based model, denoted by $\bar{S}(Z_\tau, \tau; \theta)$, to approximate the exact score function by solving the following optimization problem:

$$\hat{\theta} = \arg\min_{\theta} \mathbb{E}\left[\|S(Z_\tau, \tau) - \bar{S}(Z_\tau, \tau; \theta)\|_2^2\right], \tag{15}$$

where θ denotes the set of the tuning parameters (e.g., neural network weights) for the approximate score function. The implementation details about the loss function are given in Appendix A.3. Once the approximate score function $\bar{S}(Z_\tau, \tau; \theta)$ is well trained, it can be substituted into the Reverse-time SDE to generate any number of samples from the target distribution.

3.2. The methodology of the score-based filter

The key idea of the proposed score-based filter is to treat the prior and posterior filtering densities in Eq. (3) and Eq. (4) as the target distribution $Q(Z)$ in Eq. (10) (or the initial distribution $Q_0(Z_0)$ in Eq. (12)) in the diffusion model and utilize the score-driven reverse-time SDE to approximate the filtering densities. In other words, we store the information of the filtering densities in the continuous score function. A systemic overview of the score-based filter is given in Fig. 1.

3.2.1. The relation between the diffusion model and the filtering densities

Here we discuss how to store the information of the filtering densities in the corresponding score function and how to use the score function to generate unlimited samples of the filtering densities. To proceed, we define the notation

$$S_{t|t}(Z_\tau, \tau; \theta) \text{ with } \tau \in [0, 1], \tag{16}$$

to represent the exact score function of the posterior filtering density $P(X_t|\mathcal{Y}_t)$ at the time instant t. The diffusion model is related to the filtering density by having the filtering state X_t equal to the initial state Z_0 in the forward and reverse-time SDEs in Eq. (11) and Eq. (13),

$$Z_0 = X_t \Longrightarrow Q_0(Z_0) = P(X_t|\mathcal{Y}_t), \tag{17}$$

where $Q_0(Z_0)$ is the initial distribution of the diffusion model. As shown in Appendix A, the choice of $b(\tau)$ and $\sigma(\tau)$ in Eq. (11) can ensure that the diffusion model can transform any initial distribution Q_0 to the standard Gaussian distribution, we can see that the score function implicitly defines an invertible mapping between the filtering density and the standard Gaussian density, i.e.,

$$\Pi_{S_{t|t}}\big(\mathcal{N}(0,\mathbf{I}_d)\big) = P(X_t|\mathcal{Y}_t) \quad \text{and} \quad \Pi^{-1}_{S_{t|t}}\big(P(X_t|\mathcal{Y}_t)\big) = \mathcal{N}(0,\mathbf{I}_d), \tag{18}$$

which indicates that the complete information of the filtering densities can be stored in the score function. This is the key property we will exploit to develop the score-based filter. In Section 3.2.2 and 3.2.3, we will discuss the procedure of dynamically updating the approximate score function $\bar{S}(Z_\tau,\tau;\theta)$.

3.2.2. The prediction step of the score-based filter

We intend to evolve the score function $\bar{S}_{t|t}$ associated with $P(X_t|\mathcal{Y}_t)$ to the score function $\bar{S}_{t+1|t}$ associated with the prior filtering density $P(X_{t+1}|\mathcal{Y}_t)$. To achieve this, the prediction step consists of three stages:

- Drawing J samples from $P(X_t|\mathcal{Y}_t)$ by solving the reverse-time SDE in Eq. (13) using the score function $\bar{S}_{t|t}$. The samples are denoted by $\{x_{t,j}\}_{j=1}^J$. Unlike the particle filter, we can draw unlimited amount of samples using the diffusion model.
- Run the state equation in Eq. (1) to obtain predicted samples $\tilde{x}_{t+1,j} = f(x_{t,j},\omega_{t,j})$, where $\omega_{t,j}$ is a sample of the random variable ω_t.
- Update the score function $\bar{S}_{t|t}$ to $\bar{S}_{t+1|t}$ for the prior filtering density $P(X_{t+1}|\mathcal{Y}_t)$ using the sample set $\{\tilde{x}_{t+1,j}\}_{j=1}^J$ by solving the optimization problem in Eq. (15).

One may notice that scheme $\tilde{x}_{t+1,j} = f(x_{t,j},\omega_{t,j})$ is similar to the prediction scheme in Eq. (7) in the particle filter, and $\{\tilde{x}_{t+1,j}\}_{j=1}^J$ form a set of samples for the prior filtering density. However, it's important to recall that the score-based filtering stores the information of the target filtering density in the *score function* instead of the *finite* (or discrete) locations of the particles as in the particle filter, and we can generate unlimited number of data samples through the reverse-time SDE as needed to characterize the target distribution. Therefore, the number J in our score-based filter is an arbitrarily chosen number. In this way, as long as the exact score function is well approximated, which is typically obtained through deep learning, we can generate as many samples as needed to pass into the state dynamical model and create a distribution for the predicted filtering density as smooth as we want. On the other hand, once the number of total particles is chosen at the beginning of a particle filter algorithm, the filtering density can be only characterized by the finite locations of those particles.

3.2.3. The update step of the score-based filter

We intend to update the score function $\bar{S}_{t+1|t}$ corresponding to the prior filtering density to the score function $\bar{S}_{t+1|t+1}$ corresponding to the posterior filtering density. Unlike the prediction step in which we can generate unlimited samples $\{\tilde{x}_{t+1,j}\}_{j=1}^J$ to help us train the score function $\bar{S}_{t+1|t}$, we do not have access to samples from the posterior filtering density. Therefore, we propose to analytically add the likelihood information to the current score $\bar{S}_{t+1|t}$ to define the score $\bar{S}_{t+1|t+1}$ for the posterior filtering density $P(X_{t+1}|\mathcal{Y}_{t+1})$.

Specifically, we take the gradient of the log likelihood of the posterior filtering density defined in Eq. (4) and obtain,

$$\nabla_x \log P(X_{t+1}|\mathcal{Y}_{t+1}) = \nabla_x \log P(X_{t+1}|\mathcal{Y}_t) + \nabla_x \log P(Y_{t+1}|X_{t+1}), \tag{19}$$

where the gradient is taken with respect to the state variable at X_{t+1}. According to the discussion in Section 3.2.1, the exact score functions $S_{t+1|t+1}$ and $S_{t+1|t}$ satisfy the following constraints:

(C1): $S_{t+1|t}(Z_0,0) = \nabla_x \log P(X_{t+1}|\mathcal{Y}_t)$ and $S_{t+1|t+1}(Z_0,0) = \nabla_x \log P(X_{t+1}|\mathcal{Y}_{t+1})$,
(C2): $\Pi^{-1}_{S_{t+1|t}}\big(P(X_{t+1}|\mathcal{Y}_t)\big) = \mathcal{N}(0,\mathbf{I}_d)$ and $\Pi^{-1}_{S_{t+1|t+1}}\big(P(X_{t+1}|\mathcal{Y}_{t+1})\big) = \mathcal{N}(0,\mathbf{I}_d)$,

when setting $Z_0 = X_{t+1}$. We combine the Eq. (19) and the constraints (**C1**), (**C2**) to propose an approximation of $S_{t+1|t+1}$ of the form

$$\bar{S}_{t+1|t+1}(Z_\tau,\tau;\theta) := \bar{S}_{t+1|t}(Z_\tau,\tau;\theta) + h(\tau)\nabla_z \log P(Y_{t+1}|Z_\tau), \tag{20}$$

where $\bar{S}_{t+1|t}(Z_\tau,\tau;\theta)$ is from the prediction step, $\nabla_z \log P(Y_{t+1}|Z_0) = \nabla_x \log P(Y_{t+1}|X_{t+1})$ (if $Z_0 = X_{t+1}$) is analytically defined in Eq. (5), and $h(\tau)$ is a damping function satisfying

$$h(\tau) \text{ is monotonically decreasing in } [0,1] \text{ with } h(0) = 1 \text{ and } h(1) = 0. \tag{21}$$

We use $h(\tau) = 1 - \tau$ for $\tau \in [0,1]$ in the numerical examples in Section 4. We remark that there are multiple choices of the damping function $h(\tau)$ that satisfying Eq. (21). How to define the optimal $h(\tau)$ is still an open question that will be considered in our future work.

