# A probabilistic model for the numerical solution of initial value problems

**Michael Schober**[1] · **Simo Särkkä**[2] · **Philipp Hennig**[1]

## Abstract

We study connections between ordinary differential equation (ODE) solvers and probabilistic regression methods in statistics. We provide a new view of probabilistic ODE solvers as active inference agents operating on stochastic differential equation models that estimate the unknown initial value problem (IVP) solution from approximate observations of the solution derivative, as provided by the ODE dynamics. Adding to this picture, we show that several multistep methods of Nordsieck form can be recast as Kalman filtering on $q$-times integrated Wiener processes. Doing so provides a family of IVP solvers that return a Gaussian posterior measure, rather than a point estimate. We show that some such methods have low computational overhead, nontrivial convergence order, and that the posterior has a calibrated concentration rate. Additionally, we suggest a step size adaptation algorithm which completes the proposed method to a practically useful implementation, which we experimentally evaluate using a representative set of standard codes in the DETEST benchmark set.

## 1 Introduction

Numerical algorithms estimate intractable quantities from tractable ones. It has been pointed out repeatedly (Poincaré 1896; Diaconis 1988; O'Hagan 1992) that this process is structurally similar to statistical inference, where the tractable computations play the role of data in statistics, and the intractable quantities relate to latent, inferred quantities. In recent years, the search for numerical algorithms which return probability distributions over the solution for a given numerical problem has become an active area of research (Hennig et al. 2015). Several models and methods have been proposed for the solution of initial value problems (IVPs) (Skilling 1992; Chkrebtii et al. 2016; Schober et al. 2014a; Conrad et al. 2017; Kersting and Hennig 2016; Teymur et al. 2016). However, these probabilistic algorithms have no immediate connection to the extensive literature on this task in numerical analysis. Most importantly, such inference algorithms do not come with convergence analysis out of the box. The methods in Chkrebtii et al. (2016), Conrad et al. (2017) and Teymur et al. (2016) have convergence results, but their respective implementations are based on sampling schemes and, thus, do not offer guarantees for individual runs. The methods in Schober et al. (2014a) and Kersting and Hennig (2016) offer a deterministic execution and an analytical guarantee for the first step, but we will show that this guarantee is lacking for the whole integration domain.

In this paper, we present a class of probabilistic solvers which combine properties of the standard and the probabilistic algorithms. We formulate desiderata that users might have for a probabilistic numerical algorithm. We present one construction that fulfills these desiderata and we provide a MATLAB code[1] which we compare empirically against other available codes. The construction uses the algebra of Gaussian inference to provide a Gaussian posterior distribution over the solution of an IVP. In particular, we show that the posterior mean can be understood as a multistep method in Nordsieck representation, and thus, analytical results about

✉ Michael Schober
michael.schober5@de.bosch.com

1 Max Planck Institute for Intelligent Systems,
72076 Tübingen, Germany

2 Aalto University, Espoo, Finland

---

1 https://pn.is.tuebingen.mpg.de/code/pfos.

these methods carry over to the present algorithm. Additionally, we propose to interpret the posterior covariance as a *measure of uncertainty* or *error estimator* and argue that this interpretation can be analytically justified. In the context of a larger pipeline of empirical studies and numerical computations, the framework of probability modeling provides a common language to analyze the epistemic confidence in its result (Cockayne et al. 2017). In the framework of Cockayne et al. (2017), the code provides approximate Bayesian uncertainty quantification (Sullivan 2015) at low computational overhead and almost complete backwards compatibility to the MATLAB IVP solver suite.

## 1.1 Problem description

We study the problem of finding a real-valued curve $y : \mathbb{T} \to \mathbb{R}$ over an interval $\mathbb{T} = [t_0, T]$ such that

$$y'(t) = \frac{\mathrm{d}y}{\mathrm{d}t}(t) = f(t, y(t)) \text{ (“the ODE”)}, \tag{1a}$$

and

$$y(t_0) = y_0 \text{ (“the initial value”)}, \tag{1b}$$

with $f$ Lipschitz continuous with constant $L$ in the second argument and sufficiently many times differentiable in its second argument. Users might be interested in approximations to $y$ on either a predefined mesh $\Delta_S \subset \mathbb{T}$ or an automatically selected mesh $\Delta \subset \mathbb{T}$ of finitely many intermediate function values. The derivations will be presented with a scalar-valued problem, but the results carry over to the multivariate case.

IVPs are a particularly deeply studied class of ODE-related tasks. Part of their significance is due to the Picard–Lindelöf theorem which guarantees local unique existence of solutions. As a consequence, IVPs lend themselves to be solved by so-called *step-by-step methods*, where the solution is advanced iteratively on expanding meshes $\Delta_{n+1} := (\{t_0, \ldots, t_n\} \cup \{t_{n+1}\}) \supset \Delta_n$. The knots $t_n$ of a mesh are either generated on a regular grid $t_n := t_0 + hn$, $n = 0, \ldots, N$ for some $N \in \mathbb{N}$ and $h = (T - t_0)N^{-1}$ or the step size $h$ may very per step, thus yielding $t_n = t_0 + \sum_{i=1}^{n} h_i$.

To construct a probabilistic numerical method, we define the following list of desiderata that an algorithm should fulfill. These properties will be defined and motivated in turn below.

**Probabilistic inference** The computations should be operations on probability distributions.
**Global definition** The probabilistic model should not depend on the discretization mesh.
**Deterministic execution** When run several times on the same problem, the algorithm produces the same output each time.

**Analytic guarantees** The algorithm's output should have desirable analytic properties.
**Online execution** The algorithm execution can be extended indefinitely when required.
**Speed.** The execution time should not be prohibitively slow.
**Problem adaptiveness** The algorithm should automatically adapt parameters to problem and accuracy requirements.

Throughout this paper, we will use zero-based indexing for vectors and matrices such that a $d$-dimensional vector $\mathbf{v}$ is written as $\mathbf{v} = (v_0, \ldots, v_{d-1})^\mathsf{T}$ and the $d$ canonical basis vectors are $\mathbf{e}_0, \ldots, \mathbf{e}_{d-1}$.

## 2 From classical to probabilistic numerical algorithms

In this section, we explain and motivate the first two items from our list of desiderata in turn—*probabilistic inference* and *global definition*.

On a high-level view, numerical algorithms can be described as combinations of *tractable approximating function classes* and *computation strategies* for *informative values*. Analyses of numerical methods show to what level the approximations can converge to the true problem solution and how fast the computation strategies can be carried out. This is structurally very similar to problems in statistics where unknown quantities need to be related to *approximating function classes* via *observable informative values*. In particular, finding a function $Y = (Y_t)_{t \in \mathbb{T}}$ given a collection of information $z_n$, $n = 0, \ldots N$ about $Y_{t_n}$ at times $t_n$ is studied in *regression analysis* in statistics. In that context, the unknown function is often treated as a stochastic process and the approximating function is obtained by conditioning it on the measurements. Consequently, this paper treats the problem of finding an approximate solution $Y = (Y_t)_{t \in \mathbb{T}}$ to the true unknown solution $y(t)$ as a statistical regression problem on a stochastic process.

Accepting the probabilistic approach as a framework for plausible reasoning (Jeffreys 1969; Cox 1946; Hennig et al. 2015), we require a *probability measure* or *law* $P_Y$ over the numerical solution $Y_t$. The computations necessary for the construction of $P_Y$ should be interpretable as (approximate) probabilistic inference. When such an interpretation is admissible, we call the resulting algorithm a *probabilistic numerical method (PNM)* for the purposes of this paper. A more rigorous definition has been given by Cockayne et al. (2017). The motivation behind this requirement is that there should not be an analysis gap between statistical and numerical computations. This is particularly beneficial, when the differential equation solver is embedded in a longer chain of computations (Cockayne et al. 2017). In principle, this should allow to build fine-tuned methods adapting to sources

**Table 1** Properties of existing PNM ODE solvers

| Method | Glob. def.? | Determ.? | Guarantees? |
| --- | --- | --- | --- |
| Skilling (1992) | ✓ | × | × |
| Chkrebtii et al. (2016) | ✓ | × | ≈ |
| Schober et al. (2014a) | ≈ | ✓ | ≈ |
| Conrad et al. (2017) | × | × | ≈ |
| Kersting and Hennig (2016) | ✓ | ✓ | × |
| Teymur et al. (2016) | × | × | ≈ |
| PFOS (this paper) | ✓ | ✓ | ✓ |

of data uncertainty and computational approximation during runtime and provide richer feedback of approximation quality as recently empirically validated by Schober et al. (2014b) and Hauberg et al. (2015).

Let $z_{[n]} := \{z_k \mid k \leq n\}$ be the set of collected data up to and including step $n$. Given a *prior law* $P_Y$ over the space of solutions and a *likelihood function* $P(z_n \mid Y_{t_n})$ relating the value of the process $Y_{t_n}$ to collected data, Bayes' theorem leads to the (predictive) posterior measure

$$P_{Y \mid z_{[n]}} = \frac{P(z_{[n]} \mid Y) P_Y}{\int P(z_{[n]} \mid Y) \, \mathrm{d}P_Y}. \tag{2}$$

where $P(z_{[n]} \mid Y) = \prod_{k \leq n} P(z_k \mid Y_{t_k})$. Rigorously, the above expression is valid only for finite collections of values of $Y_t$, in which case the corresponding probability measures $P_Y$ are typically represented by their densities, but as the finite-dimensional distributions define the full measure, we use this slight abuse of notation here. In function space form, the posterior process is only defined as a Radon–Nikodym derivative with respect to the prior measure $P_Y$ which yields

$$\frac{\mathrm{d}P_{Y \mid z_{[n]}}}{\mathrm{d}P_Y} = \frac{1}{\int \exp(-\Psi(y; z)) \, \mathrm{d}P_Y} \exp(-\Psi(y; z)), \tag{3}$$

where $\Psi$ is a "potential" function analogous to a likelihood. For details, we refer the reader to Stuart (2010), Giné and Nickl (2015, §7.3). We denote the posterior distributions (typically densities) of point values of $Y$ as $P(Y_t \mid z_{[n]})$.

We propose to think about the probabilistic framework as a more informative output information than the point estimates returned by classical numerical algorithms (see also Hennig et al. 2015).

Furthermore, a probabilistic IVP solver shall be called *globally defined* on its input domain $\mathbb{T}$, if its probabilistic interpretation does not depend on the discretization mesh $\Delta$. PNMs satisfying this property provide two benefits. Users may evaluate the (predictive posterior) distribution $P(Y_t \mid z_{[n]})$ for any $t \in \mathbb{T}$. In particular, users may evaluate $P(Y_t \mid z_{[n]})$ for $t \notin \Delta$. Thus, users may request $P(Y_{t_s} \mid z_{[n]})$, $t_s \in \Delta_S$ and the support of a user-defined mesh $\Delta_S$ is not a separate requirement. Secondly, this implies

that the inference can be paused and continued after every expansion from $\Delta_n \mapsto \Delta_{n+1}$. In principle, this also enables iterative refinement of the solution quality based on its prediction uncertainty.

Table 1 lists PNM ODE solvers that have been proposed in the literature. A ✓ indicates that the method satisfies a given property, a × indicates that a method does not satisfy a given property, and a ≈ indicates that a property holds with some restrictions. The listing shows that almost all methods proposed so far are globally defined. Furthermore, we see that the definition is independent of a method being sampling based or not. The method proposed by Conrad et al. (2017) is a generative process on subintervals $[t_n, t_{n+1}] \subset \mathbb{T}$ based on a numerical discretization. It is easy to construct two different meshes $\Delta_n, \Delta'_n$ that define different distributions for $Y_t$ in the case of $y' = \lambda y$, and a general argument can be made from this example. In Teymur et al. (2016), the predictive posterior is only defined on the discretization mesh. This defect is not for lack of definition, but a consequence of the underlying numerical method the probabilistic algorithm is built upon. Since the method is defined on a windowed data frame, it is easy to construct a mesh such that the prediction $Y_t$ at time $t$ will be different depending on the window $[t_{n-i}, \ldots, t_{n+j}] \ni t$ is considered to be part of.

The analysis in Schober et al. (2014a) proposes two main modes of operation: naive chaining and probabilistic continuation. Naive chaining is not a globally defined method since mesh points $t_n$ are part of adjacent Runge–Kutta blocks, and the corresponding predictive posterior distribution $P(Y_{t_n} \mid z_{[n]})$ is different for these two blocks. Probabilistic continuation is globally defined, but there has been no convergence theory for this mode yet. This paper fills this gap.

## 2.1 State-space models for Gauss–Markov processes

Our approximate model of the true solution $y(t)$ is a vector $\mathbf{x}(t) = (y^{(0)}(t), \ldots, y^{(q)}(t))^{\mathsf{T}}$ where $y^{(i)}(t)$ is the true $i$th derivative of $y(t)$ at time $t$. We represent the prior uncertainty about $\mathbf{x}(t)$ by the distribution $P(\mathbf{X}_t)$ of the random variable $\mathbf{X}_t$—or more generally as the measure or the law

$P_{\mathbf{X}}$ of the stochastic process $\mathbf{X}$—which is then conditioned on the observed values.

The prior model, which has also been considered in Schober et al. (2014a), belongs to the class of Gauss–Markov processes. Models of this class can often be written as a *linear time-invariant (LTI) stochastic differential equation (SDE)* of the form

$$\mathrm{d}\mathbf{X}_t = \mathbf{F}\,\mathbf{X}_t\,\mathrm{d}t + \mathbf{L}\,\mathrm{d}W_t, \tag{4}$$

where $\mathbf{X}_t$ is the so-called *state* of the model, $\mathbf{F} \in \mathbb{R}^{(q+1)\times(q+1)}$ is the state feedback matrix and $\mathbf{L} \in \mathbb{R}^{(q+1)}$ is the diffusion matrix of the system. $\mathrm{d}W_t$ is the increment of a Wiener process with intensity $\sigma^2$, that is, $\mathrm{d}W_t \sim \mathcal{N}(0, \sigma^2\,\mathrm{d}t)$.

Here, we consider models where $\mathbf{L}$ is the last standard basis vector $\mathbf{e}_q$ and $\mathbf{F} = \mathbf{U}_{q+1} + \mathbf{e}_q\mathbf{f}^\mathsf{T}$ is a (transposed) companion matrix. Here, $\mathbf{U}_{q+1}$ denotes the upper shift matrix and the row vector $\mathbf{f}^\mathsf{T}$ contains the coefficients in the last row of $\mathbf{F}$. In this case, the vector-valued process $\mathbf{X}_t = (X_{t,0}, \ldots, X_{t,q})^\mathsf{T}$ obtains the interpretation $\mathbf{X}_t = (Y_t, Y_t', \ldots, Y_t^{(q)})^\mathsf{T}$, because the form of $\mathbf{F}$ and $\mathbf{L}$ implies that the realizations of $Y_t$ are $q$-times continuously differentiable on $\mathbb{R}$. Later, we will also consider scaled systems $\tilde{\mathbf{X}}_t = \mathbf{B}\mathbf{X}_t$ with an invertible linear transformation $\mathbf{B}$. In this case, we denote by $\mathbf{H}_i$ the matrix that projects onto the $i$th derivative $Y_t^{(i)} = \mathbf{H}_i\tilde{\mathbf{X}}_t := e_i^\mathsf{T}\mathbf{B}^{-1}\tilde{\mathbf{X}}_t$. Two particular models of this type are the $q$-times *integrated Wiener process* (IWP($q$)) and the continuous auto-regressive processes of order $q$. Detailed introductions can be found, for example, in Karatzas and Shreve (1991), Øksendal (2003) and Särkkä (2006). SDEs can also be seen as path-space representations of more general temporal Gaussian processes arising in machine learning models (Särkkä et al. 2013).

