Propagating Model Uncertainty through Filtering-based Probabilistic Numerical ODE Solvers

Dingling Yao

Filip Tronarp

Nathanael Bosch

Institute of Science and Technology Austria

Lund University

Tübingen AI Center, University of Tübingen

Abstract

Filtering-based probabilistic numerical solvers for ordinary differential equations (ODEs), also known as ODE filters, have been established as efficient methods for quantifying numerical uncertainty in the solution of ODEs. In practical applications, however, the underlying dynamical system often contains uncertain parameters, requiring the propagation of this model uncertainty to the ODE solution. In this paper, we demonstrate that ODE filters, despite their probabilistic nature, do not automatically solve this uncertainty propagation problem. To address this limitation, we present a novel approach that combines ODE filters with numerical quadrature to properly marginalize over uncertain parameters, while accounting for both parameter uncertainty and numerical solver uncertainty. Experiments across multiple dynamical systems demonstrate that the resulting uncertainty estimates closely match reference solutions. Notably, we show how the numerical uncertainty from the ODE solver can help prevent overconfidence in the propagated uncertainty estimates, especially when using larger step sizes. Our results illustrate that probabilistic numerical methods can effectively quantify both numerical and parametric uncertainty in dynamical systems.

1 Introduction

Ordinary differential equations (ODEs) are used throughout the sciences to describe the evolution of dynamical systems over time. They often appear in so-called initial value problems (IVP), given by equations

$$\dot{y}(t) = f_\theta(y(t), t), \qquad t \in [0, T], \tag{1a}$$

$$y(0) = c_\theta, \tag{1b}$$

where $f_\theta : \mathbb{R}^d \times \mathbb{R} \to \mathbb{R}^d$ is a vector field that describes the dynamics of the system and $c_\theta \in \mathbb{R}^d$ is the initial condition, both parameterized by some parameter $\theta \in \mathbb{R}^e$. The *solution* of the IVP is a function $y_\theta : [0, T] \to \mathbb{R}^d$ that satisfies the ODE and the initial condition for the given parameter θ.

As the ODE solution y_θ is typically not available in closed form, it has to be approximated with one of many well-established numerical methods. These include both classic, non-probabilistic approaches, in which the ODE solution is approximated by a single point estimate $\hat{y}_\theta(t)$ (see e.g. Hairer et al. (1993)), as well as more recent probabilistic numerical methods, which return a posterior distribution over the ODE solution $p(y(t) \mid \theta)$ (see e.g. Hennig et al. (2022)). The posterior in the latter approach captures the uncertainty in the ODE solution due to the numerical approximation of the solver, which for ODEs is typically a discretization of the time domain. In this paper, we follow the probabilistic numerical perspective on ODE solvers, but we take an additional source of uncertainty into account, namely uncertainty in the initial value problem itself.

In real-world applications, the initial value problem is rarely known exactly and both the initial value and the vector field parameters are often subject to uncertainty.

This uncertainty can be dealt with in two ways: If measurements of the system trajectory are available, we can try to reduce the uncertainty and learn about the system by inferring the parameter θ from data, for example via maximum likelihood estimation or Bayesian inference. This is also known as the "parameter inference" problem, and it has been studied extensively over the years, both in non-probabilistic numerics (Bard, 1974; Ramsay et al., 2007), and in the context of probabilistic numerical ODE solvers (Kersting et al., 2020a; Tronarp et al., 2022; Wu and Lysy, 2024). On the other hand, if no data is available, we can compute a distribution over possible system trajectories of the uncertain system by propagating the uncertainty from the system parameters to the ODE solution itself. This is the problem of interest in this paper, and we refer to it as the "uncertainty propagation" problem.

Previous work on uncertainty propagation mainly resides in the non-probabilistic numerics literature. Gerlach et al. (2020) consider the uncertainty propagation problem from a Koopman operator perspective, and propagate uncertainty from the initial value to the solution of the ODE by combining ODE solvers with numerical quadrature. Uncertainty propagation also plays a role in certain machine learning problems where the ODE vector field is learned from data, e.g. with a Gaussian process (Ridderbusch et al., 2023). The problem also relates to the broader field of *random ODEs*, where the ODE vector field is itself a random process, but this is a much more general setting than the one we consider here (Han and Kloeden, 2017). In the probabilistic numerics literature, Pförtner et al. (2022) have propagated uncertainty through linear PDEs with Gaussian-process-based methods. On nonlinear ODEs on the other hand, filtering-based approaches have been estab-

lished as an efficient class of methods (Kersting et al., 2020b; Schober et al., 2019; Tronarp et al., 2019)—but to the best of our knowledge, uncertainty propagation with these methods has not been studied in detail yet.

In this paper, we present a novel approach for uncertainty propagation with probabilistic numerical ODE solvers. We formalize the uncertainty propagation problem in the context of probabilistic numerical ODE solvers and show how existing methods are not able to propagate uncertainty out-of-the-box—despite them being *probabilistic*. We then propose a new approach to uncertainty propagation by combining filtering-based ODE solvers with numerical quadrature. We experimentally validate our method on multiple dynamical systems and show how the resulting uncertainty estimates closely match the true propagated uncertainty. We also show how the numerical uncertainty of the ODE solver enters the total propagated uncertainty and how it can prevent overconfidence compared to non-probabilistic ODE solvers.

2 Filtering-based Probabilistic Numerical ODE Solvers

In this section we provide a brief introduction to filtering-based probabilistic numerical ODE solvers, also known as *ODE filters*. Given an ODE-IVP as in Eq. (1) together with a known, fixed parameter vector θ, we aim to compute posterior distributions of the form

$$p\big(y(t) \mid y(0) = c_\theta, \{\dot{y}(t_n) = f_\theta(y(t_n), t_n)\}_{n=1}^N\big), \quad (2)$$

where $\{t_n\}_{n=1}^N \subset [0, T]$ is a chosen discretization of the time domain. We refer to this quantity, and approximations thereof, as the *probabilistic numerical solution* of the IVP.