We observe that the definition of $\bar{S}_{t+1|t+1}(Z_\tau,\tau;\theta)$ is compatible with the constraints (**C1**), (**C2**). Intuitively, the information of new observation data in the likelihood function is gradually injected into the diffusion model (or the score function) at the early dynamics (i.e., τ is small) of the forward SDE during which the deterministic drift (determined by $b(\tau)$) dominates the dynamics.

When the pseudo-time τ approaches 1, the diffusion term (determined by $\sigma(\tau)$) becomes dominating, the information in the likelihood is already absorbed into the diffusion model so that $h(1) = 0$ can ensure the final state Z_1 still follows the standard Gaussian distribution. The updated score function $\bar{S}_{t+1|t+1}(Z_\tau, \tau; \theta)$ can be used as the starting point of the prediction step for the time instant $t+1$.

3.2.4. Summary of the score-based filter method

In Algorithm 1, we use a brief pseudo-algorithm to summarize the score-based filter.

Algorithm 1 The score-based filter.

Input: the state equation $f(X_t, \omega_t)$, the prior density $P(X_0)$;
for $t = 0, \ldots,$
  if $t = 0$
    Generate J samples $\{x_{0,j}\}_{j=1}^J$ from $P(X_0)$;
    Train the score function $S_{0|0}$ using the sample set $\{x_{0,j}\}_{j=1}^J$;
  else
    Generate J samples $\{x_{t,j}\}_{j=1}^J$ of $P(X_t|\mathcal{Y}_t)$ using the score function $\bar{S}_{t|t}$;
  Run the state equation in Eq. (1) to obtain a predicted samples $\{\tilde{x}_{t+1,j}\}_{j=1}^J$;
  Train the score function $\bar{S}_{t+1|t}$ using the sample set $\{\tilde{x}_{t+1,j}\}_{j=1}^J$;
  Update the score function $\bar{S}_{t+1|t}$ to $\bar{S}_{t+1|t+1}$ using Eq. (20);
end

As a novel methodology for solving the nonlinear filtering problem, the score-based filter has the following advantages to provide superior accuracy:

- At any recursive stage of the data assimilation procedure, we can substitute the current score function into the reverse-time SDE in Eq. (13) to generate *unlimited* samples from the filtering density and compute any statistics of the current state.
- The score function modeled by deep neural network (DNN) can take advantage of deep learning, and the DNN learned score is potentially capable to store *complex* information contained in data and state dynamics.
- The score-based filter is equipped with an *analytical* update step, i.e., Eq. (20) to gradually inject the data information into the score model. This allows the data information to be sufficiently incorporated into the filtering density.

4. Numerical experiments

We demonstrate the performance of our score-based filter by solving three benchmark nonlinear filtering problems. In the first example, we consider a double-well potential problem, which is a 1-dimensional problem with highly nonlinear state dynamics. In the second example, we solve a bearing-tracking problem, in which the state dynamics is linear, but the measurements are nonlinear observational data, i.e., bearings of the state. This is a benchmark example to examine whether a filtering method is suitable for nonlinear problems. The third example that we shall solve in this section is the Lorenz tracking problem, and it is a well-known challenging problem for all the existing filtering methods when the dimension of the problem becomes high.[1]

4.1. Example 1: the double-well potential problem

We demonstrate the accuracy of the proposed score-based filter by solving the tracking problem of dynamical systems driven by the double-well potential. The state dynamics formulated by the double-well potential is given by the following SDE model

$$dX_t = -4X_t(X_t^2 - 1)dt + \beta d\omega_t, \tag{22}$$

where X_t is the target state, and ω_t is a standard Brownian motion with diffusion coefficient β. The drift coefficient in Eq. (22) is the derivative of a double-well energy landscape, i.e. $U(x) = (x^4 - 2x^2)$ shown in Fig. 2. The state X_t that follows the dynamics (22) describes a target particle moving on the energy landscape $U(x)$. So there are two stable energy states, i.e. $X_t = 1$ and $X_t = -1$, and there's a forcing term defined by the derivative of the energy potential, that "drags" the state towards one of the stable states.

In the numerical tests, we use the discretized double-well potential model with temporal step-size $\Delta t = 0.1$ defined by

$$X_{n+1} = X_n - 4 \cdot 0.1 \cdot X_n(X_n^2 - 1) + \beta\sqrt{0.1} \cdot \omega_n. \tag{23}$$

In order to track the target particle governed by Eq. (23), we assume that we have direct observations on X_{n+1}, i.e.,

$$Y_{n+1} = X_{n+1} + \varepsilon_{n+1}, \tag{24}$$

[1] **Reproducibility**: the score-based filter is implemented in Python. The source code is publicly available at https://github.com/zezhongzhang/Score-based-Filter. All the numerical results presented in this section can be exactly reproduced using the code on Github.

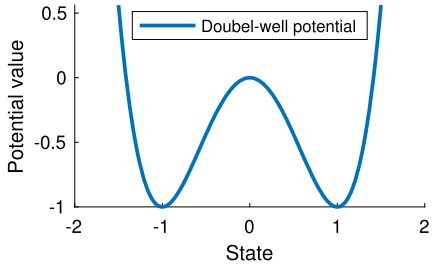


Fig. 2. The double-well potential that drives the dynamical system in Eq. (22). It's relatively easy to track the state X_t in Eq. (22) when it stays in one of the potential wells. However, when the state S_t suddenly switches from one potential well to another, the large discrepancy between the state prediction and the measurement data may lead to tracking lost. Thus, we use this problem to demonstrate the superior accuracy of the proposed score-based filter in capturing the state.

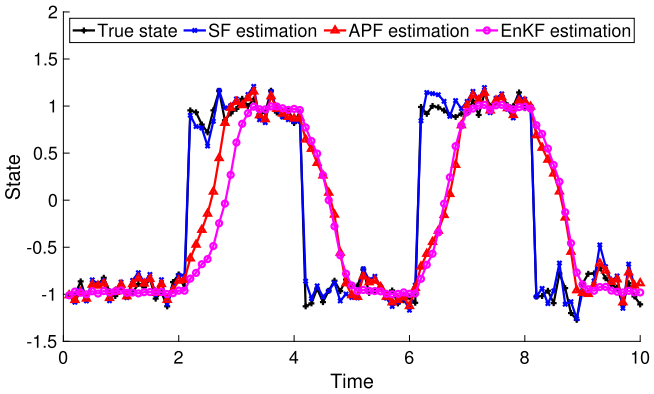


Fig. 3. Accuracy comparison for tracking the dynamical system in Eq. (22) with large noise, i.e., $\beta = 0.3$, where the accuracy is measured by the discrepancy between the true and the estimated state trajectories. All the three methods can track the true state (the black curve marked by pluses) while the state stays in the bottom of a potential well. When the true state switches from one potential well to the other, it takes several time steps for both APF (the red curve marked by triangles) and EnKF (the magenta curve marked by dots) to adjust and capture the unexpected state switch. In comparison, SF (the red curve marked by triangles) can quickly capture the unexpected state switch, which demonstrates its superior accuracy.

where $\varepsilon_n \sim N(0, R)$ is the observational noise with standard deviation $R = 0.1$. It's easy to track the state while it stays in the bottom of one of the potential wells. The challenge in solving the nonlinear filtering problem (23) - (24) is that when the state (unexpectedly) switches from one potential well to another, there's a big discrepancy between the predicted state and the measurement data. In this case, the stability of the optimal filtering algorithm becomes the main issue.