Models of form (4) are also related to nonparametric spline regression models (Wahba 1990) which often have a natural interpretation in frequentist analysis (Kimeldorf and Wahba 1970). Conceptually, these models are a compromise between globally defined parametric models, which might be too restrictive to achieve convergence, and local parametric models, which might be too expressive to be captured by a globally defined probability distribution. Models of this type have been studied in the literature (Loscalzo and Talbot 1967; Andria et al. 1973), but the presentation here starts from other principles.

Conditioning on (random) initial conditions $\mathbf{X}_{t_*}$ at a starting time $t_*$ of the process, the solution of Eq. (4) has the analytic form

$$\mathbf{X}_t = e^{\mathbf{F}\Delta_{t_*}^t}\mathbf{X}_{t_*} + \int_{t_*}^t e^{\mathbf{F}\Delta_\tau^t}\,\mathbf{L}\,\mathrm{d}W(\tau), \tag{5}$$

where $e^{\mathbf{F}\Delta_{t'}^t} := \sum_{k=0}^\infty [\mathbf{F}\Delta_{t'}^t]^k[k!]^{-1}$ is the matrix exponential of $\mathbf{F}\Delta_{t'}^t$ and $\Delta_s^t := t - s$.

If $\mathbf{X}_{t_*} \sim \mathcal{N}(\mathbf{m}_*, \mathbf{C}_*)$, then the distribution of $\mathbf{X}_t$ remains Gaussian for all $t$ by linearity and its statistics can be computed explicitly (Grewal and Andrews 2001; Särkkä 2006) via

$$\mathbf{m}_t := \mathbb{E}(\mathbf{X}_t) = e^{\mathbf{F}\Delta_{t_*}^t}\mathbf{m}_*$$

$$\begin{aligned}
\mathrm{cov}(\mathbf{X}_t, \mathbf{X}_{t'}) &= e^{\mathbf{F}\Delta_{t_*}^t}\mathbf{C}_*(e^{\mathbf{F}\Delta_{t_*}^{t'}})^\mathsf{T} \\
&\quad + \underbrace{\int_{t_*}^{\min(t,t')} e^{\mathbf{F}\Delta_\tau^t}\mathbf{L}\sigma^2\mathbf{L}^\mathsf{T}(e^{\mathbf{F}\Delta_\tau^{t'}})^\mathsf{T}\mathrm{d}\tau}_{=:\mathbf{Q}_{t_*}(t,t')}.
\end{aligned} \tag{6}$$

For practical purposes, only the covariance matrix $\mathbf{C}_t = \mathrm{cov}(\mathbf{X}_t, \mathbf{X}_t)$ of the states at a single time $t$ is needed.

The choice of prior measure $P_{\mathbf{X}}$ in Eq. (4) can be interpreted as a *prior assumption* or *belief* encoded in the algorithm, in the sense that the algorithm amounts to an autonomous agent. We emphasize that if one adopts this view, then the results reported in later sections amount to an external analysis of the effects of these assumptions. That is, we will show that if the agent is based on this prior measure $P_{\mathbf{X}}$ with a likelihood to be defined in Sect. 2.3, they give rise to a posterior distribution with certain desirable properties. By contrast, one could also take a more restrictive standpoint internal to the algorithm and state that the proposed method works well if the true solution $\mathbf{x}(t)$ is indeed a sample from $P_{\mathbf{X}}$. This is expressly *not* our viewpoint here, and it would be a flawed argument, too, given that in practice, $\mathbf{x}(t)$ is defined through the ODE, thus evidently not a sample from any stochastic process.

Denote by $\mathbf{A}(h) := e^{\mathbf{F}\Delta_t^{t+h}}$ the *discrete transition matrix* of step size $h$ and $\mathbf{Q}(h) := \mathbf{Q}_t(t+h, t+h)$ the *discrete diffusion matrix* of step size $h$, respectively. For LTI SDE systems, $\mathbf{A}(h)$ and $\mathbf{Q}(h)$ fulfill matrix-valued differential equations which can be solved analytically via matrix fraction decomposition (Grewal and Andrews 2001; Särkkä 2006). If we define

$$\mathbf{\Phi}(h) = \begin{pmatrix} \mathbf{\Phi}_{11}(h) & \mathbf{\Phi}_{12}(h) \\ \mathbf{\Phi}_{21}(h) & \mathbf{\Phi}_{22}(h) \end{pmatrix} := \exp\left\{\begin{pmatrix} \mathbf{F} & \sigma^2\mathbf{L}\mathbf{L}^\mathsf{T} \\ \mathbf{0} & -\mathbf{F}^\mathsf{T} \end{pmatrix} h\right\}, \tag{7}$$

then the matrices $\mathbf{A}(h)$ and $\mathbf{Q}(h)$ are given by

$$\mathbf{A}(h) = \exp(\mathbf{F}h), \quad \mathbf{Q}(h) = \mathbf{\Phi}_{12}(h)\mathbf{\Phi}_{22}^{-1}(h). \tag{8}$$

Above, $\mathbf{\Phi}_{22}^{-1}(h)$ can be computed efficiently: from the two properties of the matrix exponential, $\exp(\mathbf{X})^{-1} = \exp(-\mathbf{X})$ and $\exp(\mathbf{X}^\mathsf{T}) = \exp(\mathbf{X})^\mathsf{T}$, it follows that $\mathbf{\Phi}_{22}^{-1}(h) = \mathbf{A}(h)^\mathsf{T}$, and therefore, $\mathbf{Q}(h) = \mathbf{\Phi}_{12}(h)\mathbf{A}(h)^\mathsf{T}$. In the following, it will be beneficial to write $\mathbf{Q}(h)$ as $\mathbf{Q}(h)^\mathsf{T} = \mathbf{A}(h)\mathbf{\Phi}_{12}(h)^\mathsf{T}$, which is valid since $\mathbf{Q}(h)$ is symmetric.

For the rest of this paper, we will focus on the $q$-times integrated Wiener process IWP($q$), which is defined by

$$\mathrm{d}\mathbf{X}_t = \mathbf{U}_{q+1}\mathbf{X}_t \, \mathrm{d}t + \mathbf{e}_q \, \mathrm{d}W_t. \tag{9}$$

In this case, $\mathbf{f}^\mathsf{T} = (0, \dots, 0)$ and there is no feedback from higher states $X_{t,i}$ to lower states $X_{t,j}$, $i < j$. In particular, this process is nonstationary and does not revert to the initial mean $\mathbf{m}_{t_*}$. In this system, $\mathbf{A}(h)$ and $\mathbf{Q}(h)$ can be be computed analytically

$$\begin{aligned}
(\mathbf{A}(h))_{i,j} &= \mathbb{I}_{i \le j} \frac{h^{j-i}}{(j-i)!}, \\
(\mathbf{Q}(h))_{i,j} &= \sigma^2 \frac{h^{2q+1-i-j}}{(2q+1-i-j)(q-i)!(q-j)!},
\end{aligned} \tag{10}$$

which can be derived directly from Eq. (6).

## 2.2 Data generation mechanism

Many problems in statistics assume the existence of an externally produced, thus fixed data set $\{(t_n, z_n) \mid t_n \in \Delta\}$ and develop appropriate solutions from there. An analogous concept in numerical algorithms for solving differential equations would be to pose a global discretization scheme and to obtain a solution with other tools from numerical analysis. Methods of this type are often applied to boundary value problems (BVPs) and partial differential equations (PDEs) where the integration domains need to be specified a priori in any case. Cockayne et al. (2017) take this approach by assuming a fixed information operator $A$. However, there are cases where the end $T$ of the integration domain $\mathbb{T}$ cannot be stated beforehand, when the quantity of interest depends on a qualitative behavior of the solution. For example, in modeling of chemical reactions a user might be interested in the long-term behavior of the compounds and it is unknown when the reaction reaches equilibrium.

In contrast, many numerical IVP solvers proceed in a step-by-step manner. Having computed a numerical approximation $P_{Y \mid z_{[n]}}$ on the mesh $\Delta_n$, a *prediction* $y_{n+1}^-$ of $y(t_{n+1})$ is used to *evaluate* $f(t_{n+1}, y_{n+1}^-)$ and the resulting output $z_{n+1}$ is used to *update* the approximation $P_{Y \mid z_{[n+1]}}$ on the extended mesh $\Delta_{n+1}$. For example, in a deterministic IVP the data $(t_0, y_0)$ can be used to construct the observation $z_0 = f(t_0, y_0)$ which satisfies the probabilistic interpretation of $y'(t_0) \sim \delta(z_0 - y'(t_0))$. This serves as a corner case for the general situation. Setting $t_{-1} := t_0$ and $z_{-1} := y_0$, it follows that $y(t_0) \sim \delta(z_{-1} - y(t_{-1}))$ and the initial value requires almost no special treatment. The concept is illustrated in Algorithm 1 and can, in principle, be extended indefinitely, at constant cost per step. The term *predict–evaluate–correct (PEC)* or *predictor–corrector* methods have a more technical meaning in classic textbooks (Hairer et al. 1987; Deuflhard and Bornemann 2002), but the idea is common to many numerical IVP solvers. Chkrebtii et al. (2016) calls the process of evaluating $f(t_n, y_{t_n}^-)$ with tentative $y_{t_n}^-$ to generate $z_n$ a *model interrogation*. From a statistical perspective, this concept of active model interrogation is similar to the sequential analysis of Wald (1973) and Owhadi and Scovel (2016).

Algorithm 1 conveys the general idea of a probabilistic ODE solver while omitting parameter tuning aspects like error control and step size selection. The exact form of line 5 depends on the choice of observation construction and data likelihood model. Without data, the prior induces a probability distribution on the hidden state $\mathbf{X}_{t_n}$. It remains to construct an observation $z_n$ and a likelihood model $P(z_n \mid \mathbf{X}_{t_n})$.

## 2.3 Observation assumptions

Recall from Sect. 2.1 the prior state-space assumption

$$\mathbf{X}_t = (Y_t, Y_t', \dots, Y_t^{(q)})^\mathsf{T} \sim \mathcal{N}(\mathbf{m}_t, \mathbf{C}_t). \tag{11}$$

Combining Eqs. (1) and (11) gives

$$\begin{aligned}
P(Y_t') &= f(t, \cdot) \circ \mathcal{N}(Y_t; (\mathbf{m}_t)_0, (\mathbf{C}_t)_{00}) \tag{12} \\
&\ne \mathcal{N}(Y_t'; (\mathbf{m}_t)_1, (\mathbf{C}_t)_{11}) \tag{13}
\end{aligned}$$

where Eq. (12) denotes the transformed random variable. The exact form of that push-forward is not usually tractable for general $f$ (with the exception of linear ODEs, which of course do not require nontrivial numerical algorithms).

We will show below, however, that replacing the push-forward with an approximate inference step captured by a Gaussian likelihood leads to good analytic properties of the resulting Gaussian posterior. This likelihood, which ignores the recursive nature of the ODE (Eq. 12 and Fig. 1), will be parametrized as

$$P(z_n \mid Y_{t_n}') = \mathcal{N}\left(z_n; Y_{t_n}', R_n^2\right) \tag{14}$$

where $z_n$ are the observations that have yet to be constructed and $R_n^2$ can be interpreted as an observation uncertainty. Another way to phrase Eq. (14) is to write

$$z_n = \mathbf{H}_1 \mathbf{X}_{t_n} + \nu \tag{15}$$

where the latent variable $\nu := y'(t_n) - f(t_n, \mathbf{H}_0 \mathbf{X}_{t_n})$ captures the error between $f$ at the estimated solution and the true solution's derivative. The approximation in Eq. (14) is to assign a centered Gaussian density $P(\nu) = \mathcal{N}(\nu; 0, R_n^2)$ to this latent variable. Purely from a formal perspective, this $\nu$ is a "random variable", but we stress again that $P(\nu)$ captures *uncertainty* arising from lack of computational information about a deterministic quantity, not any physical sort of randomness in a frequentist sense. That is, solving the same IVP several times will always produce the exact same $\nu$, because the algorithm is deterministic. But that same $\nu$ will always be just as unknown. Repeated runs will not refine the uncertainty. Figure 1 displays a graphical model corresponding

---

**Algorithm 1:** Active probabilistic model

1: Define $t_{-1} := t_0$, $z_{-1} := y_0$ and probabilistic model $P_{\mathbf{X}}$
2: Compute $P(\mathbf{X}_{t_{-1}} \mid z_{[-1]})$.  {Add initial value information}
3: **for** n = 0 **to** $N$ **do**
4:   Compute $P(\mathbf{X}_{t_n} \mid z_{[n-1]}) \propto P(\mathbf{X}_{t_n} \mid \mathbf{X}_{t_{n-1}}) P(\mathbf{X}_{t_{n-1}} \mid z_{[n-1]})$  {**Predict** $t_n$}
5:   Compute observation model $P(z_n \mid \mathbf{X}_{t_n}) = \text{observe}(f, P(\mathbf{X}_{t_n} \mid z_{[n-1]}))$  {Evaluate/interrogate model}
6:   Compute $P(\mathbf{X}_{t_n} \mid z_{[n]}) \propto P(z_n \mid \mathbf{X}_{t_n}) P(\mathbf{X}_{t_n} \mid z_{[n-1]})$  {**Update** information}
7: **end for**
8: **return** $\{P(\mathbf{X}_{t_n} \mid z_{[n]}), \; n = -1, 0, \ldots, N\}$

---
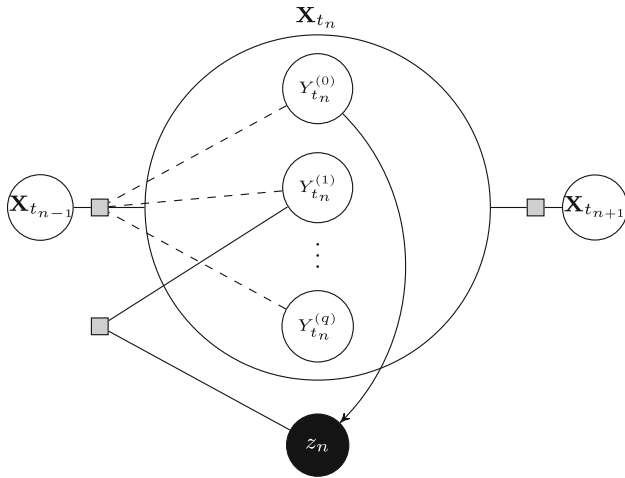


**Fig. 1** The graphical model corresponding to the proposed construction. White circles represent unobserved hidden states, and the black circle represents the observed data. Gray squares represent a jointly normal distribution. The arrow indicates a model interrogation. An implied non-Gaussian factor between $Y^{(0)}(t_n)$ and $z_n$ is ignored to obtain a practical algorithm

to the construction. All current probabilistic numerical ODE solvers share this particular assumption (14) (Skilling 1992; Chkrebtii et al. 2016; Schober et al. 2014a; Conrad et al. 2017; Kersting and Hennig 2016; Teymur et al. 2016). The differences between these algorithms chiefly lie in the prior on $\mathbf{X}_t$, and how the observation $z_n$ is produced within the algorithm.