In a nutshell, ODE filters approach this problem by posing it as a Gauss–Markov regression problem and solving it with Bayesian filtering and smoothing methods. In the following, we briefly introduce the building blocks of ODE filters, namely the Gauss–Markov process prior, the observation models, and the resulting discrete-time inference problem; for a more complete introduction to the topic, refer to Tronarp et al. (2019). We then briefly discuss the current limitations of these methods for uncertainty propagation.

2.1 Gauss–Markov process prior

ODE filters model the ODE solution $y(t)$ as a Gauss–Markov process, defined as the output of a linear time-invariant stochastic differential equation (LTI-SDE)

$$x(0) \sim \mathcal{N}\big(x(0); \mu_0^-, \Sigma_0^-\big), \quad (3a)$$
$$dx(t) = Fx(t)\, dt + \Gamma^{1/2}\, dw(t), \quad (3b)$$
$$y(t) = E_0 x(t), \quad (3c)$$
$$\dot{y}(t) = E_1 x(t), \quad (3d)$$

where $x(t) \in \mathbb{R}^D$ is the *state* of the process at time t of dimension D, $\mu_0 \in \mathbb{R}^D$ is the initial state mean, $\Sigma_0 \in$

$\mathbb{R}^{D \times D}$ is the initial state covariance, $F \in \mathbb{R}^{D \times D}$ is the drift matrix, $\Gamma^{1/2} \in \mathbb{R}^{D \times d}$ is the dispersion matrix, $w : \mathbb{R}_{\geq 0} \to \mathbb{R}^d$ is a standard Wiener process, and $E_0, E_1 \in \mathbb{R}^{\bar{d} \times D}$ are projection matrices that map the state to the ODE solution and its derivative, respectively.

In discrete time, LTI-SDEs satisfy exactly discrete-time transition densities of the form

$$x(t + h) \mid x(t) \sim \mathcal{N}(x(t + h); \Phi(h)x(t), Q(h)), \quad (4)$$

where $\Phi(h), Q(h)$ can be derived from the drift and dispersion matrices F, Γ (Särkkä and Solin, 2019).

In this paper, we consider ODE filters that model the solution with a q-times integrated Wiener process prior, which has transition and covariance matrices given by (Kersting et al., 2020b)

$$\Phi(h) = I_d \otimes \breve{\Phi}(h), \quad (5a)$$
$$Q(h) = I_d \otimes \kappa^2 \breve{Q}(h), \quad (5b)$$

where

$$\big[\breve{\Phi}(h)\big]_{ij} = \mathbb{I}_{i \leq j} \frac{h^{j-1}}{(j-i)!}, \quad (6a)$$
$$\big[\breve{Q}(h)\big]_{ij} = \frac{h^{2q+1-i-j}}{(2q+1-i-j)(q-i)!(q-j)!}, \quad (6b)$$

leading to a state dimension of $D = d \cdot (q+1)$. The *diffusion parameter* $\kappa^2 \in \mathbb{R}_{>0}$ will typically be estimated by the ODE filter via quasi-maximum likelihood estimation (see e.g. Tronarp et al. (2019); Bosch et al. (2021)). The solution $y(t)$ and its derivative $\dot{y}(t)$ can be extracted from the state $x(t)$ via multiplication with the projection matrices

$$E_0 = I_d \otimes \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \end{bmatrix}^\top, \quad (7a)$$
$$E_1 = I_d \otimes \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \end{bmatrix}^\top. \quad (7b)$$

But other Gauss–Markov priors are also suitable for ODE filtering, such as the integrated Ornstein–Uhlenbeck processes or Matérn processes. Refer to Tronarp et al. (2021) or Bosch et al. (2023) for more details.

2.2 IVP information operators

To relate the Gauss–Markov prior to the solution of the IVP, we need to enforce both the initial value and the ODE itself. In ODE filters, this is done by introducing two observation models:

$$z_0 \mid x(0) \sim \delta\big(E_0 x(0) - c_\theta\big), \quad (8)$$
$$z(t) \mid x(t) \sim \delta\big(E_1 x(t) - f\big(E_0 x(t), t, \theta\big)\big). \quad (9)$$

If $x(t)$ satisfies the IVP, then we have $z_0 = 0$ as well as $z(t) = 0$ for all $t \in [0, T]$. Thus, conversely, if we were to condition $x(t)$ on $z_0 = 0$ and $z(t) = 0$, then the resulting process would satisfy the IVP. Unfortunately this posterior is not tractable, but we will show how to approximate it in the next section.

Remark 1 (Initialization with higher-order derivatives)**.** *For improved accuracy and numerical stability, the state can also be initialized on higher-order derivatives at the*

initial value. These can be computed efficiently via Taylor-mode automatic differentiation (Krämer and Hennig, 2024), and we can then extend the initial observation model to include these higher-order derivatives. In our implementation, we use this strategy and initialize the solver on all derivatives up to order q.

2.3 Probabilistic numerical ODE solutions as Bayesian filtering and smoothing

Instead of attempting to satisfy the ODE everywhere, we discretize the time domain into a grid $\{t_n\}_{n=1}^N \subset [0, T]$ and only condition the process on ODE observations at these time points. This leads to the following inference problem:

$$x(0) \sim \mathcal{N}\big(x(0); \mu_0^-, \Sigma_0^-\big), \tag{10a}$$
$$x(t_{n+1}) \mid x(t_n) \sim \mathcal{N}(\Phi(h_n)x(t_n), Q(h_n)), \tag{10b}$$
$$z_0 \mid x(0) \sim \delta\left(E_0 x(0) - c_\theta\right), \tag{10c}$$
$$z(t_n) \mid x(t_n) \sim \delta\left(E_1 x(t_n) - f(E_0 x(t_n), t_n, \theta)\right), \tag{10d}$$

where $h_n = t_{n+1} - t_n$ is the step size for the n-th time step, and with observations $z_0 = 0$ and $z(t_n) = 0$ for all n. This problem is known as a nonlinear Gauss–Markov regression problem, and its solution can be approximated efficiently with Bayesian filtering and smoothing, for instance with an extended Rauch–Tung–Striebel smoother. This yields a Gaussian process posterior with marginals

$$p\big(x(t) \mid z_0 = 0, \{z(t_n) = 0\}_{n=1}^N, \theta\big) \approx \mathcal{N}(\mu_\theta(t), \Sigma_\theta(t)). \tag{11}$$

or more compactly,

$$p(x(t) \mid \mathcal{D}_{\mathrm{PN}}, \theta) \approx \mathcal{N}(\mu_\theta(t), \Sigma_\theta(t)), \tag{12}$$

where $\mathcal{D}_{\mathrm{PN}}$ denotes the data. The posterior ODE solution $p(y(t) \mid \mathcal{D}_{\mathrm{PN}}, \theta)$ can then be extracted from Eq. (12) via projection with E_0; this provides an approximation to Eq. (2) and is thus a *probabilistic numerical solution* of the IVP, computed by the ODE filter.