We carry out several experiments to compare the performance of our score-based filter with two state-of-the-art optimal filtering methods, i.e., the particle filter and the ensemble Kalman filter. To implement the score-based filter, we use the sliced score-matching method (see [40,41]) to train the score model defined by a fully-connected neural network with 2 hidden layers and 50 neurons per hidden layer. The sampling procedure through the reverse-time SDE is implemented by the Euler-Maruyama scheme with $K = 600$ time steps. Recall that the number of samples generated through the reverse-time SDE can be arbitrarily chosen and the computational cost for generating those samples is small. The particle filter used here is the auxiliary particle filter (APF) that is a popular particle filter method with a moderate-cost resampling procedure to improve the performance of the standard bootstrap particle filter. We use 1000 particles to produce an empirical approximation for the filtering density. The ensemble Kalman filter (EnKF) is implemented using 100 Kalman filter samples in the ensemble. We track the target state over time interval $[0, 10]$, i.e., 100 time steps, for two noise levels $\beta = 0.3$ or 0.2 in Eq. (22).

The results are shown in Fig. 3 for $\beta = 0.3$ and Fig. 4 for $\beta = 0.2$, respectively. We can see that all the three methods can track the true state while the state stays in the bottom of a potential well. However, when there's a state switch SF can quickly adjust the changes, and it takes a few time steps for APF and EnKF to capture the switch. The lagged tracking of APF and EnKF results from the fact that the state model produces strong force that drags the target towards one of the stable energy points, i.e., the bottoms. In this way, the particles/samples of APF (or EnKF) will be moved towards the bottom of a potential well. This not only makes the particles/samples concentrate at one of the bottoms but also significantly reduces the variance of the predicted filtering density, i.e., the prior density. Since APF and EnKF utilize finite particles/samples to characterize empirical approximations for the predicted filtering density, narrower distribution bands would lead to inaccurate tail approximations. When the likelihood corresponding to the measurement data lies on the tail of the prior distribution, the data cannot effectively influence the posterior distribution – due to the poor tail approximation by limited particles/samples (or no samples at all) at the distribution's tail. Several advanced techniques have been developed to address tail approximation issues in particle filters or Kalman filters [3,9,16,28,31,32]. However, these techniques often come with increased computational demands, which makes them less suitable for higher dimensional problems that closely related to real-world scenarios.

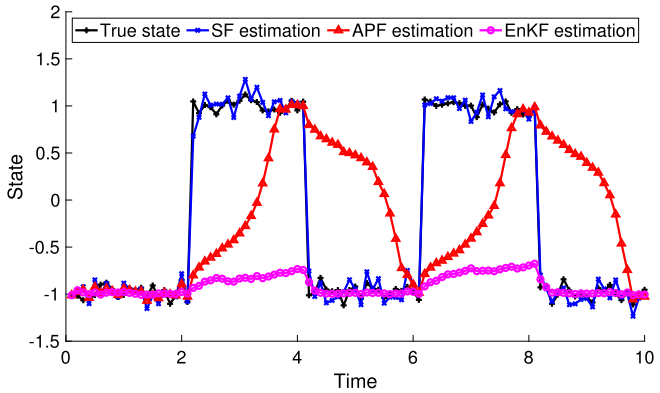


Fig. 4. Accuracy comparison for tracking the dynamical system in Eq. (22) with smaller noise, i.e., $\beta = 0.2$, where the accuracy is measured by the discrepancy between the true and the estimated state trajectories. The proposed SF shows more advantageous performance over APF and EnKF in this setting. The small variance of the noise makes the variance of the predicted filtering density obtained by APF and EnKF much narrower, such that APF and EnKF produces stronger confidence of the predicted state and therefore takes more times steps to capture the unexpected state switch. In comparison, SF uses the analytical form of the likelihood via Eq. (20) to update the filtering density, such that the measurement data information is guaranteed to be incorporated.

To further demonstrate the accuracy of SF, we reduce the noise variance in the state dynamics to $\beta = 0.2$. We can see that SF has even more advantageous performance compared with APF and EnKF. This is because a smaller noise level makes the variance of the predicted filtering density obtained by APF and EnKF narrower. This results in stronger confidence in the predicted state and therefore make the update even less effective. In comparison, SF can produce as many samples as needed for the target distribution, which leads to more accurate tail approximations. Moreover, in the SF method, the likelihood is continuously added to the posterior score corresponding to the updated filtering density. Therefore, the measurement data information is guaranteed to be incorporated into the updated filtering density through the reverse-time SDE sampler.

4.2. Example 2: bearing-only tracking

We demonstrate the performance of SF in handling highly nonlinear observational functions using the benchmark bearing-only tracking problem. Specifically, we consider the following linear dynamical system that models a moving target on the two-dimensional plane:

$$X_{n+1} = X_n + A\Delta t + B\sqrt{\Delta t}\cdot\omega_n, \tag{25}$$

where $X_n = [u_n, v_n]^\top$ is the position of the target, ω_n is a standard Brownian motion, $A = [\nu_1, 0; 0, \nu_2]$ is the velocity matrix that tells how fast the target moves, B is the diffusion coefficient, and Δt is the time step. In this example, we let $\nu_1 = 4$, $\nu_2 = 6$, $B = [0.2, 0; 0, 0.2]$, and we choose $\Delta t = 0.05$. In the bearing-only tracking problem, we can only receive bearings (i.e., the angles) for the target state, and the observational process is defined by

$$Y_{n+1} = \arctan\frac{v_n - \bar{v}}{u_n - \bar{u}} + \epsilon_{n+1},$$

where $\epsilon_{n+1} \sim N(0, R)$ is the observational noise, and $[\bar{u}, \bar{v}]^\top$ is an observation platform to locate the detector. Similar to Example 1, we compare SF with APF and EnKF, where all the three methods use the same setup as in Example 1.

Fig. 5 shows the tracking performance of SF, APF, and EnKF, where the APF is implemented with 1000 particles and the EnKF is implemented using 100 Kalman filter samples in the ensemble. The green diamond is the location of the detector platform. The initial position of the target state is chosen as $X_0 = [1, 1]^\top$. We observe that both SF and APF provide accurate estimates for the target state, but EnKF does not work well due to the nonlinear observations [19]. Additionally, Fig. 6 shows the tracking errors of SF, APF, and EnKF. In this example, we let $R = [0.01, 0; 0, 0.01]$, and the observation platform is chosen at $[\bar{u}, \bar{v}]^\top = [-5, 10]^\top$, and we track the target for 20 steps. We see that although the state model is simply a linear dynamical system, the highly nonlinear observational function arctan makes the Kalman-type filters unreliable. This verifies that both the particle filter approach and the score-based filter are suitable for solving the nonlinear filtering problem.