It remains to construct $z_n$ and $R_n^2$. One possible way to achieve this is to compute the expected value of vector field $f$ under the prediction for the true solution

$$z_n \leftarrow \int f(t_n, Y_{t_n}) \, \mathcal{N}(Y_{t_n}; (\mathbf{m}_{t_n}^-)_0, (\mathbf{C}_{t_n}^-)_{00}) \, dY_{t_n}, \qquad (16)$$

where $\mathcal{N}(\mathbf{X}_{t_n}; \mathbf{m}_{t_n}^-, \mathbf{C}_{t_n}^-) = P(\mathbf{X}_{t_n} \mid z_{[n-1]})$ is the prediction distribution of $\mathbf{X}_{t_n}$ given the data $z_{[n-1]}$ and $\leftarrow$ denotes assignment in code.

With these conventions, two new issues emerge: the evaluation of the intractable Eq. (16) and the determination of $R_n^2$. Kersting and Hennig (2016) propose to put

$$R_n^2 \leftarrow \int f(t_n, Y_{t_n})^2 \, \mathcal{N}(Y_{t_n}; (\mathbf{m}_{t_n}^-)_0, (\mathbf{C}_{t_n}^-)_{00}) \, dY_{t_n} - \mathbb{E}[f]^2 \qquad (17)$$

and to evaluate both integrals by Bayesian quadrature. Chkrebtii et al. (2016) method draws a sample $u_n \sim \mathcal{N}((\mathbf{m}_{t_n}^-)_0, (\mathbf{C}_{t_n}^-)_{00})$, computes $z_n \leftarrow f(t_n, u_n)$, and $R_n^2$ is set to $(\mathbf{C}_{t_n}^-)_{11}$. In light of Kersting and Hennig (2016), this could be thought of as a form of Monte Carlo scheme to evaluate (16).

As a further restriction to the likelihood (14) more widely used by other probabilistic numerical solvers, we will here focus on models with $R_n^2 \to 0$. That is

$$\begin{aligned} z_n &\leftarrow f(t_n, (\mathbf{m}_{t_n}^-)_0), \\ P(z_n \mid Y_{t_n}') &= \delta(z_n - Y_{t_n}') = \mathcal{N}(z_n; Y_{t_n}', 0), \end{aligned} \qquad (18)$$

This means the estimation node $y_{t_n}^-$ for the evaluation of $f$ is simply the current mean prediction, and the resulting observation is modeled as being correct.

From the analytical viewpoint external to the algorithm itself, of course, one does not expect that the model assumption of a Gaussian likelihood, much less one with vanishing width, to hold in reality. The point of the analysis in Sect. 3.1 is to demonstrate that this model and evaluation scheme yield a method satisfying sufficient conditions to prove that its point estimate converges at a nontrivial order for some choices of state spaces, while simultaneously keeping computational cost very low (that is very similar to that of classic multistep solvers). That is because the predictive posterior distributions $P(\mathbf{X}_{t_n} \mid z_{[n]})$ can be computed by the linear-time algorithm known as *Kalman filtering* (Kalman 1960; Särkkä 2006, 2013). The marginal predictive posterior distributions given all data $P(\mathbf{X}_t \mid z_{[N]})$ can be computed using the *Rauch–Tung–Striebel smoothing* equations (Rauch et al. 1965; Särkkä 2006, 2013). Simultaneously, one can draw samples from the full joint posterior. These two operations increase the computational cost minimally: They require additional computations comparable to those used for interpolation in classic solvers, but neither smoothing nor sampling requires additional evaluations of $f$. The computational complexity stays linear in number of data points collected. If the full joint posterior is also required for

some reason, this is also possible to construct (Solin 2016; Grigorievskiy et al. 2016). As a second consequence, the computation becomes deterministic which enables unit testing of the resulting code.

As a side remark, we note some obvious restrictions of the combination of Gaussian (process) prior and likelihood used here: Since this combination means the posterior is always a Gaussian process, one cannot hope to accurately capture bifurcation events, higher-order correlations in the discretization errors or other higher-order effects.

## 2.4 Detailed example

Consider a concrete example. We solve the following IVP

$$
\begin{aligned}
y' &= f(t, y) = f(y) = ry(1 - y/K), \\
y(t_0) &= y_0 = 1/10, \quad r = 3, K = 1,
\end{aligned}
\tag{19}
$$

on the interval $[0, 1.5]$. Equation (19) is the sigmoid logistic growth function. Its solution is available in closed form

$$
y(t) = \frac{K y_0 \exp(rt)}{K + y_0(\exp(rt) - 1)}.
$$

To solve this system, we apply a 2-times integrated Wiener process. For this example, we fix $h_n = h = 0.3$, such that $t_n = t_0 + hn$ for all $n$. Usually, the initial values are chosen to be $\mathbf{m}_{t_{-1}}^- = \mathbf{0}$ and $\mathbf{C}_{t_{-1}}^- = \mathbf{Q}(\infty)$, which is the so-called steady state for stationary processes (Hartikainen and Särkkä 2010). The latter does not exist in the case for the integrated Wiener process, since the IWP is not stationary. However, as has been shown in Schober et al. (2014a), this can be done analytically, collecting the first $q$ derivative observations $z_0, \ldots, z_{q-1}$ manually in the interval $[t_0, t_1]$ and then inserting them in the analytic formulas, yielding the filtering distribution $P(\mathbf{X}_{t_1} \mid z_{[q-1]}) = \mathcal{N}(\mathbf{X}_{t_1}; \mathbf{m}_{t_1}, \mathbf{C}_{t_1})$ (see also Sect. 3.2). The remainder of the interval $[t_1, \ldots, t_N = T]$ is solved with the familiar Kalman filter equations

$$
\mathbf{m}_{t_n}^- = \mathbf{A}(h)\mathbf{m}_{t_{n-1}},
\tag{20}
$$

$$
\mathbf{C}_{t_n}^- = \mathbf{A}(h)\mathbf{C}_{t_{n-1}}\mathbf{A}(h)^\mathsf{T} + \mathbf{Q}(h)
\tag{21}
$$

and

$$
\lambda_n = f(t_n, \mathbf{H}_0\mathbf{m}_{t_n}^-) - \mathbf{H}_1\mathbf{m}_{t_n}^-,
\tag{22}
$$

$$
\mathbf{K}_n = \mathbf{C}_{t_n}^-\mathbf{H}_1^\mathsf{T}[\mathbf{H}_1\mathbf{C}_{t_n}^-\mathbf{H}_1^\mathsf{T}]^{-1},
\tag{23}
$$

$$
\mathbf{m}_{t_n} = \mathbf{m}_{t_n}^- + \mathbf{K}_n\lambda_n,
\tag{24}
$$

$$
\mathbf{C}_{t_n} = \mathbf{C}_{t_n}^- - \mathbf{K}_n[\mathbf{H}_1\mathbf{C}_{t_n}^-\mathbf{H}_1^\mathsf{T}]\mathbf{K}_n^\mathsf{T}.
\tag{25}
$$

Figure 2 shows the state of the algorithm after 2 steps have been taken. The solution looks discontinuous, because

the information of later updates $z_n$ has not been propagated to previous time points $t_m, m < n$. The last column of Fig. 2 shows the *(predictive posterior) smoothing distribution* wherein all the information is globally available.

# 3 Classical analysis for the probabilistic method

The most important test for any numerical algorithm is that it works in practice and under the requirements of potential users. The proposed probabilistic numerical algorithm has been motivated and derived from the computational properties that established classical algorithms provide. The classical algorithms have been studied intensely for over a century, to a point where the theory could almost be considered complete (Gear 1981). Thus, a newly proposed algorithm—even when motivated from a different background—should stand up to classical analysis.

While many specialized models and algorithms have been proposed, two standard classes of algorithms have become prevalent for the solution of (1): *Runge–Kutta (RK)* methods and *(linear) multistep methods (LMMs)* or combinations thereof (general linear methods, GLMs Butcher 1985). These classes share a similar type of algorithmic structure and analysis: At time $t_n$, evaluate $f$ with a numerical approximation $y_{t_n}$ to construct an updated numerical approximation $y_{t_{n+1}}$ from linear combinations of the function evaluations $f_{t_n}$ (exact definitions below). The update weights are parameters of a given method and, if chosen appropriately, can be shown to coincide with the Taylor approximation of the true solution $y$ up to $q$ terms.

In the following, we present results relating the newly proposed probabilistic method to existing algorithms, which allows us to transfer the known results to our method. Interpreting the probabilistic model from the viewpoint of classical analysis adds a justification to the assumptions made in the previous sections by saying that these assumptions—unintuitive as they may be at first—are the same assumptions that are implied by the application of a classical algorithm.

## 3.1 On the connection to Nordsieck methods

Linear multistep methods are defined by the relationship

$$
\sum_{i=0}^{q} \alpha_i y_{t_{n-i}} = h \sum_{i=0}^{q} \beta_i f_{t_{n-i}},
\tag{26}
$$

where $f_{t_n}$ are approximations to $y'(t_n)$, $h$ is the step size and the $\alpha_i$ and $\beta_i$ are parameters of the method. If $\beta_0 = 0$, then (26) defines $y_{t_n}$ without requiring $f_{t_n}$ and we can set $f_{t_n} := f(t_n, y_{t_n})$ for the computation of $y_{t_{n+1}}$. This is called
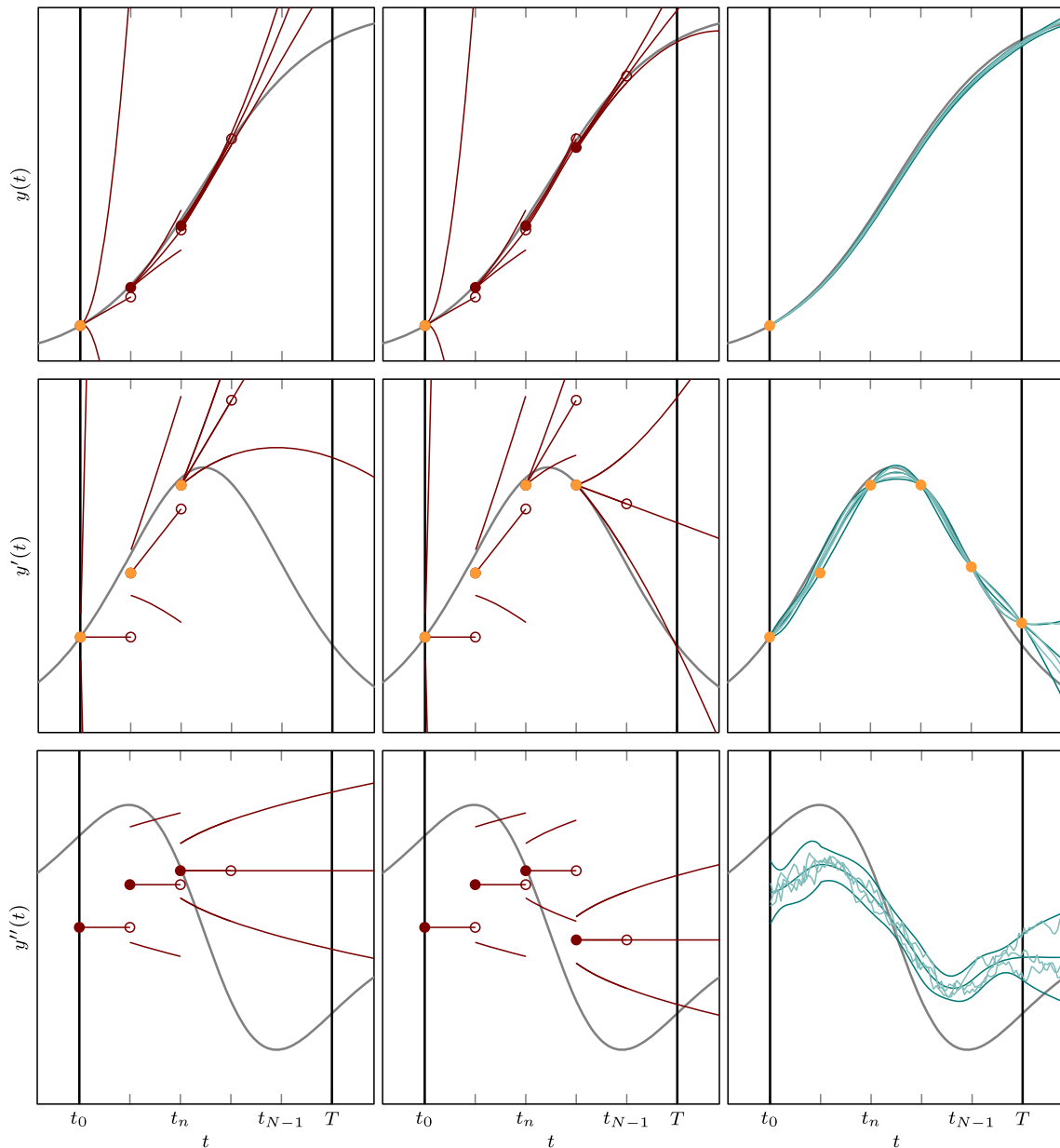
**Fig. 2** The 2-times integrated Wiener process $d\mathbf{X}_t = \mathbf{U}_3\mathbf{X}_t \, dt + \mathbf{e}_2 \, dW_t$ applied to the logistic growth problem $y' = ry(1 - y/K)$. The plot shows the true solution (gray) of the function $y$ and its first two derivatives, as well as the numerical solution $Y$, given by its mean $\mathbf{m}_i$ (red line) and covariance $\mathbf{C}$, visualized as point-wise plusminus twice the standard deviation $\mathbf{m}_i \pm 2\sqrt{\mathbf{C}_{ii}}$. Empty circles are predicted values at time

$t_n$, filled circles represent updated values, and orange dots are function and derivative observations. The first two columns display two *predict–evaluate–update–predict* cycles. The last column shows the smoothed final solution (green, thick lines) and three samples from the predictive posterior (thin lines). (Color figure online)

an *explicit method*. In contrast, if $\beta_0 \neq 0$, we still define $f_{t_n} = f(t_n, y_{t_n})$, but (26) now defines a nonlinear equation for $y_{t_n}$ given a nonlinear $f$. We say that $y_{t_n}$ is *implicitly defined* and, therefore, methods with $\beta_i \neq 0$ are called *implicit methods*. Assuming that at least one of $\{\alpha_q, \beta_q\}$ does not vanish, the method requires the numerical approximation on $q$ previous grid points and (26) is called a $q$-step method.

Skeel showed in 1979 that implicit LMMs can be written in Nordsieck 1962 form:

$$\mathbf{x}_n = \left(y_{t_n}, hy'_{t_n}, \dots, h^q y^{(q)}_{t_n}/q!\right)^{\mathsf{T}}, \tag{27}$$

$$\mathbf{x}_{n+1} = \left(\mathbf{I} - \mathbf{l}\mathbf{e}_1^{\mathsf{T}}\right)\mathbf{P}\mathbf{x}_n + h\mathbf{l}z_n, \tag{28}$$

where $\mathbf{P}$ is the Pascal triangle matrix with entries $p_{ij} = \mathbb{I}_{i \leq j}\binom{j}{i}$ and $\mathbf{l} = \mathbf{l}(\{\alpha_i, \beta_i\})$ is the weight vector defining the

method. The vector $\mathbf{x}_n$ is called the *Nordsieck vector* in honor of its inventor (Nordsieck 1962), and a LMM written in Nordsieck form is also called a *Nordsieck method*. The intuition behind this family of methods is to replace $y(t)$ or $y'(t)$ on $[t_{n-q}, \ldots, t_n]$ with a local polynomial $\pi(t) = \pi_{[t_{n-q}, \ldots, t_n]}(t)$ of order $q$.