2.4 Can ODE filters propagate Gaussian uncertainty out of the box?

The ODE filters we introduced above formulate "ODE solving" as probabilistic inference. But, contrary to what one might think at first, these methods do not automatically also solve the uncertainty propagation problem of interest: The posterior computed by the ODE filter, shown in Eq. (12), conditionally depends on the parameter θ, but the desired posterior in the uncertainty propagation problem of interest requires its marginalization, i.e. $p(y(t) \mid \mathcal{D}_{\mathrm{PN}}) = \int p(y(t) \mid \mathcal{D}_{\mathrm{PN}}, \theta)p(\theta) \,\mathrm{d}\theta$. Even in the case in which only the initial value of the IVP is uncertain this additional marginalization is necessary; Fig. 1 demonstrates this visually with a simple linear example. In the following, we will show why this is generally the case and how Baysian filters do not *propagate uncertainty* but rather *estimate states*.

To this end, we consider a generic state-space model with only a single time step:

$$\begin{array}{lll} \text{Initial state:} & p(x_0), & \text{(13a)} \\ \text{Transition model:} & p(x_1 \mid x_0), & \text{(13b)} \\ \text{Observation model:} & p(z_1 \mid x_1). & \text{(13c)} \end{array}$$

Our goal here is now to compute the distribution over the state x_1 given the observation z_1, while also propagating the uncertainty from the initial state x_0 to the state x_1. That is, we want to compute

$$p_{\mathrm{goal}}(x_1 \mid z_1) = \int p(x_1 \mid z_1, x_0)p(x_0) \,\mathrm{d}x_0. \tag{14}$$

To relate this to the given state-space model, we can expand the conditional probability with Bayes' rule and obtain

$$p_{\mathrm{goal}}(x_1 \mid z_1) = \int \frac{p(z_1 \mid x_1) \, p(x_1 \mid x_0)}{p(z_1 \mid x_0)} \, p(x_0) \,\mathrm{d}x_0. \tag{15}$$

Now if we were to simply apply a generic Bayesian filter to this problem (e.g. described by Särkkä (2013)), we would first compute the *predict* step and obtain

$$p_{\mathrm{predict}}(x_1) = \int p(x_1 \mid x_0)p(x_0) \,\mathrm{d}x_0, \tag{16}$$

and then copmute the *update* step which yields

$$p_{\mathrm{filter}}(x_1 \mid z_1) = \frac{p(z_1 \mid x_1) \, p_{\mathrm{predict}}(x_1)}{p(z_1)}; \tag{17}$$

or altogether, we compute

$$p_{\mathrm{filter}}(x_1 \mid z_1) = \int \frac{p(z_1 \mid x_1) \, p(x_1 \mid x_0)}{p(z_1)} \, p(x_0) \,\mathrm{d}x_0. \tag{18}$$

Comparing Eq. (18) to Eq. (15), we see that the normalization constant differs between the two expressions and thus both quantities differ, i.e. $p_{\mathrm{filter}} \neq p_{\mathrm{goal}}$. Thus, a general Bayesian filter does not actually propagate the uncertainty from the initial state forward in time, but rather reduces the uncertainty by estimating the state from the observations at each time step—or in short, it performs *state estimation* and not *uncertainty propagation*. Figure 1 also visualizes this difference with a simple linear example. Overall, this shows the need for a modification or extension of the existing ODE filtering algorithm to perform uncertainty propagation.

3 Propagating Uncertainty through ODE Filters

This section introduces the uncertainty propagation problem of interest and presents a general approach to approximate its solution with probabilistic numerical ODE solvers. Consider an initial value problem (IVP) as given in Eq. (1) and let $p(\theta)$ be a distribution over the system parameter θ. Our goal in uncertainty propagation is then to compute the expected value and variance of the ODE solution $y_\theta(t)$ with respect to the parameter θ
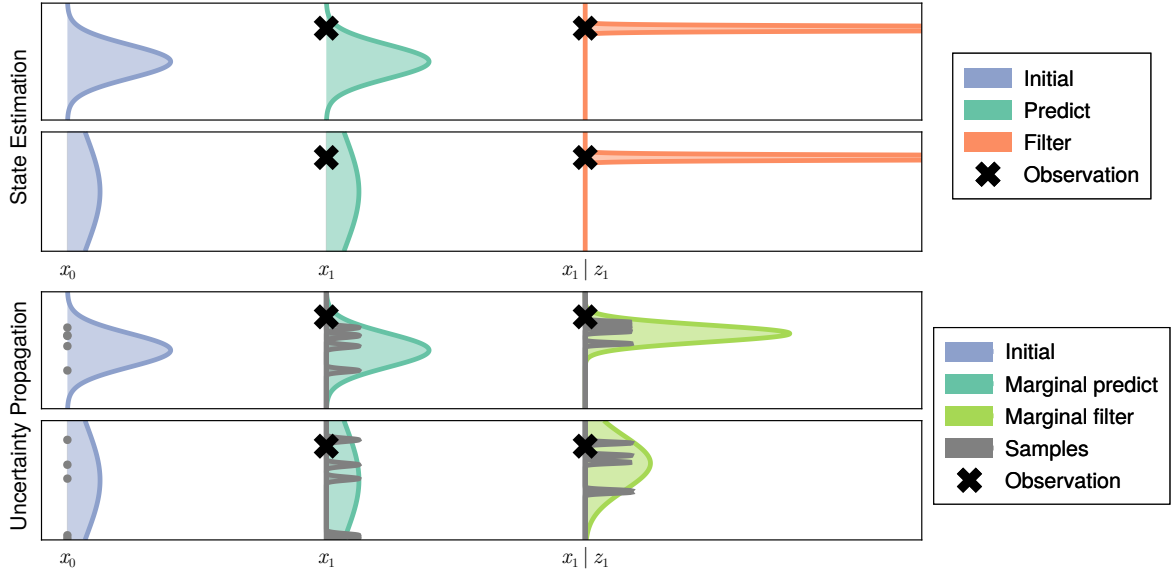
Figure 1: *Difference between state estimation and uncertainty propagation.* We consider two simple linear state-space models, both with transition model $p(x_1 \mid x_0) = \mathcal{N}(x_0, 0.01)$, observation model $p(y_1 \mid x_1) = \mathcal{N}(x_0, 0.01)$, and observation $y_1 = 2$, and with two different initial distributions $p(x_0) = \mathcal{N}(0, 1)$ and $p(x_0) = \mathcal{N}(0, 10)$. The top rows show the distributions computed by a regular Kalman filter applied to these models, which solves the corresponding *state estimation* problems. The bottom rows instead show the solutions to the *uncertainty propagation* problem, in which the initial state should not be inferred but marginalized out. This aspect is visualized by the samples in gray, which are drawn from the initial distribution, propagated forward by running a separate Kalman filter for each sample, and then aggregated. In addition, the figure also demonstrates how the filter distribution barely changes when the initial uncertainty is increased, whereas the marginal filter distribution becomes significantly wider.