To demonstrate the capability of the score-based filter in approximating the filtering densities, we plot the posterior samples generated by the posterior filtering score in the SF (at time instants $t = 4$, $t = 8$, $t = 12$ and $t = 12$) as well as posterior particles in the APF in Fig. 7. At each time instant, 300 samples/particles are presented. We can see that the posterior distributions described by SF samples are similar to the empirical distributions of the particles in the APF. Since for this relatively low dimensional problem, the APF is known to be asymptotically correct in approximating posterior filtering densities, Fig. 7 indicates that the SF can provide reasonably good descriptions for filtering densities.

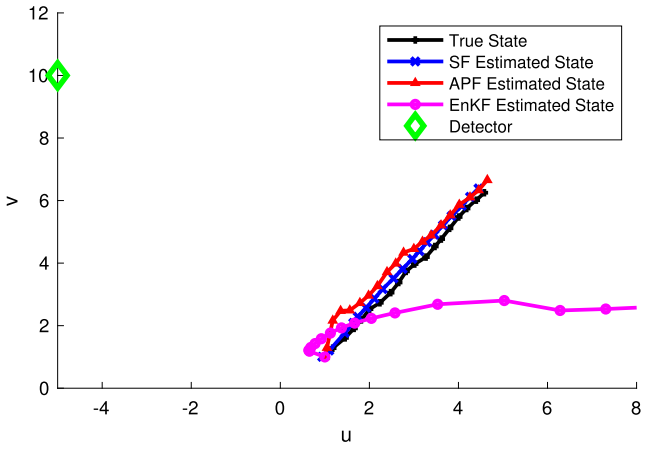


Fig. 5. Comparison for tracking the dynamical system in Eq. (25), where the accuracy is measured by the discrepancy between the true and the estimated state trajectories. We observe that SF and APF successfully capture the target state using the highly nonlinear observation, but EnKF does not work well due to limited capabilities in handling the nonlinear observations.

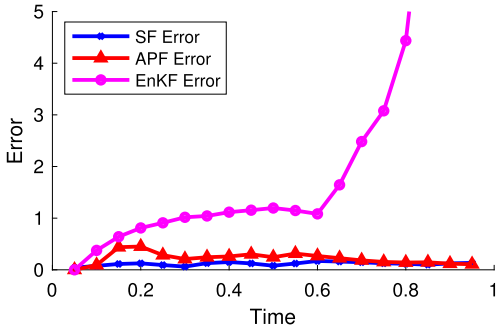


Fig. 6. The l^2 error between the estimated state and the true state for the results in Fig. 5. It verifies that SF and APF have comparable accuracy in handling low-dimensional and nonlinear problems, while EnKF does not perform well.

4.3. Example 3: high-dimensional Lorenz attractor

We demonstrate the SF's capability in handling high-dimensional (up 100 dimension) Lorenz attractor problems. Specifically, we track the state of the Lorenz 96 model described as follows:

$$\frac{dx_i}{dt} = (x_{i+1} - x_{i-2})x_{i-1} + F, \qquad i = 1, 2, \cdots, d, \quad d \geq 4, \tag{26}$$

where $X_t = [x_1(t), x_2(t), \cdots, x_d(t)]^\top$ is a d-dimensional target state, and it is assumed that $x_{-1} = x_{d-1}$, $x_0 = x_d$, and $x_{d+1} = x_1$. The term F is a forcing constant. When $F = 8$, the Lorenz 96 dynamics (26) becomes a chaotic system, which makes tracking the state S_t a challenging task for all the existing filtering techniques, especially in dimensional spaces. We discretize Eq. (26) through the Euler scheme with temporal step size $\Delta t = 0.01$. To test SF's performance in a realistic scenario, we make the following two changes to the original Lorenz attractor model, i.e.,

- *Nonlinear observation:* we assume that the observational process in Eq. (1) is a cubic function of the state, i.e.,

$$Y_{n+1} = (X_{n+1})^3 + \varepsilon_{n+1}, \tag{27}$$

  which is commonly seen in real-world applications.
- *Random state perturbation:* we add a d-dimensional white noise (with the standard deviation 0.1) to perturb the Lorenz 96 model, which intertwines with the chaotic property making the filtering problem more challenging.

4.3.1. Test on the 10-dimensional Lorenz system

We track the target state for 100 time steps in the 10-dimensional space, i.e. $d = 10$. We use a 2-layer fully connected neural network with 200 neurons per layer to define the score model in Eq. (20), and the sampling procedure through the reverse-time SDE is implemented with $K = 800$ time steps. For APF, we use $20,000$ particles to construct the empirical distribution for the filtering density. For EnKF, we use $1,000$ Kalman filter samples, which is already a very large number of samples for the EnKF for

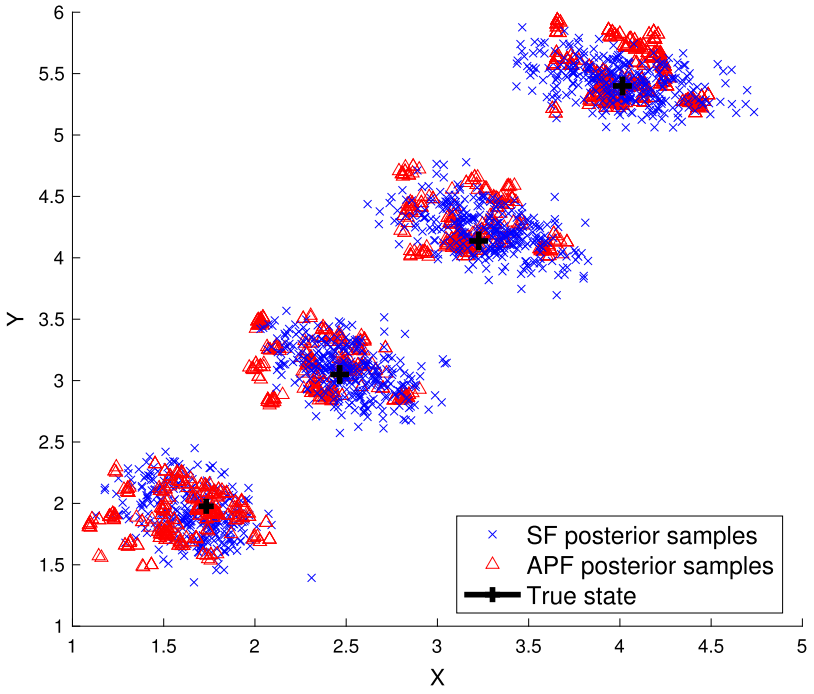


Fig. 7. Comparison of posterior samples between the SF and the APF with 300 samples at time instants $t = 4$, $t = 8$, $t = 12$ and $t = 12$. The blue crosses provide posterior samples of the SF. The red triangles are the posterior samples of the APF. The true state is given by black pluses. The posterior distributions described by SF samples are similar to the empirical distributions of the particles in the APF.

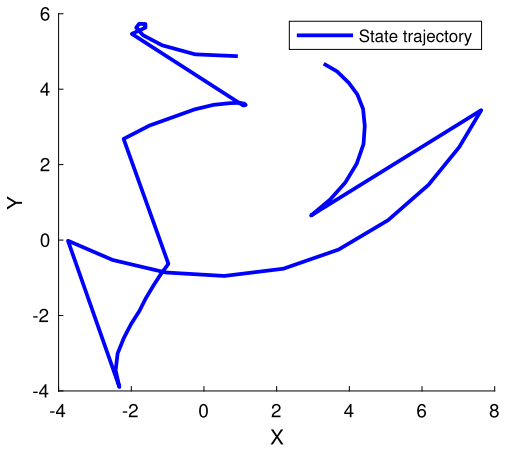
a 10-dimensional problem. To make the tracking task even more difficult, we add two random shocks to mimic the severe chaotic behavior of the Lorenz system at time instants $n = 21$ and $n = 41$, and those unexpected shocks could challenge the stability of different nonlinear filtering methods. In practical applications, those unexpected random shocks can be interpreted as incomplete knowledge about the state dynamics. This kind of incomplete knowledge would typically cause large errors between predictions and true states.