The difference in presentation between (26) and (27) can be understood as expressing $\pi(t)$ in either Lagrange notation (Eq. 26) or Taylor expansion notation (Eq. 27). In this case, $\mathbf{P}\mathbf{x}_n$ yields a prediction of the numerical Taylor expansion at $t_{n+1}$ and the scalar increment $z_n$ is implicitly defined as the solution to

$$h^{-1}(\mathbf{P}\mathbf{x}_n)_1 + l_1 z_n = f(t_n + h, (\mathbf{P}\mathbf{x}_n)_0 + h l_0 z_n), \qquad (29)$$

which is the correction from $\mathbf{x}_n$ to $\mathbf{x}_{n+1}$ to the Taylor coefficients. Equation (29) can be solved by iterated function evaluations of the form

$$z_n^{(1)} := f(t_n + h, (\mathbf{P}\mathbf{x}_n)_0) \qquad (30)$$

$$z_n^{(M)} := l_1^{-1} \left[ f\left(t_n + h, (\mathbf{P}\mathbf{x}_n)_0 + h l_0 z_n^{(M-1)}\right) - h^{-1}(\mathbf{P}\mathbf{x}_n)_1 \right] \qquad (31)$$

or by directly solving (29) with some variant of the Newton–Raphson method.

If $z^{(M)}$ is used in the computation of (28), the resulting algorithm is called a $P(EC)^M$ method. If Eq. (29) is solved up to numerical precision, the method is called a $P(EC)^\infty$ method. Nordsieck methods with suitable weights $\mathbf{l}$ can be shown to have local truncation error of order $q$ or $q + 1$ (Skeel and Jackson 1977; Skeel 1979). More details can also be found in standard textbooks (Hairer et al. 1987; Deuflhard and Bornemann 2002).

We will now show how the Kalman filter (20)–(25) can be rewritten such that the mean prediction takes the form of (28). This enables to analyze the proposed algorithm in light of classical Nordsieck method results, but can also guide the further development of the probabilistic approach with the experience of existing software.

Considering a fixed step size $h_n = h$, $n = 1, \ldots, N$, we rescale the state space and SDE of the IWP($q$) by scaling matrix $\mathbf{B}$ to define an equivalent notation

$$\tilde{\mathbf{X}}_t = \left(Y_t, hY_t', \frac{h^2}{2!}Y_t'', \ldots \frac{h^q}{q!}Y_t^{(q)}\right)^\mathsf{T}$$
$$= \underbrace{\mathrm{diag}\left(1, h, \frac{h^2}{2!}, \ldots \frac{h^q}{q!}\right)}_{=:\mathbf{B}} \mathbf{X}_t, \qquad (32)$$

This state vector is the *Nordsieck vector*. The advantage of this notation is that (4) simplifies to

$$\mathrm{d}\tilde{\mathbf{X}}_t = \mathbf{B}\mathbf{U}_{q+1}\mathbf{B}^{-1}\tilde{\mathbf{X}}_t \, \mathrm{d}t + \mathbf{B}\mathbf{e}_q \, \mathrm{d}W_t, \qquad (33)$$

where $\tilde{\mathbf{A}}(h) = \mathbf{P}$, the Pascal triangle matrix, and

$$(\tilde{\mathbf{Q}}(h))_{ij} = (\mathbf{B}\mathbf{Q}(h)\mathbf{B}^\mathsf{T})_{ij}$$
$$= \frac{h^i}{i!}\sigma^2 \frac{h^{2q+1-i-j}}{(2q+1-i-j)(q-i)!(q-j)!} \frac{h^j}{j!}$$
$$= \frac{\sigma^2 h^{2q+1}}{(2q+1-i-j)(q-i)!(q-j)!i!j!} \qquad (34)$$

which can be seen by inserting (33) into (6) and simplifying. Furthermore, the observation matrices become $\tilde{\mathbf{H}}_0 = \mathbf{H}_0\mathbf{B}^{-1} = \mathbf{e}_0$ and $\tilde{\mathbf{H}}_1 = \mathbf{H}_1\mathbf{B}^{-1} = h^{-1}\mathbf{e}_1$. Rewriting the filtering equations, we arrive at

$$\mathbf{C}_{t_n}^- = \mathbf{P}\mathbf{C}_{t_{n-1}}\mathbf{P}^\mathsf{T} + \tilde{\mathbf{Q}}(h), \qquad (35)$$

$$\mathbf{K}_n = \mathbf{C}_{t_n}^- \tilde{\mathbf{H}}_1^\mathsf{T} \left[\tilde{\mathbf{H}}_1 \mathbf{C}_{t_n}^- \tilde{\mathbf{H}}_1^\mathsf{T}\right]^{-1} \qquad (36)$$

and

$$\mathbf{m}_{t_n} = (\mathbf{I} - \mathbf{K}_n \tilde{\mathbf{H}}_1)\mathbf{P}\,\mathbf{m}_{t_{n-1}} + \mathbf{K}_n z_n^{(1)}, \qquad (37)$$

$$\mathbf{C}_{t_n} = (\mathbf{I} - \mathbf{K}_n \tilde{\mathbf{H}}_1)\mathbf{P}\left(\mathbf{C}_{t_{n-1}}\mathbf{P}^\mathsf{T} + \tilde{\mathbf{\Phi}}_{12}(h)^\mathsf{T}\right). \qquad (38)$$

Choosing a prior covariance matrix with entries $(\mathbf{C}_{t_{-1}}^-)_{ij} = \sigma^2 h^{2q+1} c_{ij}$, for some $c_{ij} \in \mathbb{R}$ such that $\mathbf{C}_{t_{-1}}^-$ is a valid covariance matrix, it can be shown by induction that all entries of $\mathbf{C}_{t_n}$ for all $n$ have this structural form. As a by-product, $\mathbf{K}_n = h\,(k_{n,0}, 1, k_{n,2}, \ldots, k_{n,q})^\mathsf{T}$ for some $k_{n,i} \in \mathbb{R}$ which follows from (36).

Given these invariants, Eq. (37) has the structure of a multistep method written in Nordsieck form (28). The only difference is the changing weight vector $\mathbf{K}_n$ (37) as compared to the constant weights in (28). Multistep methods with varying weights have been studied in the literature (Crouzeix and Lisbona 1984; Brown et al. 1989). These works are often in the context of variable step sizes $h_n \neq h$, but variable-coefficient methods have also been studied for other purposes, for example cyclic methods (Albrecht 1978). These works have in common that the weights are free variables that are not limited through the choice of model class. As a consequence, determining optimal weights can be algebraically difficult (Hairer et al. 1987, §III.5).

Here, variable step sizes are easily obtained by working with representation (4) instead of (33) and computing (8) according to $h_n$. In contrast to classical methods, the weights $\mathbf{K}_n$ cannot be chosen freely, but are determined through the choice of model (4) and the evolution of the underlying uncertainty $\mathbf{C}_{t_n}$. While Kersting and Hennig (2016) provide some preliminary empirical evidence that these adaptive weights $\mathbf{K}_n$ might actually improve the estimate, more rigorous analysis is required for theoretical guarantees.

In fact, Skeel (and Jackson) (1976, 1977) consider more general *propagation matrices* $\mathbf{S}$ for $\mathbf{x}_{t_n} = \mathbf{S}\mathbf{x}_{t_{n-1}}$ in Eq. (28).

Every model of form (4) implies such a general propagation matrix by identifying $\mathbf{S}_n = (\mathbf{I} - \mathbf{K}_n\mathbf{H}_1)\mathbf{A}(h_n)$. Thus, applying the Kalman filter to LTI SDE models is structurally equivalent to a variable-coefficient multistep method. This motivates the following definition and Algorithm 2 for the probabilistic solution of initial value problems.

**Definition 1** A *probabilistic filtering ODE solver (PFOS)* is the Kalman filter applied to an initial value problem with an underlying Gauss–Markov linear, time-invariant SDE and Gaussian observation likelihood model.

---

**Algorithm 2:** Probabilistic filtering ODE solver

---

1: Define $t_{-1} := t_0$, choose $\mathbf{F}$, $\mathbf{L}$, $\sigma^2$, initialize $\mathbf{m}_{t_{-1}}^-$, $\mathbf{C}_{t_{-1}}^-$ accordingly
2: $\mathbf{K}_{-1} \leftarrow \mathbf{C}_{t_{-1}}^- \mathbf{H}_0^\mathsf{T}[\mathbf{H}_0\mathbf{C}_{t_{-1}}^-\mathbf{H}_0^\mathsf{T}]^{-1}$
3: $\mathbf{m}_{t_{-1}} \leftarrow \mathbf{m}_{t_{-1}}^- + \mathbf{K}_{-1}[y_0 - \mathbf{H}_0\mathbf{m}_{t_{-1}}^-]$
4: $\mathbf{C}_{t_{-1}} \leftarrow (\mathbf{I} - \mathbf{K}_n\mathbf{H}_0)\mathbf{C}_{t_{-1}}^-$
5: **for** n = 0 **to** N **do**
6: $\quad h_n \quad \leftarrow t_n - t_{n-1}$
7: $\quad$ Compute $\mathbf{A}(h_n)$, $\mathbf{Q}(h_n)$
8: $\quad \mathbf{m}_{t_n}^- \quad \leftarrow \mathbf{A}(h_n)\mathbf{m}_{t_{n-1}}$ {Predict}
9: $\quad \mathbf{C}_{t_n}^- \quad \leftarrow \mathbf{A}(h_n)\mathbf{C}_{t_{n-1}}\mathbf{A}(h_n)^\mathsf{T} + \mathbf{Q}(h_n)$
10: $\quad z_n \quad \leftarrow f(t_n, \mathbf{H}_0\mathbf{m}_{t_n}^-)$ {Evaluate}
11: $\quad \lambda_n \quad \leftarrow z_n - \mathbf{H}_1\mathbf{m}_{t_n}^-$ {Update}
12: $\quad \mathbf{K}_n \quad \leftarrow \mathbf{C}_{t_n}^-\mathbf{H}_1^\mathsf{T}[\mathbf{H}_1\mathbf{C}_{t_n}^-\mathbf{H}_1^\mathsf{T}]^{-1}$
13: $\quad \mathbf{m}_{t_n} \quad \leftarrow \mathbf{m}_{t_n}^- + \mathbf{K}_n\lambda_n$
14: $\quad \mathbf{C}_{t_n} \quad \leftarrow (\mathbf{I} - \mathbf{K}_n\mathbf{H}_1)\mathbf{C}_{t_n}^-$
15: **end for**
16: **return** $\{\mathbf{m}_{t_N}, \mathbf{C}_{t_N}, \ n = -1, \ldots, N\}$

---

As was the case in Algorithm 1, the exact form of lines 10–12 depend on the choice of likelihood model (cf. Kersting and Hennig 2016).

We will now study the long-term behavior of the PFOS. In particular, we will ask what is the long-term behavior for the sequence of Kalman gains $(\mathbf{K}_n)_{n=0,\ldots}$ and how this will influence the solution quality. It can be shown that its properties are linked to properties of the *discrete algebraic Riccati equation*, of which the theory has largely been developed (Lancaster and Rodman 1995). Denote by $\gamma : \mathbb{R}^{(q+1)\times(q+1)} \to \mathbb{R}^{(q+1)\times(q+1)}$ the function that maps the covariance matrix $\mathbf{C}_{t_{n-1}}$ of the previous knot $t_{n-1}$ to the covariance matrix $\mathbf{C}_{t_n}$ at the current knot $t_n$ (Eq. (38)). If there exists a (unique) fixed point $\mathbf{C}^*$ of $\gamma$, it is called the *steady state* of model (4). Associated with a fixed point $\mathbf{C}^*$ is also a constant Kalman gain $\mathbf{K}^*$ that is obtained at the (numerical) convergence of $\mathbf{C}^*$.

We will now show that there is a subset of model (4) that converges to a steady state. This subsystem completely determines a constant Kalman gain $\mathbf{K}^*$ at least in the case of the IWP(1) and IWP(2). Thus, just like in the equivalence result for the Runge–Kutta methods in Schober et al. (2014a), the

result of the PFOS is equivalent (in the sense of numerically identical) after an initialization period to a corresponding classical Nordsieck method defined by the weight vector $\mathbf{K}^*$ and we can apply all the known theory of multistep methods to the mean of the PFOS.

**Proposition 1** *The PFOS arising from the once integrated Wiener process IWP(1) is equivalent in its predictive posterior mean to the $P(EC)^1$ implementation of the trapezoidal rule.*

*Proof* The trapezoidal rule, written as in Eq. (26), is

$$y_{t_n} = y_{t_n} + \frac{h}{2}(f_{t_{n-1}} + f_{t_n}). \tag{39}$$

We will show that $(\mathbf{m}_{t_n})_0 = (\mathbf{m}_{t_{n-1}})_0 + h/2[(\mathbf{m}_{t_{n-1}})_1 + (\mathbf{m}_{t_n})_1)]$ for all $n$ by induction. Let $\mathbf{m}_{t_{-1}}^- = \mathbf{0}$ and $\mathbf{C}_{t_{-1}}^- \in \mathbb{R}^{2\times2}$ be an arbitrary covariance matrix. Applying the first three lines of Algorithm 2 algebraically, the predicted values are

$$\mathbf{m}_{t_{-1}} = \begin{pmatrix} y_0 \\ m_{t_0,1}^- \end{pmatrix}, \qquad \mathbf{C}_{t_{-1}} = \begin{pmatrix} 0 & 0 \\ 0 & c_{t_0,11}^- \end{pmatrix} \tag{40}$$

for some $m_{t_0,1}^-$, $c_{t_0,11}^-$. Continuing in this fashion yields $z_0 := f(t_0, y_0)$ and $\mathbf{m}_{t_0} = (y_0, z_0)^\mathsf{T}$, $\mathbf{C}_{t_0} = \mathbf{0}$. Using (20) and (21) to compute the predictions at $t_1$, we arrive at

$$\mathbf{m}_{t_1}^- = \begin{pmatrix} y_0 + hz_0 \\ z_0 \end{pmatrix}, \qquad \mathbf{C}_{t_1}^- = \mathbf{Q}(h) \tag{41}$$

and we see that $\mathbf{H}_0\mathbf{m}_{t_0+h}^- = y_0 + hz_0 = (\mathbf{P}(y_0, hz_0)^\mathsf{T})_0$. Completing the Kalman step by applying Eqs. (22)–(25) yields

$$\mathbf{m}_{t_1} = \begin{pmatrix} y_0 + \frac{h}{2}[z_0 + z_1] \\ z_1 \end{pmatrix}, \qquad \mathbf{C}_{t_1} = \sigma^2\begin{pmatrix} \frac{h^3}{12} & 0 \\ 0 & 0 \end{pmatrix}, \tag{42}$$

where $z_1 := f(t_1, y_0 + hz_0)$. Comparing with (30), we see that $z_1$ is of the desired form, which completes the start of the induction. Finally, we observe that the second column of $\mathbf{C}_{t_1} = \mathbf{0} = \mathbf{C}_{t_0}$, i.e., this will be invariant and, as a consequence, the second column of $\mathbf{C}_{t_n}^-$ is simply the second column of $\mathbf{Q}(h)$, and the induction is completed. $\square$

The following Theorem 1 for the IWP(2) requires a bit more algebra, but is based on the same principle.