distribution $p(\theta)$, which correspond to expectations of the form

$$\mathbb{E}[g(y_\theta(t))]_{p(\theta)} = \int g(y_\theta(t))p(\theta)\, \mathrm{d}\theta \qquad (19)$$

with $g(y) = y$ and $g(y) = (y - \mathbb{E}[y])^2$ for the expectation and variance, respectively, for any chosen time point $t \in [0, T]$ (or a collection thereof).

As the true ODE solution $y_\theta(t)$ is not available in closed form, we approximate it with the probabilistic numerical ODE solution $p(y(t) \mid \theta)$ and compute its moments instead, while also marginalizing out the numerical uncertainty of the ODE solver itself. This yields

$$\mathbb{E}[g(y_\theta(t))]_{p(\theta)} \approx \int \int g(y(t))p(y(t) \mid \theta)\, \mathrm{d}y(t)\, p(\theta)\, \mathrm{d}\theta. \qquad (20)$$

Equation (20) now involves only known quantities, but the nested integrals are in general intractable. In this section, we present our two-step approach to approximate these integrals via numerical integration.

Re-ordering the terms in Eq. (20), we obtain

$$\mathbb{E}[g(y_\theta(t))]_{p(\theta)} \approx \int g(y(t)) \left(\int p(y(t) \mid \theta)p(\theta)\, \mathrm{d}\theta \right) \mathrm{d}y(t). \qquad (21)$$

Our proposed two-step approach now proceeds as follows: First, we approximate the marginal PN solution

$$p(y(t)) = \int p(y(t) \mid \theta)p(\theta)\, \mathrm{d}\theta \qquad (22)$$

with numerical integration; this yields a Gaussian mixture distribution. Second, we compute the expectation and variance of the Gaussian mixture distribution and obtain the desired moments of the ODE solution posterior. We will discuss each of these steps in more detail in the following.

3.1 Approximating the ODE solution posterior with numerical integration

As the parameter θ enters the ODE solution $p(y(t) \mid \theta)$ in a nonlinear way, the integral

$$p(y(t)) = \int p(y(t) \mid \theta)p(\theta)\, \mathrm{d}\theta \qquad (23)$$

is again intractable. One common way to approximate intractable integrals is with numerical quadrature. In a nutshell, a numerical quadrature scheme approximates an integral by a weighted sum of evaluations of the integrand at a finite set of nodes, i.e.

$$\int p(y(t) \mid \theta)p(\theta)\, \mathrm{d}\theta \approx \sum_{i=1}^{N} w_i \cdot p(y(t) \mid \theta_i), \qquad (24)$$

with *nodes* $\theta_i \in \mathbb{R}^e$ and weights $w_i \in \mathbb{R}$. These nodes and weights are provided by the chosen numerical integration scheme, and the quality of the approximation depends on the number of nodes and the choice of the nodes and weights.

In this paper, we focus on Gaussian parameter distributions $p(\theta) = \mathcal{N}(\mu_\theta, \Sigma_\theta)$ and mainly consider the *third-degree spherical cubature* scheme (Genz and Monahan, 1999; Lu and Darmofal, 2004), given by

$$\theta_i = \begin{cases} \mu_\theta + \sqrt{d}\Sigma_\theta^{1/2}e_i, & \text{for } i = 1, \dots, d, \\ \mu_\theta - \sqrt{d}\Sigma_\theta^{1/2}e_{i-d}, & \text{for } i = d+1, \dots, 2d, \end{cases} \qquad (25\text{a})$$

$$w_i = \frac{1}{2d}, \qquad \text{for } i = 1, \dots, 2d, \qquad (25\text{b})$$

where d is the dimension of the parameter space, e_i are the canonical basis vectors in $\mathbb{R}^d$, and where $\Sigma_\theta^{1/2}$ denotes the left Cholesky factor of Σ_θ, i.e. $\Sigma_\theta = \Sigma_\theta^{1/2}(\Sigma_\theta^{1/2})^\top$.

This scheme is well-suited for Gaussian parameter distributions $p(\theta)$ and is known to be quite efficient in moderate to high nonlinear systems (Särkkä, 2013). But other numerical quadrature methods could also be used, as discussed in Remark 2.