Fig. 8 illustrates the high discrepancy, caused by the nonlinearity, between one target state's trajectory and the corresponding observation's trajectory. We plot the first two directions of the original signal X_t in subplot (a), and we compare the observations with the real signal in subplot (b), where the observation trajectory is the red curve (marked by triangles) and the true signal is the blue curve. We can see that the cube measurements are highly nonlinear and only limited data information is contained in the cube observations. Therefore, being able to handling nonlinear observation processes is essential to solve the nonlinear filtering problem in Eq. (26)-(27).

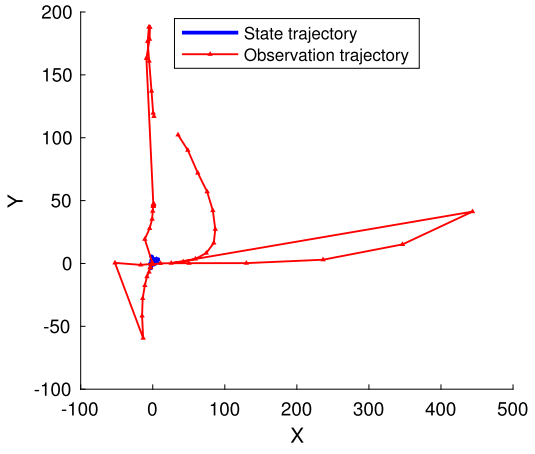
Fig. 9 presents the state estimation comparison between SF, APF, and EnKF. In subplots (a), (b), and (c) of Fig. 9, we compare the accuracy of state estimation between SF and APF in estimating x_1, x_5, and x_9, respectively, where the black curves marked pluses describe the true states, the red curves marked by triangles are the APF estimated states, and the blue curves marked by crosses are the SF estimated states. We can see from those subplots that the APF works well at beginning. However, when the random shock occurs, the large discrepancy between the state prediction and the measurement data makes the particle filter method degenerate. This is indicated by the flat and unresponsive estimation curve. On the other hand, although the SF also suffers from the unexpected random shocks, it still captures the true state with acceptable accuracy. Given the high difficulty of the problem, SF's performance is promising and significantly better than APF and EnKF.

4.3.2. Test on the 100-dimensional Lorenz system

We further challenge our SF method by solving a 100-dimensional Lorenz attractor problem. Note that as a nonlinear filtering method, we need to characterize a non-Gaussian distribution in the 100-dimensional space, which would certainly encounter the difficulty of "curse of dimensionality". To implement the SF, we increase the size of the neural network for the score model to 400 neurons per hidden layer, and the sampling procedure through the reverse-time SDE is implemented with 1000 time steps. Since the SF outperformed the APF and the EnKF in the 10-dimensional Lorenz problem, we compare SF with an Online Variational Filtering method (Online-Var) [14], which is an advanced variational approach for optimal filtering problems known for its efficacy in addressing high-dimensional nonlinear problems. To challenge both SF and the Online-Var method, we add an unpredictable random shock at time instant $n = 26$. The addition of random shocks mimics the scenario of having unexpected extreme events in real-world data assimilation problems, e.g., weather forecasting. In Fig. 10, we compare the performance of SF with Online-Var in six selected dimensions. We can see from this figure that both SF and Online-Var can capture the main features of the state dynamics, and SF outperformed Online-Var in accuracy.

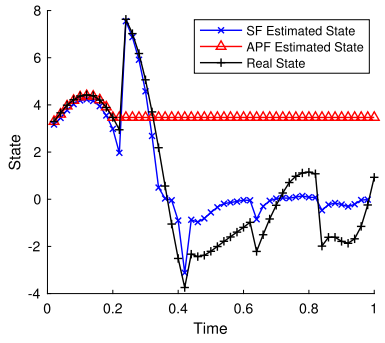


(a) One state trajectory of the true signal

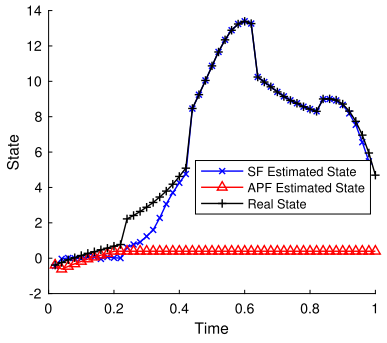


(b) Comparison of the true state and the observation trajectories

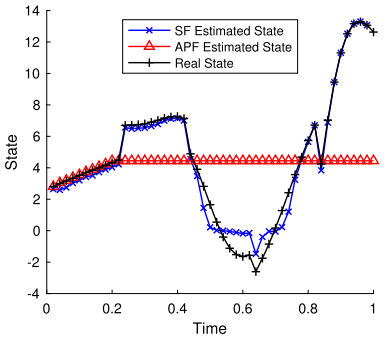
Fig. 8. Illustration of the high discrepancy, caused by the nonlinearity, between one target state's trajectory and the corresponding observation's trajectory for the 10-dimensional Lorenz model in Eq. (26)-(27), which presents a significant challenge for nonlinear filtering methods.



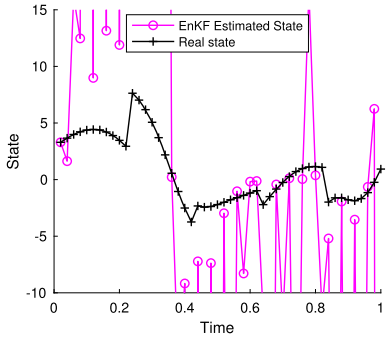
(a) SF & APF: x_1



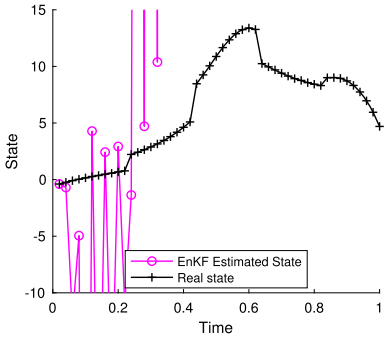
(b) SF & APF: x_5



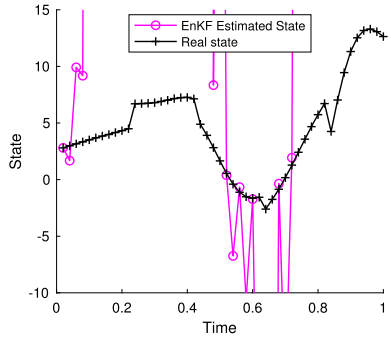
(c) SF & APF: x_9



(d) EnKF: x_1



(e) EnKF: x_5



(f) EnKF: x_9

Fig. 9. Accuracy comparison for tracking the 10-dimensional Lorenz model, where the accuracy is measured by the discrepancy between the true and the estimated state trajectories. We illustrate three state components in Eq. (26), i.e., x_1, x_5, and x_9. It is easy to see that SF provides a significantly better accuracy than APF and EnKF, where APF degenerates after the random shock occurs and EnKF cannot handle the high nonlinearity illustrated in Fig. 8.