**Theorem 1** *The predictive posterior mean of the IWP(2) with fixed step size $h$ is a third-order Nordsieck method, when the predictive distribution has reached the steady state.*

*Proof* The proof proceeds in two steps. First, we show that the update equations induce a specific form for the covariance matrix $\mathbf{C}_{t_n}$. Then, we will analyze individual entries.

We proof by induction that $\mathbf{C}_{t_n}$ is of the form

$$\mathbf{C}_{t_n} = \sigma^2 h^5 \begin{pmatrix} c_{t_n,00} & 0 & c_{t_n,02} \\ 0 & 0 & 0 \\ c_{t_n,02} & 0 & c_{t_n,22} \end{pmatrix} \tag{43}$$

with coefficients $c_{t_n,ij}$ such that $\mathbf{C}_{t_n}$ is a valid covariance matrix. The base case is achieved after the first derivative observation $f(t_0, y_0)$ at $t_0$ which can be checked by algebraic computation. The inductive step can be verified by assuming form 43 for $t_{n-1}$ and compute one step ahead using Eqs. (36) and (38) similar to the base case. Next, for the individual entries we find

$$c_{t_{n+1},00} = \sigma^2 h^5 \frac{3840c_{00}c_{22} + 320c_{00} - 3840c_{02}^2 + 110c_{02} + 32c_{22} + 1}{320(12c_{22} + 1)}$$

$$c_{t_{n+1},02} = \sigma^2 h^5 \frac{-(48c_{02} + 24c_{22} + 1)}{96(12c_{22} + 1)} = (\mathbf{C}_{t_{n+1}})_{20}$$

$$c_{t_{n+1},22} = \sigma^2 h^5 \frac{16c_{22} + 1}{16(12c_{22} + 1)}$$

$$c_{t_{n+1},ij} = 0, \quad i, j = 0, 1, 2, i \vee j = 1 \tag{44}$$

where we put $c_{ij} := c_{t_n,ij}$ on the respective right-hand sides of Eq. 44 for brevity. We will now consider the behavior of the coefficients $c_{ij}$. Consider the dynamical system $\bar{\gamma}_{22}(c) = (16c + 1)[16(12c + 1)]^{-1}$ which maps the coefficient of the last entry in $\mathbf{C}_{t_n}$ to the next. The range and image of $\bar{\gamma}_{22}$ are the nonnegative reals, since variances cannot be negative. On this domain, $\bar{\gamma}_{22}$ has a continuous and bounded derivative $|\bar{\gamma}'_{22}| \leq \frac{1}{4}$. In particular, $\bar{\gamma}_{22}$ is a contraction with Lipschitz constant $\frac{1}{4}$. Thus, the entries converge to the fixpoint $c_{22}^* = \frac{\sqrt{3}}{24}$ (which can be found with some simple algebra). Similarly, one can either insert $c_{22}^*$ into the respective form of $\bar{\gamma}_{02}$ or one considers the two-dimensional mapping of both entries. In both cases, a similar argument guarantees the convergence to a fixpoint, which is found to be $c_{02}^* = -\frac{\sqrt{3}}{144}$. Inserting these into Eq. (36), we find that $\mathbf{K}_n = \mathbf{K}^* = (\frac{3+\sqrt{3}}{12}, 1, \frac{3-\sqrt{3}}{2})^\top$ is the static probabilistic Nordsieck method of the IWP(2) filter. Inserting these weights into (Skeel 1979, Theorem 4.2) yields the result. □

Although Theorem 1 is only valid when the system has reached its steady state, we find that the convergence (visualized in Fig. 3) is rapid in practice. In the extreme case of $q = 1$ (not shown), in fact it is instantaneous, and Proposition 1 is valid from the second step onwards. This limitation could also be circumvented in practice by initializing $\mathbf{C}_{t_{-1}}$ at steady-state coefficients, but this possibility is not required to achieve high-order convergence on the benchmark problems we considered.

Figure 3 shows the situation for a constant value of the diffusion amplitude $\sigma^2$. In Sect. 4, we will discuss error estimation and step size adaptation. This process leads to a continuous adaptation of this variable, which in turn means
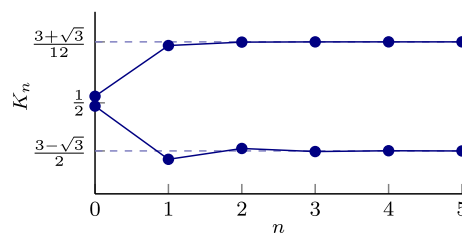


**Fig. 3** The weights $(K_n)_0$ and $(K_n)_2$ for $n = 0, \ldots, 5$

that the convergence shown in the figure continues throughout the run of the algorithm. So the practical algorithm presented here and empirically evaluated in Sect. 5 is not formally identical to Nordsieck methods, merely conceptually closely related.

Inspecting the weights of the IWP(2), we find that this method has not been considered previously in the literature, and, in particular, cannot be related to any of the typical formulas, such as Adams–Moulton or backward differentiation formulas. This is not surprising, since the result of this method has been constructed to be twice continuously differentiable, whereas there is no such guarantee for the solution provided by the typical methods. In fact, the probabilistic Nordsieck method is much closer related to spline-based multistep methods such as Loscalzo and Talbot (1967), Loscalzo (1969), Byrne and Chi (1972) and Andria et al. (1973) since Gaussian process regression models have a one-to-one correspondence to spline smoothing in a reproducing kernel Hilbert space of appropriate choice (Kimeldorf and Wahba 1970; Wahba 1990). This also justifies the application of a full-support distribution, even though it is known that the solution will remain in a compact set. In the former case, the interpretation is one of average-case error, whereas in the latter, the bound corresponds to the worst-case error (Paskov 1993).

The forms of $C_{t_n}$ found in Eqs. (42) and (44) also show that the standard deviation $\text{std}[Y_{t_n}] = (\mathbf{C}_{t_n})_{00}^{1/2}$ can be meaningfully, if weakly, interpreted as an approximation to the *expected error* $|y_{t_n} - y(t_n)|$ of the numerical solution in the following local, asymptotic sense: From our analysis of the IWP($q$), $q \in \{1, 2\}$, we have $|y_{t_n} - y(t_n)| \leq Ch^{q+1}$, whereas $(\mathbf{C}_{t_n})_{00}^{1/2} \in \mathcal{O}(\sigma h^{q+1/2})$. Estimating the intensity $\sigma$ of the stochastic process amounts to estimating the unknown constant $C$.

Figure 4 displays the work-precision diagram for the IWP(1) and IWP(2) applied to the examplary problem of Sect. 2.4. The plot shows a good agreement between the theoretical rate and the observed rate of convergence.

We conclude this section by considering some implications of the probabilistic interpretation in contrast to other classical multistep methods.

Keeping all hyperparameters (order $q$, prior diffusion intensity $\sigma^2$ and step size $h$) fixed, the gain $\mathbf{K}_n$ is completely
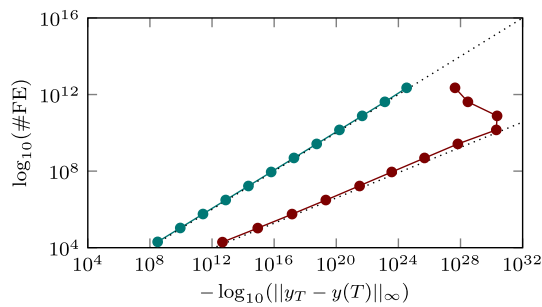
**Fig. 4** Work-precision diagram for the IWP(1) (green) and IWP(2) (red) applied to the logistic growth problem from Sect. 2.4. Plotted are the logarithms of the number of function evaluations (#FE) against the logarithmic error at the end of the integration domain. Dotted lines mark ideal convergence rates of orders two and three, respectively. (Color figure online)

determined, and, as a consequence, we could have chosen to fully solve implicit Eq. (29) for the generation of $z_n$. Solving (29) up to numerical precision can be interpreted as learning the true value of the model (4) at $t_n$ which gives another justification for using $R_n^2 = R^2 = 0$. Since the $P(EC)^\infty$ and the $P(EC)^M$ have the same order for all $M$ (Deuflhard and Bornemann 2002), this argument can be extended to the case of the $PEC^1$ implementation which gives the most natural connection to the Kalman filter.

In fact, a $P(EC)^M$, $M > 1$, implementation would collect and put aside the values $z_n^{(1)}, \ldots, z_n^{(M-1)}$, which seems unintuitive from an inference point of view, where it is natural to assume that more data should yield a better approximation. A natural question would be whether this is a case of diminishing returns of approximation quality for computational power, but this is beyond the scope of this paper.

One current limitation of the PFOS is its fixed integration order $q$ over the whole integration domain $\mathbb{T}$. The reason for this limitation is that it is conceptually not straight forward to connect spline models of different orders at knots $t_n$. However, the ability to adapt the integration order during runtime has been key in improving the efficiency of modern solvers (Byrne and Hindmarsh 1975). Furthermore, the method corresponding to the IWP(2) model has a rather small region of stability which is depicted in Fig. 5, specially in comparison with backward differentiation formulas (BDFs) (Deuflhard and Bornemann 2002). This makes the method impractical for stiff equations.

It is natural to ask what happens in the case of the IWP($q$), $q > 2$. Using techniques from the analysis of Kalman filters, one can show that these models also contain a stable subsystem and that the weights $\mathbf{K}_n$ will converge to a fixed point $\mathbf{K}^*$, even for nonzero, but constant, $R^2$. However, it remains unclear whether they will be practical. In particular, these methods might even be unstable for most spline models (Loscalzo and Talbot 1967). We have tested the IWP($q$), $q \in \{1, \ldots, 4\}$, empirically on the Hull et al. benchmark (see
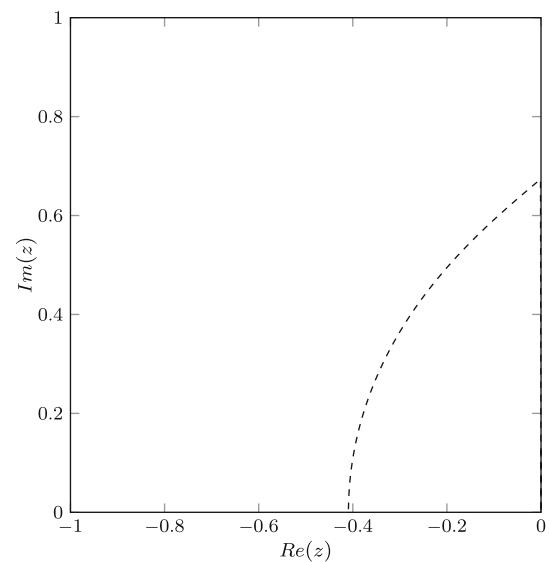


**Fig. 5** Partial stability domain of the probabilistic Nordsieck method using the IWP(2) in the negative real, positive imaginary quadrant. The method converges for step sizes $h$ on linear problems $y' = \lambda y$, if $h\lambda := z \in \mathbb{C}$ lies in the region of stability in the lower right corner. See Deuflhard and Bornemann (2002) for details

Sect. 5) and have observed that these converge in practice on these nonstiff problems.

### 3.2 Initialization via Runge–Kutta methods

Thus far, we have provided the analysis of the long-term behavior of the algorithm, when several Kalman filter steps have been computed and the steady state is reached. Crucially, a necessary condition for this analysis is that enough updates have been performed such that the observable space spans the entire state space, which is $q + 1$ updates in the case for the IWP($q$).

Thus, the question remains how to initialize the filter. Schober et al. (2014a) have shown that there are Runge–Kutta steps that coincide with the maximum a posteriori (MAP) of the IWP($q$) for $q \leq 3$. This requires $q + 1$ updates using a diffuse prior with $\mathbf{C}_{t_{-1}} = \lim_{\mathcal{H} \to \infty} \mathbf{Q}(\mathcal{H})$. In practice, one takes a Runge–Kutta step with the corresponding formula and plugs the resulting values into the analytic expressions for $\mathbf{m}_{t_1}$ and $\mathbf{C}_{t_1}$ at $t_1$. Additionally to the cases presented in Schober et al. (2014a), we can report a match between a four-step Runge–Kutta formula of order four and the IWP(4). This match is obtained for the evaluation knots $t_0 + c_i h$ with the vector $\mathbf{c} = (0, 1/3, 1/2, 1)^\top$. Details and exact expressions are given in Appendix B. This approach is structurally similar to an algorithm given by Gear (1980) for the case of classical Runge–Kutta and Nordsieck methods.

However, we want to stress that the analysis by Schober et al. (2014a) is done under exactly the same model and with the same assumptions that have been presented here in dif-

ferent notation. Therefore, the initialization does not require a separate model and our requirement of a globally defined solver still holds.

Finally, it should be pointed out that this is only one feasible initialization. In cases where automatic differentiation (Griewank and Walther 2008) is available, this can be used to initialize the Nordsieck vector up to numerical precision and set $\mathbf{C}_{t_{-1}}$ to $\mathbf{0}$. Nordsieck originally proposed (Nordsieck 1962) start with an initial vector $\mathbf{m}_{t_{-1}} = 0$, followed by $q+1$ steps with positive and $q+1$ with negative direction (that is, integrating backwards to the start). One interpretation is that the method uses $\mathbf{m}_{t_{-1}} = \mathbb{E}[X_{t_{-1}} \mid \tilde{z}_{-1}, \ldots, \tilde{z}_q]$, with tentative $\tilde{z}_i$ computed out of this process.

## 4 Error estimation and hyperparameter adaptation

While the general algorithm described in Sect. 3.1 can be applied to any IVP at this stage, a modern ODE solver also requires the ability to automatically select sensible values for its hyperparameters. The filter has three remaining parameters to choose: the dimensionality $q$ of the state space, the diffusion amplitude $\sigma^2$ and the step size $h$.

To obtain a globally consistent probability distribution, we fix $q = 2$ throughout the integration to test the third-order method presented in Sect. 3.1. For the remaining two parameters, we first note that estimating $\sigma^2$ will lend itself naturally to choose the step size. To see this, one needs to make the connection to classical ODE solvers and the interpretation of the state-space model. In classical ODE solvers, $h_n$ is determined based on local error analysis, that is, $h_n$ is a function of the error $e_{t_n}$ introduced from step $t_{n-1}$ to step $t_n$. Then, $h_n$ is computed as a function of the allowed tolerance and the expected error which is assumed to evolve similarly to the current error.

As is common in solving IVPs, we base error estimation on *local errors*. Assume that the predicted solution $\mathbf{m}_{t_{n-1}}$ at time $t_{n-1}$ is error free, that is, $\mathbf{C}_{t_{n-1}} = \mathbf{0}$. Then, by Eqs. (21) and (22), we have

$$p(\lambda_n \mid \sigma^2) = \mathcal{N}\left(\lambda_n; z_n - \mathbf{H_1}\mathbf{m}_{t_n}^-, \mathbf{H_1}\sigma^2\bar{\mathbf{Q}}(h)\mathbf{H_1^\intercal}\right). \quad (45)$$

One way to find the optimal $\sigma^2$ is to construct the maximum likelihood estimator from Eq. (45) which is given by

$$\hat{\sigma}^2 = \left(z_n - \mathbf{H_1}\mathbf{m}_{t_{n-1}}^-\right)^\intercal \left(\mathbf{H_1}\bar{\mathbf{Q}}(h)\mathbf{H_1^\intercal}\right)^{-1} \left(z_n - \mathbf{H_1}\mathbf{m}_{t_{n-1}}^-\right)$$
$$= \frac{\left(z_n - \mathbf{H_1}\mathbf{m}_{t_{n-1}}^-\right)^2}{\mathbf{H_1}\bar{\mathbf{Q}}(h)\mathbf{H_1^\intercal}}. \quad (46)$$

For the last equation, we used the fact that all the involved quantities are scalars.