Remark 2 (Other numerical quadrature methods). *Choosing a suitable quadrature method typically depends on the integrand and the parameter distribution. For Gaussian parameter distributions,* sigma point methods *such as spherical cubature, the unscented transform (Julier and Uhlmann, 1997), or Gauss–Hermite quadrature (Abramowitz and Stegun, 1968) are particularly well-suited. For non-Gaussian parameter distributions,* generalized Gaussian quadrature rules *can be used to approximate the integral, such as Gauss–Legendre quadrature for uniform distributions or Gauss–Laguerre quadrature for exponential distributions (Golub and Welsch, 1969; Stroud et al., 1966). For more general distributions, if the functional form of $p(\theta)$ is known, one can approximate the integral by combining Gauss–Hermite Quadrature with importance sampling (adaptive GHQ). Otherwise, Monte Carlo integration simply averages the integrand over samples from the distribution and is thus a general-purpose method that can be used for any distribution, but is typically less efficient than quadrature methods.*

By applying the chosen numerical quadrature scheme to the ODE solution posterior $p(y(t))$, we obtain a weighted sum of the form

$$p(y(t)) \approx \sum_{i=1}^{N} w_i p(y(t) \mid \theta_i), \qquad (26)$$

with w_i, θ_i as in Eq. (25). Since each individual probabilistic numerical ODE solution is a Gaussian process posterior with Gaussian marginals $p(y(t) \mid \theta_i) = \mathcal{N}(\mu_i(t), \Sigma_i(t))$, the mixture distribution $p(y(t))$ is a Gaussian mixture distribution of the form

$$p(y(t)) \approx \sum_{i=1}^{N} w_i \, \mathcal{N}(\mu_i(t), \Sigma_i(t)). \qquad (27)$$

3.2 Computing the expectation and covariance of the ODE solution posterior

Now that we approximated the ODE solution posterior with a Gaussian mixture distribution as in Eq. (27), we can compute its expectation and variance in closed form: The mean of a mixture distribution is given by the weighted sum of the means of the mixture components, i.e.

$$\mathbb{E}[y(t)]_{p(y(t))} = \sum_{i=1}^{N} w_i \mu_i(t), \qquad (28)$$

and its covariance is given by

$$\mathbb{V}[y(t)]_{p(y(t))} =$$
$$\sum_{i=1}^{N} w_i \left[\Sigma_i(t) + (\mu_i(t) - \bar{\mu}(t)) (\mu_i(t) - \bar{\mu}(t))^\top \right], \qquad (29)$$

where $\mu_i(t), \Sigma_i(t)$ are the mean and covariance of the i-th mixture Gaussian component at time t, respectively, and with $\bar{\mu}(t) = \mathbb{E}[y(t)]_{p(y(t))}$ as in Eq. (28). This yields the desired moments of the ODE solution with respect to the parameter distribution $p(\theta)$ and the numerical uncertainty of the ODE solver itself and concludes the algorithm.

Remark 3 (The role of numerical uncertainty and relation to non-probabilistic numerical simulators). *The proposed algorithm can also be applied to non-probabilistic ODE solvers. If we interpret non-probabilistic numerical solvers as solvers that return a Dirac distribution $p(y(t) \mid \theta) = \delta(y(t) - \hat{y}_\theta(t))$, the derivations above still hold, and the only difference is that the term in the covariance, that corresponds to the numerical uncertainty of the ODE solver, vanishes:*

$$\mathbb{V}[y(t)]_{p(y(t))} =$$
$$\sum_{i=1}^{N} w_i \left[\cancel{\Sigma_i(t)} + (\mu_i(t) - \bar{\mu}(t)) (\mu_i(t) - \bar{\mu}(t))^\top \right], \qquad (30)$$

These results correspond exactly to the uncertainty propagation through non-probabilistic numerical solvers approach proposed by Gerlach et al. (2020) (for our particular choice of quadrature scheme). We will expore this aspect and the utility of the ODE solver's uncertainty further in Section 4.2.

4 Experiments

This section evaluates the proposed probabilistic uncertainty propagation method on a range of dynamical systems. We first validate the proposed uncertainty propagation method on multiple dynamical systems and qualitatively compare the resulting mean and variance estimates to an accurate reference solution (Section 4.1). We then investigate the utility of the numerical uncertainty estimates provided by the underlying ODE solver by disentangling the resulting numerical uncertainty into PN and non-PN parts (Section 4.2).

Implementation details. In our implementation we build on the ODE filters provided by the ProbNum python package (Wenger et al., 2021), which follow a number of practices for numerically stable implementation (Krämer and Hennig, 2024). Reference solutions are computed with SciPy (Virtanen et al., 2020). All experiments were conducted on standard consumer-grade CPU. Code for the implementation and experiments is publicly available on GitHub[1].

4.1 Propagating uncertainty through nonlinear systems

We apply the proposed uncertainty propagation method to a variety of nonlinear ODE systems, including the Logistic system, the FitzHugh–Nagumo equation, Lotka–