To validate the superior performance of SF over Online-Var, we conduct the above experiment 20 times and plot the root mean square errors (RMSEs) across all 100 dimensions over the 20 repeated trials. The comparison of RMSEs is depicted in Fig. 11, with shaded areas representing 95% (2σ) error regions for both methods. We can see that both SF and Online-Var method can recover from the random shock, which generated unexpected large errors. However, throughout the entire tracking period, SF consistently outperformed Online-Var.

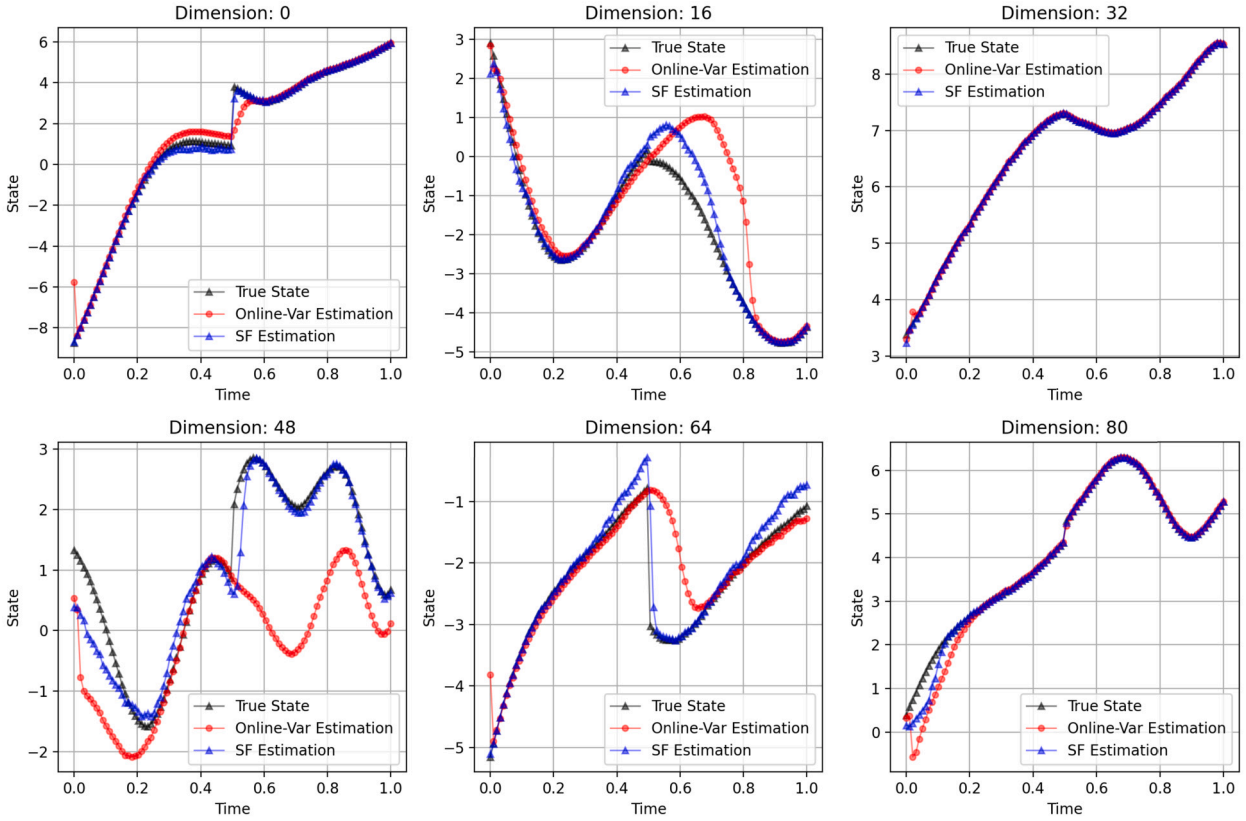


Fig. 10. Comparison between the SF and the Online-Var method in estimating 100-dimensional Lorenz 96 model. The SF outperforms the Online-Var in accuracy in this 100-dimensional test.

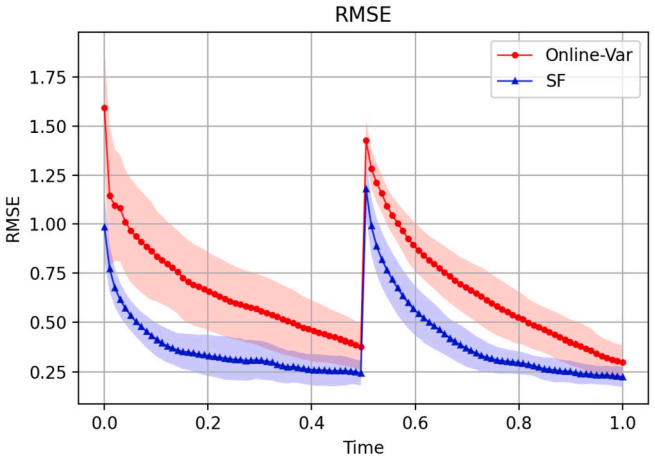


Fig. 11. Comparison of average RMSEs over 100 dimensions between SF and Online-Var in solving the 100-dimensional Lorenz problem with 20 repeated trials.

5. Concluding remarks

We propose the SF method to solve high-dimensional nonlinear filtering problems. The outstanding performance of SF, especially in solving the 100-dimensional Lorenz system, shows a great potential to handling much higher dimensional problems, which motivate our future research from the following perspectives. First, the efficiency of the training the neural network to approximate the score function is not ideal, which limits the applicability of the current version of SF in nonlinear filtering problems that require fast feedback. We will exploit data and model parallelism strategies on modern computing platforms or even super computers to improve the training accuracy. Second, the efficiency of reverse-time sampling can also be improved by incorporating advanced stable time stepping schemes, e.g., the exponential integrator, to significantly reduce the number of time steps in the discretization

of the reverse-time process in the diffusion model. Third, the choice of damping function h in Eq. (21) in the current format is not optimal. We will explore better ways to define the damping function h in future studies.

CRediT authorship contribution statement

Feng Bao: Conceptualization, Funding acquisition, Methodology, Writing – original draft, Writing – review & editing. **Zezhong Zhang:** Conceptualization, Methodology, Software, Writing – original draft, Writing – review & editing. **Guannan Zhang:** Funding acquisition, Investigation, Methodology, Writing – original draft, Writing – review & editing, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

We have shared the source code and data on Github. The Github link is included in the manuscript.

Acknowledgement

This work is supported by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research, Applied Mathematics program, under the contract ERKJ387, and accomplished at Oak Ridge National Laboratory (ORNL), and under Grant DE-SC0022254. ORNL is operated by UT-Battelle, LLC., for the U.S. Department of Energy under Contract DE-AC05-00OR22725. The first author (FB) would also like to acknowledge the support from U.S. National Science Foundation through project DMS-2142672 and the support from the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research, Applied Mathematics program under Grant DE-SC0022297.