To allow for a greater flexibility of the model, we allow amplitude $\sigma^2$ to vary for different steps $\sigma^2_{t_n}$. Note that the mean values are then no longer independent of $\sigma^2$, because the factor no longer cancels out in the computation of $K_n$ in Eq. (24). However, this situation is indeed intended: If there was more diffusion in $[t_{n-1}, t_n]$, we want a stronger update to the mean solution as the observed value is more informative. Additionally, Eq. (22) is independent of $\sigma^2_{t_n}$ or any other covariance information $\mathbf{P}_{t_n}^-$, $\mathbf{Q}(h)$. Therefore, we can apply Eq. (22) before (21), update $\sigma^2_{t_n}$ and then continue to compute the rest of the Kalman step. This idea is similar in spirit to (Jazwinski 1970, §11), but follows the general idea of error estimation in numerical ODE solvers, where local error information is available only.

At this point, the inference interpretation of numerical computation comes to bear: once the initial modeling decision—modeling a deterministic object with a probability measure to describe the uncertainty over the solution—is accepted, everything else follows naturally from the probabilistic description. Most importantly, there are no neglected higher-order terms, as they are all incorporated in the diffusion assumption.

This kind of lightweight error estimation is a key ingredient to probabilistic numerical methods: one goal of a probabilistic model is improved decisions under *uncertainty*. This uncertainty is necessarily a crude approximation, since a more accurate error estimator could be used to improve the overall solution quality. However, the reduction in computational efforts up to a tolerated error is exactly what modern numerical solvers try to achieve.

This error estimate can now be used in the conventional way of adapting the step size which we will restate here to give a complete description of the inference algorithm (see also Byrne and Hindmarsh 1975). Given an error weighting vector $\mathbf{w}$, the algorithm computes the weighted expected error

$$(\mathbf{D}_{t_n})_i = \left(\mathbf{H_1}\sigma^2_{t_n}\bar{\mathbf{Q}}(h_n)\mathbf{H_1^\intercal}\right)_i^{1/2} \mathbf{w}_i, \quad (47)$$

where $\bar{\mathbf{Q}}(h_n) = [\sigma^2_{t_n}]^{-1}\mathbf{Q}(h_n)$ is the normalized diffusion matrix and checks whether some error tolerance with parameter $\epsilon$ is met

$$\mathbf{D}_{t_n} \leq \bar{\epsilon} := \epsilon \frac{h_n}{S} \quad (48)$$

where $h_n$ is the step length and $S$ can be either chosen to be $S = 1$ (error per unit step) or $S = h_n$ (error per step). If Eq. (48) holds, the step is accepted and integration continues. Otherwise, the step is rejected as too inaccurate and the step is repeated. In both cases, a new step length is computed

which will likely satisfy Eq. (48) on the next step attempt. The new step size is computed as

$$h_{n+1} = \rho \left( \frac{\bar{\epsilon}}{\mathbf{D}_{t_n}} \right)^{\frac{1}{q+1}} \qquad (49)$$

where $\rho \in (0, 1)$, $\rho \approx 1$ is a safety factor. Additionally, we also follow best practices (Hairer et al. 1987) limit the rate of change $\eta_{min} < h_{n+1}/h_n < \eta_{max}$. In our code, we set $\rho := 0.95$, $\eta_{min} := 0.1$ and $\eta_{max} := 5$.

## 4.1 Global versus local error estimation

The results presented in preceding sections pertain to the estimation of *local* extrapolation errors. It is a well-known aspect of ODE solvers (Hairer et al. 1987, §III.5) that the *global* error can be exponentially larger than the local error. More precisely, to scale the stochastic process such that the variance of the resulting posterior measure relates to the square *global* error, the intensity $\sigma_n^2$ of the stochastic process must be multiplied by a factor (Hairer et al. 1987, Thm III.5.8) $\exp(L^*(T - t_0))$, where $L^*$ is a constant depending on the problem. Although related, $L^*$ is not the same as the local Lipschitz constant $L$ and harder to estimate in practice (more details in Hairer et al. 1987, §III.5). We stress that this issue does not invalidate the probabilistic interpretation of the posterior measure as such. It is just that the scale of the posterior has to be estimated differently if the posterior is supposed to capture global error instead of local error. In practice, the global error estimate resulting from this re-scaling is often very conservative.

## 5 Experiments

To evaluate the model, we provide two sets of experiments. First, we qualitatively examine the uncertainty quantification by visualizing the posterior distribution of two example problems. We also compare our proposed observation assumption against the model described by Chkrebtii et al. (2016). Second, we more rigorously evaluate the solver on a benchmark and compare it to existing non-probabilistic codes. Our goal in this work is to construct an algorithm that produces meaningful probabilistic output at a computational cost that is comparable to that of classic, non-probabilistic solvers. The experiments will show that this is indeed possible. Other probabilistic methods, in particular that of Chkrebtii et al. (2016), aim for a more expressive, non-Gaussian posterior. In exchange, the computational cost of these methods is at least a large multiple of that of the method proposed here, or even polynomially larger. These methods and ours differ in their intended use cases: More elaborate but expensive

posteriors are valuable for tasks in which uncertainty quantification is a central goal, while our solver aims to provide a meaningful posterior as additional functionality in settings where fast estimates are the principal objective.

### 5.1 Uncertainty quantification

We apply the probabilistic filtering ODE solver on two problems with attracting orbits: the Brusselator (Lefever and Nicolis 1971) and van der Pol's equation (1926). The filter is applied twice on each problem, once with a fixed step size and once with the adaptive step size algorithm described in Sect. 4. To get a visually interesting plot, the fixed step size and the tolerance threshold were chosen as large as possible without causing instability. Both cases are modeled with a local diffusion parameter $\sigma_n^2$ which is estimated using the maximum likelihood estimator of Sect. 4. In the following plots, the samples use the scale $\sigma_n^2$ arising from the local error estimate. Because these systems are attractive, the global error correction mentioned in Sect. 4.1 would lead to significantly more conservative uncertainty.

The Brusselator is the idealized and simplified model of an autocatalytic multi-molecular chemical reaction (Lefever and Nicolis 1971). The rate equations for the oscillating reactants are

$$\begin{aligned} y_1' &= A + y_1^2 y_2 - (B + 1)y_1 \\ y_2' &= B y_1 - y_1^2 y_2, \end{aligned} \qquad (50)$$

where $A$ and $B$ are positive constants describing the initial concentrations of two reactants. Following Hairer et al. (1987), we set $A = 1$, $B = 3$ and $(y_1(0), y_2(0))^\top = (1.5, 3)^\top$. The integration domain $\mathbb{T} = [0, 10]$ has been chosen such that the solution completes one cycle on the attractor after an initial convergence phase.

The results in Fig. 6 demonstrate the effectiveness of the error estimator. This problem also demonstrates the quality and utility of the step size adaptation algorithm, since on the majority of the solution trajectory the algorithm is not limited by stability constrains. In the right plot, it can be seen how an increase in step size $h_{n+1} > h_n$ can also lead to a reduction in posterior uncertainty. This is a consequence of $\sigma_{t_{n+1}}^2 / \sigma_{t_n}^2 < 1$.

Figure 10 in Appendix also displays the solution as a function of time.

Van der Pol's equation (1926) describes an oscillation with a nonlinear damping factor $\alpha$

$$\begin{aligned} 0 &= y'' + \alpha y' + y \\ \alpha &= \mu(y^2 - 1) \end{aligned} \qquad (51)$$

with a positive constant $\mu > 0$. Originally, this model has been used to describe vacuum tube circuits. The limit cycle
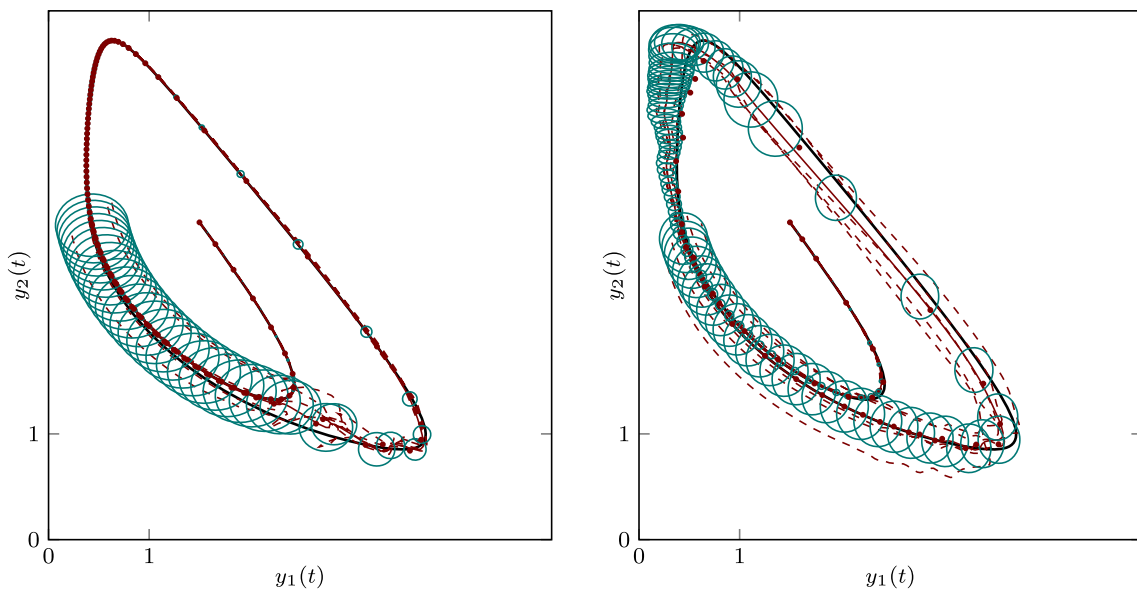
**Fig. 6** Numerical solution of the Brusselator (50) using the probabilistic filtering ODE solver. The plots show the solution computed by `ode45` using `RelTol = AbsTol = 1 × 10^{-13}` (black, background), the posterior mean (red, thick line), iso-contourlines of twice the posterior standard deviation at a subsample of the knots (green) and samples from the posterior distribution (red, dashed lines). Left: Using a fixed step size of $h = h_n = 0.0834$. The computation requires 120 steps. Right: Using the adaptive step size selection with error weighting $w_i(y) = (\tau y_i + \tau)^{-1}, \tau = 0.1$. The computation requires 43 steps. See Hairer et al. (1987, §1.6) for details. (Color figure online)
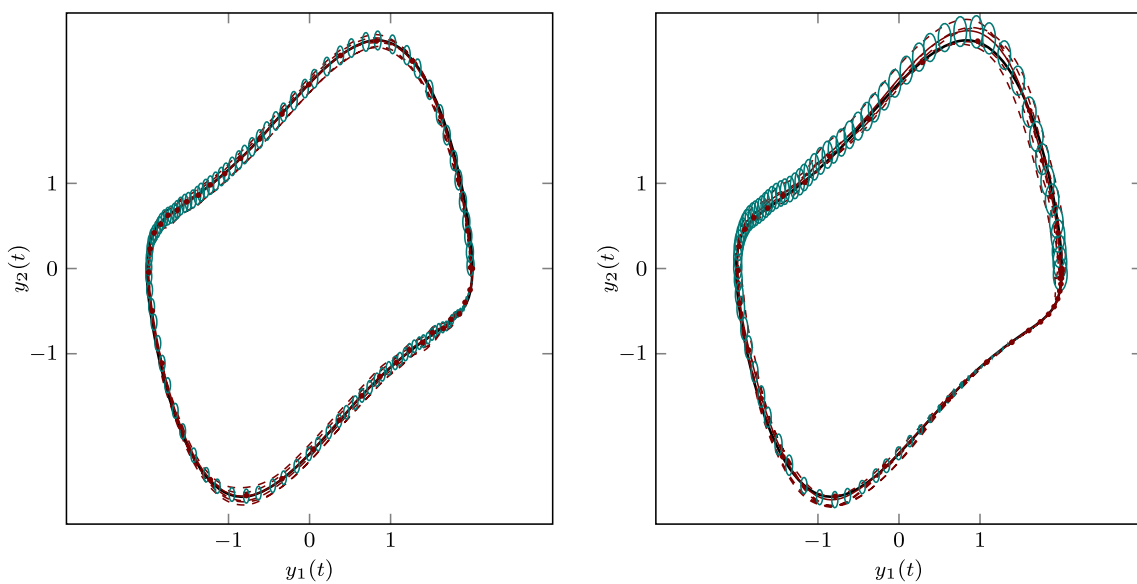


**Fig. 7** Numerical solution of van der Pol's equation (51) using the probabilistic filtering ODE solver, integrated over one limit cycle period $\mathbb{T} = [0, T]$ with initial value $y(0) = (A, 0)^{\mathsf{T}}$, where $T \approx 6.6633$ and $A \approx 2.0086$. The plots show the solution computed by `ode45` using `RelTol = AbsTol = 1 × 10^{-13}` (black, background), the posterior mean (red, thick line), iso-contourlines of twice the posterior standard deviation at a subsample of the knots (green) and samples from the posterior distribution (red, dashed lines). Left: Using a fixed step size of $h = h_n = 0.1667$. The computation requires 40 steps. Right: Using the adaptive step size selection with error weighting $w_i(y) = (\tau y_i + \tau)^{-1}, \tau = 0.1$. The computation requires 41 steps. See Hairer et al. (1987, §1.6) for details. (Color figure online)

alternates between a nonstiff phase of rapid change and a stiff phase of slow decay. The larger $\mu$, the more pronounced both effects are. In our example, we set $\mu = 1$ and integrate over one period with the initial value on the graph of the limit cycle. Exact values can be found in Hairer et al. (1987, §I.16).