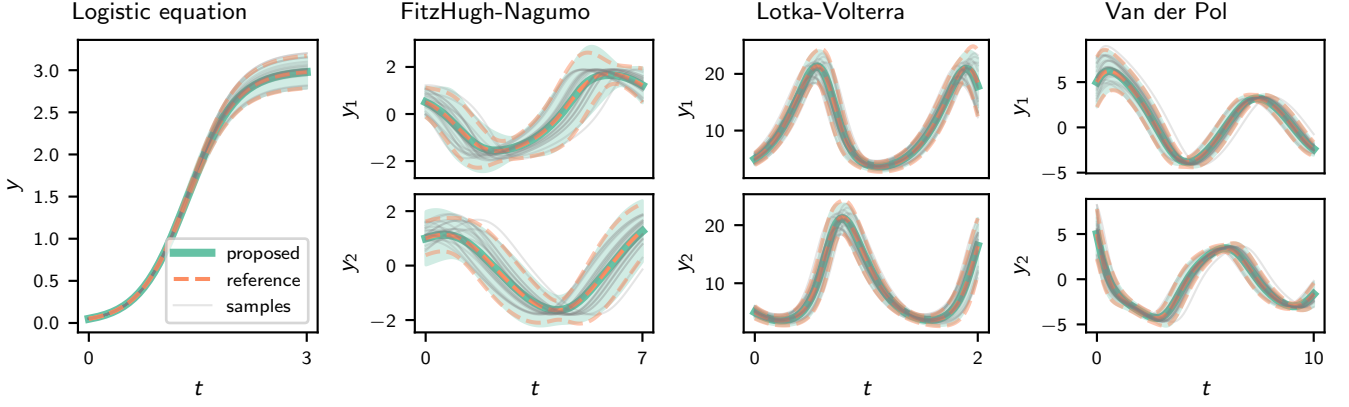[1] https://github.com/DDCoan/pn-ode-up

Figure 2: *Propagated uncertainties for the Logistic equation, FitzHugh–Nagumo system, Lotka–Volterra system, and Van der Pol oscillator.* The figure shows both the mean (colored lines) and the 95% credible interval (shaded area) computed by our proposed method ("proposed") and by the reference solution ("reference") for each system. We observe that our method returns mean and variance estimates that largely align with the reference solution.

Table 1: Uncertain initial value problems used in the experiments.

Name	Vector field	Initial value	Vector field parameters
Linear	$f(y,t) = ay + b$	$y(0) \sim \mathcal{N}(1, 0.01)$	$a=1,\ b=0$
Logistic	$f(y,t) = ay\left(1 - \frac{y}{b}\right)$	$y(0) = 0.05$	$a=3,\ b\sim\mathcal{N}(3, 0.01)$
FitzHugh–Nagumo	$f(t,y) = \begin{bmatrix} y_1 - \frac{1}{3}y_1^3 - y_2 + a \\ \frac{1}{d}(y_1 + b - cy_2) \end{bmatrix}$	$y(0) \sim \mathcal{N}\left(\begin{bmatrix} 0.5 \\ 1 \end{bmatrix}, 0.1 \cdot \mathrm{I}_2\right)$	$a=0,\ b=0.08,\ c=0.07,$ $d=1.25$
Lotka–Volterra	$f(y,t) = \begin{bmatrix} ay_1 - by_1y_2 \\ -cy_2 + dy_1y_2 \end{bmatrix}$	$y(0) \sim \mathcal{N}\left(\begin{bmatrix} 5 \\ 5 \end{bmatrix}, 0.3 \cdot \mathrm{I}_2\right)$	$a=5,\ b=0.5,\ c=5,\ d=0.5$
Van der Pol	$f(y,t) = \begin{bmatrix} y_2 \\ a \cdot (1 - y_1^2)y_2 - y_1 \end{bmatrix}$	$y(0) \sim \mathcal{N}\left(\begin{bmatrix} 5 \\ 5 \end{bmatrix}, 2 \cdot \mathrm{I}_2\right)$	$a=0.05$

Volterra system and the Van der Pol oscillator. Table 1 provides more information for each system.

To apply our proposed method, we use an extended Kalman filtering-based ODE filter (also known as EK1) as the underlying ODE solver, with a once-integrated Wiener process prior and with automatic calibration of the time-constant diffusion parameter; all provided by the ProbNum python package (Wenger et al., 2021). We use step sizes of 10^{-2} for the logistic ODE and for Van der Pol, and $5 \cdot 10^{-3}$ for FitzHugh–Nagumo and Lotka–Volterra. For numerical quadrature, we use the spherical cubature scheme introduced in Section 3.1, which requires $2d$ applications of the ODE solver for each system, where d is the dimension of the uncertain parameter space. The reference solution for each system is computed with Monte Carlo integration over a classic ODE solver (LSODA as provided by scipy; Virtanen et al. (2020)), using 100,000 samples and small time steps of 0.001.

Figure 2 shows the results. The mean and variance estimates by our probabilistic uncertainty propagation method largely align with the reference solution. This demonstrates that our proposed method provides reliable uncertainty propagation results across multiple nonlinear systems. In Section A, we show that our proposed method can also deliver reliable uncertainty propagation results for non-Gaussian distributed parameters.

4.2 Investigating the utility of numerical uncertainty for uncertainty propagation

In this section, we investigate the utility of the numerical uncertainty estimates provided by the underlying ODE solver. To this end, we revisit the expression for the computed variance from Eq. (29), and decompose it into two parts:

$$\mathbb{V}[y(t)]_{p(y(t))} =$$
$$\underbrace{\sum_{i=1}^{N} w_i \Sigma_i(i)}_{\text{PN}} + \underbrace{\sum_{i=1}^{N} w_i \left(\mu_i(t) - \bar{\mu}(t)\right)\left(\mu_i(t) - \bar{\mu}(t)\right)^\top}_{\text{non-PN}}.$$
$$(31)$$

The non-PN part corresponds exactly to the variance that one would compute without any numerical ODE solver uncertainty (see also Remark 3). The PN part on the other hand depends only on the ODE solver uncertainties, and uses these to inflate the resulting total covariance. In the following, we visualize this variance decomposition to show the utility of the numerical uncertainty estimates provided by the ODE solver.

We consider a linear ODE and a Lotka–Volterra system as described in Table 1, with time spans of $[0,3]$ and $[0,0.5]$ respectively, and with a reduced initial value variance of $10^{-5} \cdot \mathrm{I}_2$ for the Lotka–Volterra problem. We consider the same ODE filter as in Section 4.1 for the linear problem and use an ODE filter with twice-
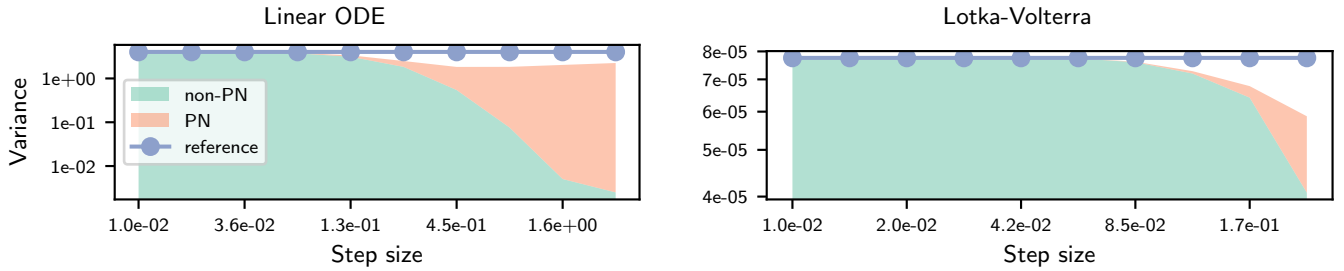
Figure 3: *Decomposition of the propagated variance into its non-PN and PN parts.* For both the linear system and the Lotka–Volterra ODE, the non-PN variance decreases for larger step sizes and thereby diverges from the accurate reference variance. The PN variance on the other hand, which depends on the numerical uncertainty of the ODE filter, grows for increasing step sizes and helps fill the gap between the non-PN variance and the reference variance to reduce the resulting overconfidence.