Appendix A. Additional information on the implementation of diffusion models

A.1. The choice of the coefficients for the forward SDE

The task of the forward SDE in Eq. (11) is to transform any given initial distribution $Q_0(Z_0)$ to the standard Gaussian distribution $\mathcal{N}(0, \mathbf{I}_d)$. It is shown in [41,42,21] that such task can be done by a linear SDE with properly chosen drift and diffusion coefficients. For example, we can define $b(\tau)$ and $\sigma(\tau)$ in Eq. (11) by

$$b(\tau) = \frac{\mathrm{d}\log\alpha_\tau}{\mathrm{d}\tau} \quad \text{and} \quad \sigma^2(\tau) = \frac{\mathrm{d}\beta_\tau^2}{\mathrm{d}\tau} - 2\frac{\mathrm{d}\log\alpha_\tau}{\mathrm{d}\tau}\beta_\tau^2, \tag{A.1}$$

where the two processes α_τ and β_τ are defined by

$$\alpha_\tau = 1 - \tau, \ \ \beta_\tau^2 = \tau \quad \text{for} \quad \tau \in [0, 1]. \tag{A.2}$$

The definitions in Eq. (A.1) and Eq. (A.2) can ensure that the conditional density function $Q_\tau(Z_\tau|Z_0)$ for any fixed Z_0 is the following Gaussian distribution:

$$Q_\tau(Z_\tau|Z_0) = \mathcal{N}(\alpha_\tau Z_0, \beta_\tau^2 \mathbf{I}_d). \tag{A.3}$$

It is easy to see that the choice of α_τ and β_τ can ensure that

$$Q_1(Z_1|Z_0) = \mathcal{N}(0, \mathbf{I}_d) \implies Q_1(Z_1) = \int_{\mathbb{R}^d} Q_1(Z_1|Z_0)Q_0(Z_0)dZ_0 = \mathcal{N}(0, \mathbf{I}_d),$$

which is the property we need for the forward SDE.

A.2. Discretization of the forward and reverse-time SDEs

Taking the reverse-time SDE in Eq. (13) as an example, we use the Euler-Maruyama scheme to discretize the reverse-time SDE and transform any set of Gaussian samples $\{z_{1,j}\}_{j=1}^J$ of the final state Z_1 to a set of samples, denoted by $\{z_{0,j}\}_{j=1}^J$, approximately following the target distribution $Q_0(Z_0)$. Specifically, we first introduce a partition of the pseudo-temporal domain $\mathcal{T} = [0, 1]$, i.e.,

$$\mathcal{D}_K := \{\tau_k \mid 0 = \tau_0 < \tau_1 < \cdots < \tau_k < \tau_{k+1} < \cdots < \tau_K = 1\}$$

with uniform step-size $\Delta\tau = \frac{1}{K}$. For each sample $z_{1,j}$, we obtain the approximate solution $z_{0,j}$ by recursively evaluating the following scheme

$$z_{\tau_k,j} = z_{\tau_{k+1},j} - \left[b(\tau_{k+1})z_{\tau_{k+1},j} - \sigma^2(\tau_{k+1})S(z_{\tau_{k+1},j}, \tau_{k+1})\right]\Delta\tau + \sigma(\tau_{k+1})\Delta W_{\tau_{k+1},j}, \tag{A.4}$$

for $k = K-1, K-2, \cdots, 1, 0$, where $\Delta W_{\tau_{k+1},j}$ is a realization of the Brownian increment. The accuracy of $\{z_{0,j}\}_{j=1}^{J}$ is determined by the number of pseudo-time steps K.

A.3. The loss function for training the diffusion model

We provide details of the general loss function in Eq. (15) for training the approximate score function. The full definition of the loss function in Eq. (15) is

$$Loss = \mathbb{E}_{\tau\sim\mathcal{U}[0,1],\, Z_0\sim Q_0(Z_0),\, Z_\tau\sim Q_\tau(Z_\tau|Z_0)}\left[\lambda(\tau)\beta_\tau^2\|\nabla_z \log Q_\tau(Z_\tau) - \bar{S}(Z_\tau,\tau;\theta)\|_2^2\right], \tag{A.5}$$

where $\mathcal{U}[0,1]$ is the uniform distribution, $Q_0(Z_0)$ is the target distribution, $Q_\tau(Z_\tau|Z_0)$ is the conditional distribution given in Eq. (A.3), β_τ is defined in Eq. (A.2), and $\lambda(\tau)$ is a weighting function. This formulation is not practical because we do not know the exact score function $\nabla_z \log Q_\tau(Z_\tau)$. Thanks to the derivation in [42], the loss in Eq. (A.5) is equivalent to

$$Loss = \mathbb{E}_{\tau\sim\mathcal{U}[0,1],\, Z_0\sim Q_0(Z_0),\, Z_\tau\sim Q_\tau(Z_\tau|Z_0)}\left[\lambda(\tau)\beta_\tau^2\|\nabla_z \log Q_\tau(Z_\tau|Z_0) - \bar{S}(Z_\tau,\tau;\theta)\|_2^2\right] + const, \tag{A.6}$$

where the exact score function is replaced by the gradient of the logarithm of the conditional distribution $Q_\tau(Z_\tau|Z_0)$. It makes the task much easier because we know $Q_\tau(Z_\tau|Z_0)$ is the Gaussian distribution $\mathcal{N}(\alpha_\tau Z_0, \beta_\tau^2\mathbf{I}_d)$. Thus, we have

$$\nabla_z \log Q_\tau(Z_\tau|Z_0) = -\frac{Z_\tau - \alpha_\tau Z_0}{\beta_\tau^2},$$

such that the loss function in Eq. (A.6) is computable, i.e.,

$$Loss = \mathbb{E}_{\tau\sim\mathcal{U}[0,1],\, Z_0\sim Q_0(Z_0),\, Z_\tau\sim Q_\tau(Z_\tau|Z_0)}\left[\lambda(\tau)\left\|-\frac{Z_\tau - \alpha_\tau Z_0}{\beta_\tau} - \beta_\tau\bar{S}(Z_\tau,\tau;\theta)\right\|_2^2\right] + const.$$

Moreover, because $\frac{Z_\tau-\alpha_\tau Z_0}{\beta_\tau}$ follows the standard Gaussian distribution, the final loss used to solve the optimization problem in Eq. (15) is

$$Loss = \mathbb{E}_{\tau\sim\mathcal{U}[0,1],\, Z_0\sim Q_0(Z_0),\, \zeta\sim\mathcal{N}(0,\mathbf{I}_d)}\left[\lambda(\tau)\left\|\zeta - \beta_\tau\bar{S}(Z_\tau,\tau;\theta)\right\|_2^2\right].$$