Figure 7 plots the filter results. Figure 11 displays the solution as a function of time. In the case of van der Pol's
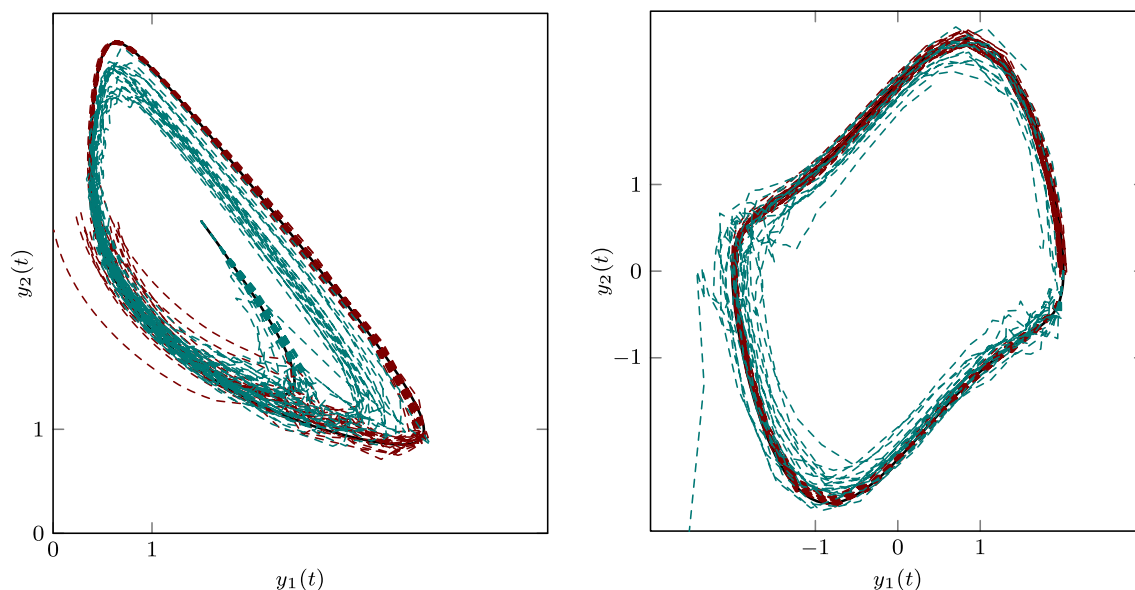
**Fig. 8** Comparison of two different evaluation strategies on problems (50) and (51). Red: samples from the posterior as in Figs. 6 and 7. Green: Similar, but evaluating at $z_n = f(t_n, (u_{t_n})_0)$, $u_{t_n} \sim \mathcal{N}(\mathbf{m}_{t_n}^-, \mathbf{C}_{t_n}^-)$. This is similar to Chkrebtii et al. (2016). (Color figure online)

equation, the benefit of step size adaptation is essentially nil, because conservative adaptation—in particular from a cautious starting step size—consumes the gains on the nonstiff parts. However, the example demonstrates the capability to learn an anisotropic diffusion model for individual components.

Finally, we compare two different strategies of quantifying the uncertainty. To this end, we compare our proposed model to the observation model proposed by Chkrebtii et al. (2016, §3.1). In this case, we set $z_n = f(t_n, (u_{t_n})_0)$, $u_{t_n} \sim \mathcal{N}(\mathbf{m}_{t_n}^-, \mathbf{C}_{t_n}^-)$. Figure 8 shows samples of the posterior distribution, computed with two different evaluation schemes. This scheme is not exactly the same as the one proposed by Chkrebtii et al.—their algorithm actually has cubic complexity in the number of $f$-evaluations; thus, it is limited to a relatively small number of evaluation steps. But our version captures the principal difference between their algorithm and the simpler filter proposed here: Their algorithm draws separate samples involving independent evaluations of $f$ at perturbed locations, while ours draws samples from a single posterior constructed from one single set of $f$-evaluations. As expected, the model of Chkrebtii et al. provides a richer output structure, for example, by identifying divergent solutions (right subplot) if the solver leaves the region of attraction. However, to obtain individual samples, the entire algorithm has to run repeatedly, so the cost of producing $S$ samples is $S$ times that of our algorithm, which produces all its samples in one run, without requiring additional evaluations of $f$.

## 5.2 Benchmark evaluation

As is the case with many modern solvers, the theoretical guarantees do not extend to the full implementation with error estimation and step size control. Therefore, an empirical assessment is necessary to compare against trusted implementations. We compare the proposed Kalman filter to a representative set of standard algorithms on the DETEST benchmark set (Hull et al. 1972). While other standardized tests have been proposed (Crane and Fox 1969; Krogh 1973), DETEST has repeatedly been described as representative (Shampine et al. 1976; Deuflhard 1983). By choosing the same comparison criteria across all test problems and tested implementations, the benchmark provides the necessary data to make predictions on the behavior on a large class of problems.

Two different dimensions of performance are considered in Hull et al. (1972): the *computational cost* and the *solution quality*. Computational cost is reported in execution time (in seconds) and number of function evaluations (abbreviated as #FE). Although the former is more relevant in practice, we only report the latter here as the codes in Hull et al. (1972) are outdated and our proof-of-concept code is not yet optimized for speed. Nevertheless, since the execution times are proportional to the #FE, this provides a reliable estimator of computational efficiency. DETEST only considers methods with automatic step size adaptation and thus measures the solution quality by comparing the local error with the requested tolerance $\epsilon$. A code is considered to produce high-quality solutions if the results are within the requested tolerance, but are also not of excessive unrequested higher accuracy.

**Table 2** Summary of DETEST results

| Method | Total fcn. evals. | Avg. % deceived | Max. error |
|---|---|---|---|
| $\epsilon = 10^{-3}$ | | | |
| Extrapolation | 16553 | 2.0 | 7.8 |
| Adams (Krogh) | 5394 | 1.1 | 5.3 |
| Adams (Gear) | 9498 | 0.9 | 1.5 |
| RK (4th, Kutta) | 8363 | 5.1 | 25.9 |
| RK (6th, Butcher) | 11105 | 5.1 | 1788.1 |
| RK (8th, Shanks) | 12355 | 6.3 | 1120.6 |
| RK (3th, Shampine) | 15085 | 5.9 | 2.4 |
| RK (5th, Shampine) | 5785 | 11.2 | 9.5 |
| Adams (Shampine) | 5692 | 6.5 | 7.7 |
| PNM | 19091 | 0.2 | 1.5 |
| $\epsilon = 10^{-6}$ | | | |
| Extrapolation | 26704 | 0.1 | 2.3 |
| Adams (Krogh) | 11353 | 1.4 | 7.3 |
| Adams (Gear) | 18155 | 0.8 | 2.6 |
| RK (4th, Kutta) | 30763 | 1.8 | 29.1 |
| RK (6th, Butcher) | 23540 | 1.6 | 142.5 |
| RK (8th, Shanks) | 20493 | 4.2 | 4.7 |
| RK (3th, Shampine) | 430975 | 0.0 | 1.9 |
| RK (5th, Shampine) | 19879 | 0.0 | 1.1 |
| Adams (Shampine) | 10777 | 3.6 | 6.3 |
| PNM | 405469 | 0.0 | 1.4 |
| $\epsilon = 10^{-9}$ | | | |
| Extrapolation | 43054 | 0.0 | 0.6 |
| Adams (Krogh) | 18984 | 0.5 | 4.0 |
| Adams (Gear) | 38439 | 2.3 | 2.7 |
| RK (4th, Kutta) | 146262 | 0.3 | 2.9 |
| RK (6th, Butcher) | 58634 | 0.9 | 443.4 |
| RK (8th, Shanks) | 39663 | 2.1 | 20.9 |
| RK (3th, Shampine) | 13587187 | 3.1 | 689.0 |
| RK (5th, Shampine) | 103345 | 0.1 | 2.4 |
| Adams (Shampine) | 18274 | 2.2 | 11.5 |
| PNM | 12731730 | 4.5 | 1938.0 |

Therefore, errors are reported per unit step. Reported are the maximum error $\max\{\xi_n [h_n \epsilon]^{-1} \mid n = 1, \ldots, N\}$ per unit step and the percentage of deceived steps $|\{\xi_n \mid \xi_n > h_n \epsilon, n = 1, \ldots, N\}|/N$, where the local errors $\xi_n$ are defined as $||y_{t_n} - y(t_n; y(t_{n-1}) = y_{t_{n-1}})||_\infty$ and $y(t_n; y(t_{n-1}) = y_{t_{n-1}})$ defines the IVP $y' = f(t, y)$, $y(t_{n-1}) = y_{t_{n-1}}$, $t \in [t_{n-1}, t_n]$.

Here, we report the results from the proposed solver originating from the IWP(2) model as well as the results from the original Hull et al. paper 1972. We have not been able to obtain a copy of the codes used in Hull et al. and only report their numbers for sake of completeness. We also ran the tests on the solvers provided in MATLAB. Table 2 lists the sum-

mary results for all methods and all tolerances. For detailed results on individual problems, see Figs. 12, 13 and 14 in Appendix section. For a complete and detailed description of the benchmark, we refer to Hull et al. (1972). Our implementation is publicly available.[2]

In addition to the benchmark results, we analyze the error estimation model from a probabilistic perspective. Figure 9 plots the cumulative distribution function (CDF) of the local error $\xi_n$, as defined above, divided by the estimated local error $(Q(t_n))_{00}^{1/2} = (\sigma_n^2 \bar{Q}(h_n))_{00}^{1/2}$ for each set of five tasks (different blue colored lines) of each of the five problem classes (figures from left to right). Under the algorithm's internal model, the error is assumed to be Gaussian distributed:

$$P(y_{t_n} \mid \hat{y}_{t_n}) = \mathcal{N}\left(y_{t_n}; \hat{y}_{t_n}, (Q(h_n))_{00}\right), \tag{52}$$

Hence, if that model were a perfect fit, the scaled absolute error plotted in this figure would be $\chi$-distributed:

$$P\left(|y_{t_n} - \hat{y}t_n|(Q(h_n))_{00}^{-1/2}\right) = \chi(1). \tag{53}$$

The comparison with the CDF of $\chi(1)$ shows that the true local error has weaker tails than the predicted $\chi$-distribution.

So, as expected, the error estimator is typically a conservative one.

While the probabilistic method does not achieve the same high performance as modern higher-order codes, the performance matches the results of a production Runge–Kutta code of the same order. This is of particular interest since applications in the low-accuracy regime could benefit the most from accurate error indicators (Gear 1981).

## 6 Conclusions

We proposed a probabilistic inference model for the numerical solution of ODEs and showed the connections with established methods. In particular, we showed how probabilistic inference in Gauss–Markov systems given by a linear time-invariant stochastic differential equations leads to Nordsieck-type methods. The maximum a posteriori estimate of the once integrated Wiener process IWP(1) is equivalent to the trapezoidal rule. The twice integrated Wiener process IWP(2) is equivalent to a third-order Nordsieck-type method which can be thought of as a spline-based multistep method. We demonstrated the practicality of this probabilistic IVP solver by comparing against other state-of-the-art implementations.

The probabilistic formulation has already proven to be beneficial in larger chains of computations involving boundary value problems (Schober et al. 2014b; Hauberg et al.
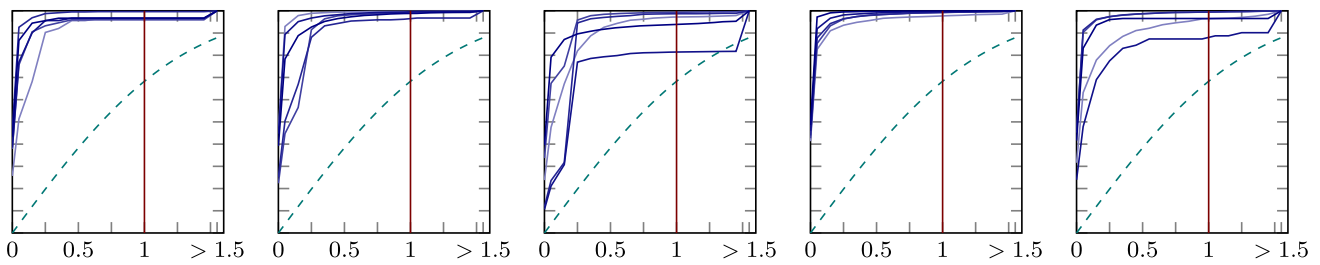
---

[2] https://pn.is.tuebingen.mpg.de/code/pfos.

**Fig. 9** Empirical cumulative distribution function (CDF) of true local errors $\xi_n$ divided by the estimated local errors $(Q(t_n))_{00}^{1/2}$. Ticks on the $y$-axis are spaced at 0.1 intervals from 0 to 1. Values less than 1 (red line) are *over-estimated* leading to a conservative step size adaptation. Green dashed line shows the CDF of the $\chi(1)$-distribution which implies that the empirical distribution has weaker tails. See text for more details. (Color figure online)

2015). While the method presented in this paper is restricted to IVPs, there has also been work on extending the formalism of splines to boundary value problems (Mazzia et al. 2006, 2009). We expect that similar classical guarantees should be transferable to probabilistic boundary value problem solvers as well. Conversely, the probabilistic treatment of the IVP may be beneficial in bigger pipelines as well (cf. Chkrebtii et al. 2016).

## A Detailed results

Figures 10 and 11 in this section plot the solutions to the problems presented in Sect. 5.1 as a function of time. For details see Sect. 5.1.

Figures 12, 13 and 14 in this section present detailed results from the DETEST test set. For a detailed description, see Sect. 5 and Hull et al. (1972).

## B A fourth-order four-step Runge–Kutta formula expressed as LTI SDE filtering solution

Runge–Kutta (RK) formulas are a family of one step solvers for ODEs. At time $t_{n+1}$, the numerical approximation is defined

$$
\begin{aligned}
y_{t_{n+1}} &= y_{t_n} + h \sum_{i=1}^{s} b_i k_{i,n} \\
k_{i,n} &= f\left(t_n + hc_i, \ y_{t_n} + h \sum_{j=1}^{s} a_{ij} k_{j,n}\right)
\end{aligned}
\tag{54}
$$

The parameters $a_{ij}, c_i$ and $b_i$ are usually expressed as a matrix $\mathbf{A}$ and two vectors $\mathbf{b}, \mathbf{c}$, written compactly in a so-called Butcher tableau:

$$
\begin{array}{c|ccc}
c_1 & a_{11} & \dots & a_{1s} \\
\vdots & \vdots & & \vdots \\
c_s & a_{s1} & \dots & a_{ss} \\
\hline
& b_1 & \dots & b_s
\end{array}
\tag{55}
$$