integrated Wiener process for the Lotka–Volterra ODE. The reference variance for Lotka–Volterra is again computed by Monte Carlo integration over a classic ODE solver (LSODA as provided by scipy; Virtanen et al. (2020)), with 100,000 samples and small time steps of 0.001.

For the linear ODE, we can compute the ground-truth variance analytically. We then compute the propagated uncertainties for the final time step with our proposed method for 10 different ODE solver step sizes, equally-spaced in log-space in $[0.01, 3]$ for the linear ODE and $[0.01, 0.25]$ for the Lotka–Volterra ODE, respectively, and decompose the total variance into the PN and non-PN parts.

Figure 3 shows the resulting variance decompositions. On the considered problems, the non-PN part decreases with increasing step sizes and diverges from the accurate reference variance. But as the numerical error of the ODE solver grows with increasing step sizes, the PN part grows for increasing step sizes and helps fill the gap between the non-PN variance and the reference variance. This shows that the numerical uncertainty estimates provided by the ODE solver can help to prevent overconfidence in the propagated uncertainty estimates.

5 Limitations

The proposed method and the evaluation shown in this paper have several limitations.

Quantification of the quadrature error. While we build on probabilistic numerical ODE solvers to quantify errors due to discretization in the time-domain, we do not account for the numerical error due to numerical quadrature. Bayesian quadrature methods could be used instead to approximate both the integral and the integration error (Hennig et al., 2022), but their application to our problem setting is more involved than simply replacing the numerical quadrature scheme and requires a fully probabilistic treatment of the problem, with joint priors and inference over both time and the ODE parameter. We leave this for future work.

Higher-order moments and generic expectations. Our proposed method only computes the mean and

variance of the ODE solution with respect to the parameter distribution, but expectations $\mathbb{E}[g(y(t))]_{p(\theta)}$ for a general function g are also often of interest. These can in general not be computed analytically for a Gaussian mixture distribution. One possible approach could be to use a second numerical quadrature scheme to approximate this expectation, but this is left for future work.

6 Conclusion

This paper presents a novel approach for propagating uncertainty through filtering-based probabilistic numerical ODE solvers. While existing probabilistic ODE solvers provide uncertainty estimates for numerical errors, we show that they do not automatically solve the uncertainty propagation problem. Our method addresses this limitation by combining filtering-based ODE solvers with numerical quadrature to properly marginalize over uncertain system parameters.

The experimental results demonstrate that our approach successfully propagates uncertainty through various non-linear dynamical systems, with the resulting uncertainty estimates closely matching reference solutions. Furthermore, we show that the numerical uncertainty estimates provided by probabilistic ODE solvers serve an important role in preventing overconfidence, particularly when using larger step sizes. This work demonstrates that probabilistic numerical methods can provide meaningful uncertainty quantification not just for numerical errors, but also for uncertainty propagation in dynamical systems.

Acknowledgements

References

M. Abramowitz and I. A. Stegun. *Handbook of mathematical functions with formulas, graphs, and mathematical tables*, volume 55. US Government printing office, 1968.

Y. Bard. *Nonlinear parameter estimation*, volume 1209. Academic press New York, 1974.

N. Bosch, P. Hennig, and F. Tronarp. Calibrated adaptive probabilistic ODE solvers. In *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*, volume 130 of *Proceedings of Machine Learning Research*. PMLR, 2021.

N. Bosch, P. Hennig, and F. Tronarp. Probabilistic exponential integrators. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

A. Genz and J. Monahan. A stochastic algorithm for high-dimensional integrals over unbounded regions with Gaussian weight. *Journal of computational and applied mathematics*, 112(1-2), 1999.

A. R. Gerlach, A. Leonard, J. Rogers, and C. Rackauckas. The koopman expectation: An operator theoretic method for efficient analysis and optimization of uncertain hybrid dynamical systems. *arXiv preprint arXiv:2008.08737*, 2020.

G. H. Golub and J. H. Welsch. Calculation of Gauss quadrature rules. *Mathematics of computation*, 23 (106), 1969.

E. Hairer, S. Norsett, and G. Wanner. *Solving Ordinary Differential Equations I: Nonstiff Problems*, volume 8. Springer-Verlag, 1993. ISBN 978-3-540-56670-0. doi: 10.1007/978-3-540-78862-1.

X. Han and P. E. Kloeden. *Random Ordinary Differential Equations and Their Numerical Solution*. Springer Publishing Company, Incorporated, 1st edition, 2017. ISBN 9789811062643.

P. Hennig, M. A. Osborne, and H. P. Kersting. *Probabilistic Numerics: Computation as Machine Learning*. Cambridge University Press, 2022. doi: 10.1017/9781316681411.

S. J. Julier and J. K. Uhlmann. New extension of the Kalman filter to nonlinear systems. In *Signal processing, sensor fusion, and target recognition VI*, volume 3068. Spie, 1997.

H. Kersting, N. Krämer, M. Schiegg, C. Daniel, M. Tiemann, and P. Hennig. Differentiable likelihoods for fast inversion of 'Likelihood-free' dynamical systems. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*. PMLR, 2020a.