References

[1] A. Aksoy, D. Dowell, C. Snyder, A multicase comparative assessment of the ensemble Kalman filter for assimilation of radar observations. Part I: storm-scale analyses 137 (2009) 1805–1824.
[2] T. Amit, T. Shaharbany, E. Nachmani, L. Wolf, Segdiff: Image segmentation with diffusion probabilistic models, 2022.
[3] J.L. Anderson, A local least squares framework for ensemble filtering 131 (2003) 634–642.
[4] C. Andrieu, A. Doucet, R. Holenstein, Particle Markov chain Monte Carlo methods, J. R. Stat. Soc. B 72 (2010) 269–342.
[5] J. Austin, D.D. Johnson, J. Ho, D. Tarlow, R. van den Berg, Structured denoising diffusion models in discrete state-spaces, in: Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, 2021, pp. 17981–17993, virtual.
[6] F. Bao, Y. Cao, P. Maksymovych, Backward sde filter for jump diffusion processes and its applications in material sciences, Commun. Comput. Phys. 27 (2020) 589–618.
[7] F. Bao, Y. Cao, C. Webster, G. Zhang, A hybrid sparse-grid approach for nonlinear filtering problems based on adaptive-domain of the Zakai equation approximations, SIAM/ASA J. Uncertain. Quantificat. 2 (2014) 784–804.
[8] F. Bao, N. Cogan, A. Dobreva, R. Paus, Data assimilation of synthetic data as a novel strategy for predicting disease progression in alopecia areata, Math. Med. Biol. (2021).
[9] F. Bao, V. Maroulas, Adaptive meshfree backward SDE filter, SIAM J. Sci. Comput. 39 (2017) A2664–A2683.
[10] D. Baranchuk, A. Voynov, I. Rubachev, V. Khrulkov, A. Babenko, Label-efficient semantic segmentation with diffusion models, in: International Conference on Learning Representations, 2022.
[11] E.A. Brempong, S. Kornblith, T. Chen, N. Parmar, M. Minderer, M. Norouzi, Denoising pretraining for semantic segmentation, in: IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, CVPR Workshops 2022, New Orleans, LA, USA, June 19-20, 2022, IEEE, 2022, pp. 4174–4185.
[12] M.F. Bugallo, T. Lu, P.M. Djuric, Target tracking by multiple particle filtering, in: 2007 IEEE Aerospace Conference, 2007, pp. 1–7.
[13] R. Cai, G. Yang, H. Averbuch-Elor, Z. Hao, S.J. Belongie, N. Snavely, B. Hariharan, Learning gradient fields for shape generation, in: Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part III, in: Lecture Notes in Computer Science, vol. 12348, Springer, August 2020, pp. 364–381.
[14] A. Campbell, Y. Shi, T. Rainforth, A. Doucet, Online variational filtering and parameter learning, 2021.
[15] H.G. Chipilski, X. Wang, D.B. Parsons, Impact of assimilating PECAN profilers on the prediction of bore-driven nocturnal convection: a multiscale forecast evaluation for the 6 July 2015 case study 148 (2020) 1147–1175.
[16] A.J. Chorin, X. Tu, Implicit sampling for particle filters, Proc. Natl. Acad. Sci. USA 106 (2009) 17249–17254.
[17] P. Dhariwal, A. Nichol, Diffusion Models Beat Gans on Image Synthesis, Advances in Neural Information Processing Systems, vol. 34, Curran Associates, Inc., 2021, pp. 8780–8794.
[18] G. Evensen, The ensemble Kalman filter for combined state and parameter estimation: Monte Carlo techniques for data assimilation in large systems, IEEE Control Syst. Mag. 29 (2009) 83–104.
[19] N. Gordon, D. Salmond, A. Smith, Novel approach to nonlinear/non-gaussian bayesian state estimation, IEE Proc. F 140 (1993) 107–113.
[20] A. Graikos, N. Malkin, N. Jojic, D. Samaras, Diffusion models as plug-and-play priors, CoRR, arXiv:2206.09012 [abs], 2022.

[21] J. Ho, A. Jain, P. Abbeel, Denoising diffusion probabilistic models, in: Advances in Neural Information Processing Systems, vol. 33, Curran Associates, Inc., 2020, pp. 6840–6851.
[22] J. Ho, C. Saharia, W. Chan, D.J. Fleet, M. Norouzi, T. Salimans, Cascaded diffusion models for high fidelity image generation, J. Mach. Learn. Res. 23 (2022) 47:1–47:33.
[23] E. Hoogeboom, D. Nielsen, P. Jaini, P. Forré, M. Welling, Argmax flows and multinomial diffusion: learning categorical distributions, in: Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems, 2021, NeurIPS 2021, December 6-14, 2021, 2021, pp. 12454–12465, virtual.
[24] K. Kang, V. Maroulas, I. Schizas, F. Bao, Improved distributed particle filters for tracking in a wireless sensor network, Comput. Stat. Data Anal. 117 (2018) 90–108.
[25] B. Kawar, G. Vaksman, M. Elad, Stochastic image denoising by sampling from the posterior distribution, in: IEEE/CVF International Conference on Computer Vision Workshops, ICCVW 2021, Montreal, BC, Canada, October 11-17, 2021, IEEE, 2021, pp. 1866–1875.
[26] B. Kim, I. Han, J.C. Ye, Diffusemorph: unsupervised deformable image registration along continuous trajectory using diffusion models, CoRR, arXiv:2112.05149 [abs], 2021.
[27] H. Li, Y. Yang, M. Chang, S. Chen, H. Feng, Z. Xu, Q. Li, Y. Chen, Srdiff: single image super-resolution with diffusion probabilistic models, Neurocomputing 479 (2022) 47–59.
[28] X. Li, F. Bao, K. Gallivan, A drift homotopy implicit particle filter method for nonlinear filtering problems, Discrete Contin. Dyn. Syst., Ser. S 15 (2022) 727–746.
[29] X.L. Li, J. Thickstun, I. Gulrajani, P. Liang, T.B. Hashimoto, Diffusion-lm improves controllable text generation, CoRR, arXiv:2205.14217 [abs], 2022.
[30] S. Luo, W. Hu, Score-based point cloud denoising, in: 2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10–17, 2021, IEEE, 2021, pp. 4563–4572.
[31] V. Maroulas, X. Pan, J. Xiong, Large deviations for the optimal filter of nonlinear dynamical systems driven by Lévy noise, Stoch. Process. Appl. 130 (2020) 203–231.
[32] V. Maroulas, P. Stinis, Improved particle filters for multi-target tracking, J. Comput. Phys. 231 (2012) 602–611.
[33] C. Meng, Y. He, Y. Song, J. Song, J. Wu, J. Zhu, S. Ermon, SDEdit: guided image synthesis and editing with stochastic differential equations, in: The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022, OpenReview.net, 2022.
[34] M.K. Pitt, N. Shephard, Filtering via simulation: auxiliary particle filters, J. Am. Stat. Assoc. 94 (1999) 590–599.
[35] B. Ramaprasad, Stochastic filtering with applications in finance, 2010.
[36] C. Saharia, J. Ho, W. Chan, T. Salimans, D.J. Fleet, M. Norouzi, Image super-resolution via iterative refinement, IEEE Trans. Pattern Anal. Mach. Intell. 45 (2023) 4713–4726.
[37] N. Savinov, J. Chung, M. Binkowski, E. Elsen, A. van den Oord, Step-unrolled denoising autoencoders for text generation, in: The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022, OpenReview.net, 2022.
[38] C. Snyder, T. Bengtsson, P. Bickel, J. Anderson, Obstacles to high-dimensional particle filtering, Mon. Weather Rev. 136 (2008) 4629–4640.
[39] J. Sohl-Dickstein, E.A. Weiss, N. Maheswaranathan, S. Ganguli, Deep unsupervised learning using nonequilibrium thermodynamics, J. Mach. Learn. Res. Workshop Conf. Proc. 37 (2015) 2256–2265, JMLR.org.
[40] Y. Song, S. Ermon, Generative modeling by estimating gradients of the data distribution, Adv. Neural Inf. Process. Syst. 32 (2019).
[41] Y. Song, J. Sohl-Dickstein, D.P. Kingma, A. Kumar, S. Ermon, B. Poole, Score-based generative modeling through stochastic differential equations, in: International Conference on Learning Representations, 2021.
[42] P. Vincent, A connection between score matching and denoising autoencoders, Neural Comput. 23 (2011) 1661–1674.
[43] J. Whang, M. Delbracio, H. Talebi, C. Saharia, A.G. Dimakis, P. Milanfar, Deblurring via stochastic refinement, in: IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022, IEEE, 2022, pp. 16272–16282.
[44] P. Yu, S. Xie, X. Ma, B. Jia, B. Pang, R. Gao, Y. Zhu, S. Zhu, Y.N. Wu, Latent diffusion energy-based model for interpretable text modelling, Proc. Mach. Learn. Res., PMLR 162 (2022) 25702–25720.
[45] M. Zakai, On the optimal filtering of diffusion processes, Z. Wahrscheinlichkeitstheor. Verw. Geb. 11 (1969) 230–243.