H. Kersting, T. J. Sullivan, and P. Hennig. Convergence rates of Gaussian ODE filters. *Statistics and Computing*, 30(6), 2020b. ISSN 1573-1375. doi: 10.1007/s11222-020-09972-4.

N. Krämer and P. Hennig. Stable implementation of probabilistic ODE solvers. *Journal of Machine Learning Research*, 25(111), 2024.

J. Lu and D. L. Darmofal. Higher-dimensional integration with Gaussian weight for applications in probabilistic design. *SIAM Journal on Scientific Computing*, 26(2), 2004.

M. Pförtner, I. Steinwart, P. Hennig, and J. Wenger. Physics-informed gaussian process regression generalizes linear pde solvers. *arXiv preprint arXiv:2212.12474*, 2022.

J. O. Ramsay, G. Hooker, D. Campbell, and J. Cao. Parameter estimation for differential equations: a generalized smoothing approach. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 69(5), 2007.

S. Ridderbusch, S. Ober-Blöbaum, and P. Goulart. The past does matter: correlation of subsequent states in trajectory predictions of Gaussian process models. In *Proceedings of the Thirty-Ninth Conference on Uncertainty in Artificial Intelligence*, volume 216 of *Proceedings of Machine Learning Research*. PMLR, 2023.

S. Särkkä. *Bayesian Filtering and Smoothing*. Bayesian Filtering and Smoothing. Cambridge University Press, 2013. ISBN 9781107030657.

M. Schober, S. Särkkä, and P. Hennig. A probabilistic model for the numerical solution of initial value problems. *Statistics and Computing*, 29(1), 2019. ISSN 1573-1375. doi: 10.1007/s11222-017-9798-7.

A. H. Stroud, D. Secrest, et al. *Gaussian quadrature formulas*, volume 374. Prentice-Hall Englewood Cliffs, NJ, 1966.

S. Särkkä and A. Solin. *Applied Stochastic Differential Equations*. Institute of Mathematical Statistics Textbooks. Cambridge University Press, 2019. doi: 10.1017/9781108186735.

F. Tronarp, H. Kersting, S. Särkkä, and P. Hennig. Probabilistic solutions to ordinary differential equations as nonlinear Bayesian filtering: a new perspective. *Statistics and Computing*, 29(6), 2019. doi: 10.1007/s11222-019-09900-1.

F. Tronarp, S. Särkkä, and P. Hennig. Bayesian ODE solvers: the maximum a posteriori estimate. *Statistics and Computing*, 31(3), 2021. ISSN 1573-1375. doi: 10.1007/s11222-021-09993-7.

F. Tronarp, N. Bosch, and P. Hennig. Fenrir: Physics-enhanced regression for initial value problems. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*. PMLR, 2022.

P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov,

A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, İ. Polat, Y. Feng, E. W. Moore, J. Vander-Plas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17, 2020. doi: 10.1038/s41592-019-0686-2.

J. Wenger, N. Krämer, M. Pförtner, J. Schmidt, N. Bosch, N. Effenberger, J. Zenn, A. Gessner, T. Karvonen, F.-X. Briol, et al. Probnum: Probabilistic numerics in python. *arXiv preprint arXiv:2112.02100*, 2021.

M. Wu and M. Lysy. Data-adaptive probabilistic likelihood approximation for ordinary differential equations. In *Proceedings of The 27th International Conference on Artificial Intelligence and Statistics*, volume 238 of *Proceedings of Machine Learning Research*. PMLR, 2024.

A Additional experiment: Propagating uniform uncertainty through non-linear systems

Figure 4 provides additional experimental results for uncertainty propagation for uniform parameters. The parameters are sampled from a uniform distribution $\text{Unif}[\mu - 1.96\sigma, \mu + 1.96\sigma]$, where μ and σ are chosen as the means and standard deviations of the Gaussian distributions described in Table 1. The uncertainty is propagated using 1000 Monte Carlo samples. Other implementational settings follow the details provided in Section 4.
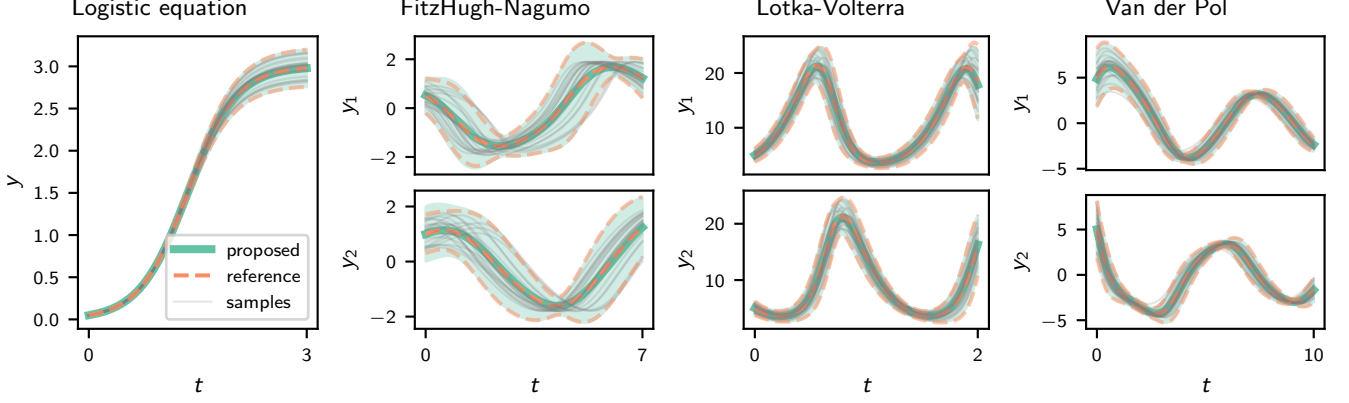

Figure 4: *Propagated uncertainties for the Logistic equation, FitzHugh–Nagumo system, Lotka–Volterra system, and Van der Pol oscillator for non-Gaussian parameters.* The figure shows both the mean (colored lines) and the 95% credible interval (shaded area) computed by our proposed method ("proposed") and by the reference solution ("reference") for each system. We observe that our method returns mean and variance estimates that largely align with the reference solution.