If the matrix $\mathbf{A}$ is strictly lower triangular and $c_1 = 0$, Eq. (54) simplifies to an iterative procedure of explicit equations

$$
\begin{aligned}
k_{1,n} &= f(t_n, \ y_{t_n}), \\
k_{i,n} &= f\left(t_n + c_i h, \ y_{t_n} + h \sum_{j=1}^{i-1} a_{ij} k_{j,n}\right), \quad i = 2, \dots, s \\
y_{t_{n+1}} &= y_{t_n} + h \sum_{i=1}^{s} b_i k_{i,n},
\end{aligned}
\tag{56}
$$

and in this case, the formula is called an *explicit RK method*. A RK method is said to be of order $q$, iff

$$
|y(t_0 + h) - y_{t_1}| \le Ch^{q+1},
\tag{57}
$$

where $C$ is a constant independent of $h$. In this case, the global error is of order $h^q$. It can be shown that there exist RK methods for which the order $q = s$ the number of vector field evaluations up to and including $q = s = 4$. Furthermore, there are no RK methods for which $q = s$, if $q \ge 5$. For a full description, we refer the reader to Hairer et al. (1987).
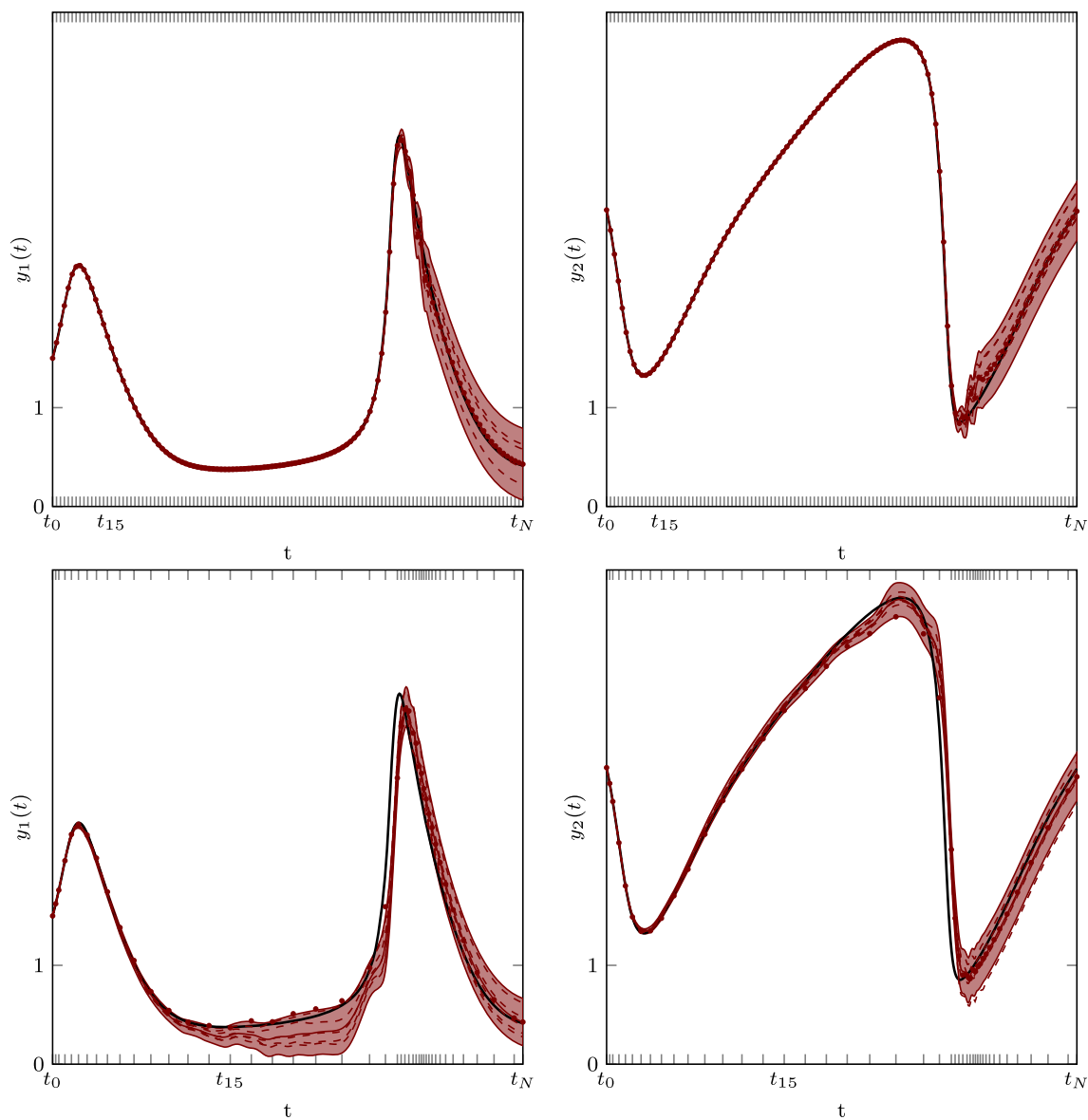
**Fig. 10** Numerical solution of the Brusselator (50) using the probabilistic filtering ODE solver plotted against time. The plot shows the true solution (black line), the mean of the filtering distribution (red dots), the posterior mean (red, thick line) plusminus two times standard deviation (light red, filled area) and samples from the posterior (red, dashed line). Tickmarks in $t$ indicate mesh points. Top: using a fixed step size. Bottom: using adaptive step size selection (see Fig. 11). (Color figure online)

Schober et al. (2014a) have presented probabilistic models whose MAP at $t_0 + h = t_1$ is equivalent to an explicit RK method of type Eq. (56) in the case of $q \in \{1, 2, 3\}$. The corresponding probabilistic model is given by

$$d\mathbf{X} = \mathbf{U}_{q+1}\mathbf{X}\, dt + \mathbf{e}_q\, dW, \tag{58}$$

where $(\mathbf{U}_{q+1})_{ij} = \delta_{i,j+1}$, $i, j = 0, \ldots, q$ is the $q + 1$-dimensional upper shift matrix and $\mathbf{e}_q = (0, \ldots, 0, 1)^{\mathsf{T}}$ is the $q+1^{\text{th}}$ standard basis vector. Process (58) is known as $q$-times

*integrated Wiener process* IWP($q$). The initial conditions are

$$\mathbf{m}_{t_{-1}} = \mathbf{0}, \quad \mathbf{C}_{t_{-1}} = \lim_{\mathcal{H} \to \infty} \mathbf{Q}(\mathcal{H}). \tag{59}$$

See Schober et al. (2014a) for details.

Assume that Algorithm 2 has been run until the loop has been evaluated four times with $h_n$ such that $(t_0, \ldots, t_3) = \mathbf{c}^{\mathsf{T}}h = (0, uh, vh, 1)$, where $u, v$ are two constant in $[0, 1]$ chosen by the user. Then, $lim_{\mathcal{H} \to \infty}\mathbf{m}_{t_3}$ for the IWP(4) is
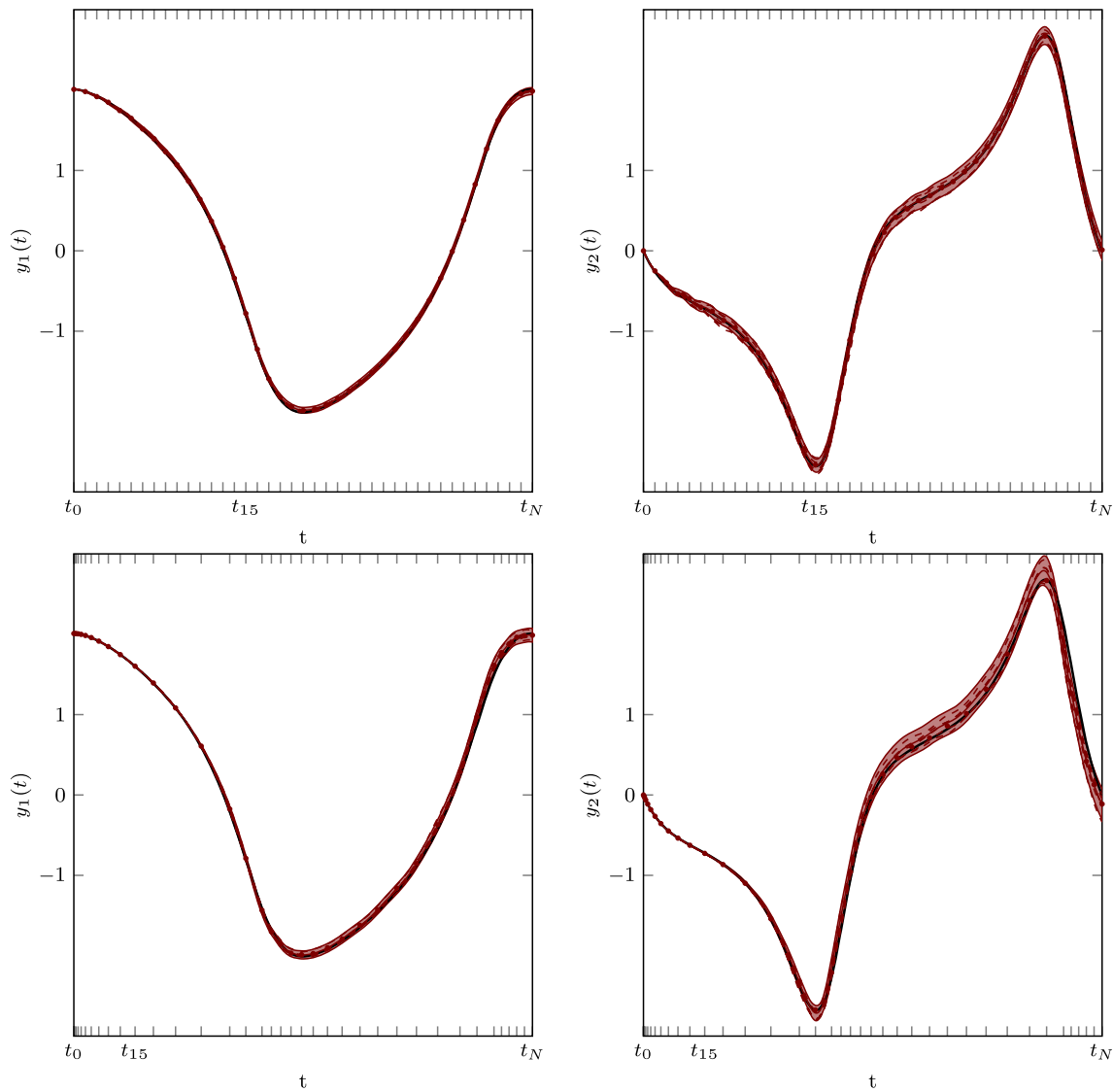
**Fig. 11** Numerical solution of van der Pol's equation (51) using the probabilistic filtering ODE solver plotted against time. The plot shows the true solution (black line), the mean of the filtering distribution (red dots), the posterior mean (red, thick line) plusminus two times stan- dard deviation (light red, filled area) and samples from the posterior (red, dashed line). Tickmarks in $t$ indicate mesh points. Top: using a fixed step size. Bottom: using adaptive step size selection. (Color figure online)

$$(\mathbf{m}_{t_3})_0 = y_0 + h\frac{1 - 2(u+v) + 6uv}{12uv}z_0 + h\frac{2v - 1}{12u(u-v)(u-1)}z_1$$
$$+ h\frac{1 - 2u}{12v(u-v)(v-1)}z_2 + h\frac{3 - 4(u+v) + 6uv}{12(u-1)(v-1)}z_3$$

$$(\mathbf{m}_{t_3})_1 = z_3$$

$$(\mathbf{m}_{t_3})_2 = \frac{1}{h}\frac{u+v-uv-1}{uv}z_0 + \frac{1}{h}\frac{1-v}{u(u-v)(u-1)}z_1$$
$$+ \frac{1}{h}\frac{u-1}{v(u-v)(v-1)}z_2 + \frac{1}{h}\frac{3 - 2(u+v) + uv}{(u-1)(v-1)}z_3$$

$$(\mathbf{m}_{t_3})_3 = \frac{1}{h^2}\frac{2(u+v-2)}{uv}z_0 + \frac{1}{h^2}\frac{2(2-v)}{u(u-v)(u-1)}z_1$$
$$+ \frac{1}{h^2}\frac{2(u-2)}{v(u-v)(v-1)}z_2 + \frac{1}{h^2}\frac{2(3-u-v)}{(u-1)(v-1)}z_3$$

$$(\mathbf{m}_{t_3})_4 = \frac{1}{h^3}\frac{-6}{uv}z_0 + \frac{1}{h^3}\frac{6}{u(u-v)(u-1)}z_1$$

$$+ \frac{1}{h^3}\frac{-6}{v(u-v)(v-1)}z_2 + \frac{1}{h^3}\frac{6}{(u-1)(v-1)}z_3$$

Furthermore, we get the following algebraic equations for the elements of the covariance matrix $lim_{\mathcal{H}\to\infty}\mathbf{C}_{t_3}$:

$$(\mathbf{C}_{t_3})_{00} = \sigma^2 h^9 \Big[ 6u^6v^2 - 3u^6v + 6u^5v^3 - 27u^5v^2 + 20u^5v$$
$$- 4u^5 + 6u^4v^4 - 27u^4v^3 + 28u^4v^2 - 12u^4v$$
$$+ 2u^4 + 6u^3v^5 - 27u^3v^4 + 28u^3v^3 - 12u^3v^2$$
$$+ 2u^3v + 6u^2v^6 - 27u^2v^5 + 28u^2v^4 + 68u^2v^3$$
$$- 78u^2v^2 + 20u^2v - 9uv^6 + 38uv^5 - 42uv^4$$
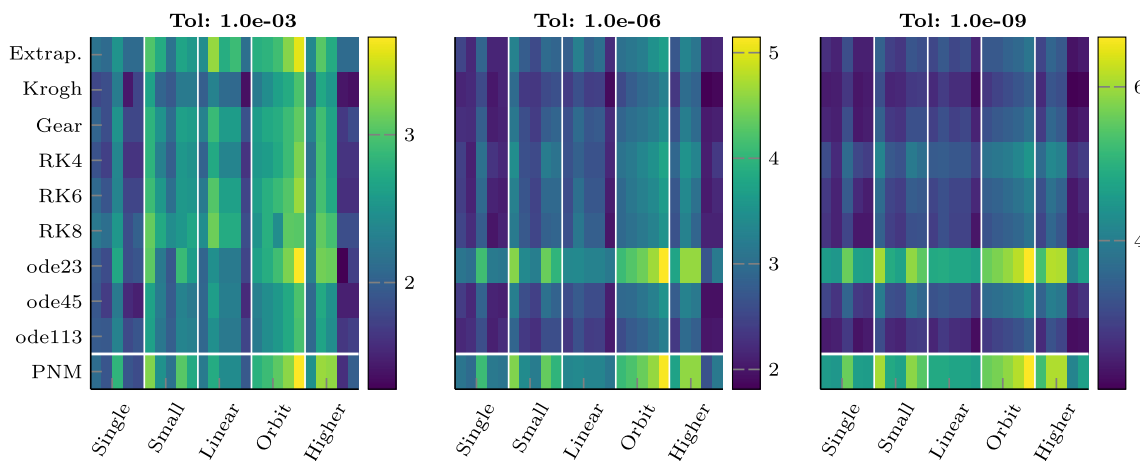$$- 48uv^3 + 70uv^2 - 20uv + 3v^6 - 13v^5 + 17v^4$$

**Fig. 12** $\log_{10}(\#FE)$, the number of function evaluations in logarithmic scale, for all tested methods and individual problems
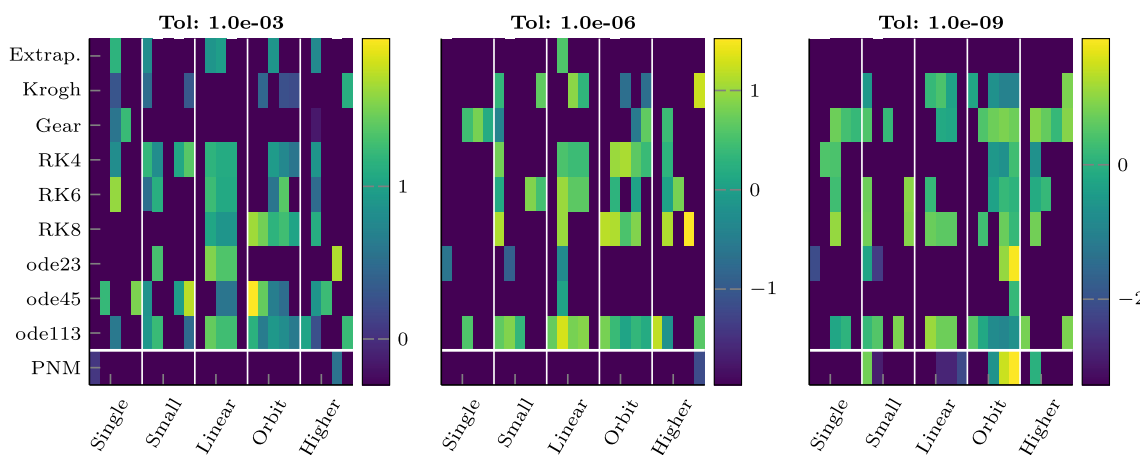


**Fig. 13** $\log_{10}(|\{\xi_n \,|\, \xi_n > \epsilon, n = 1, \ldots, N\}|N^{-1})$, the percent of deceived steps in logarithmic scale, for all tested methods and individual problems
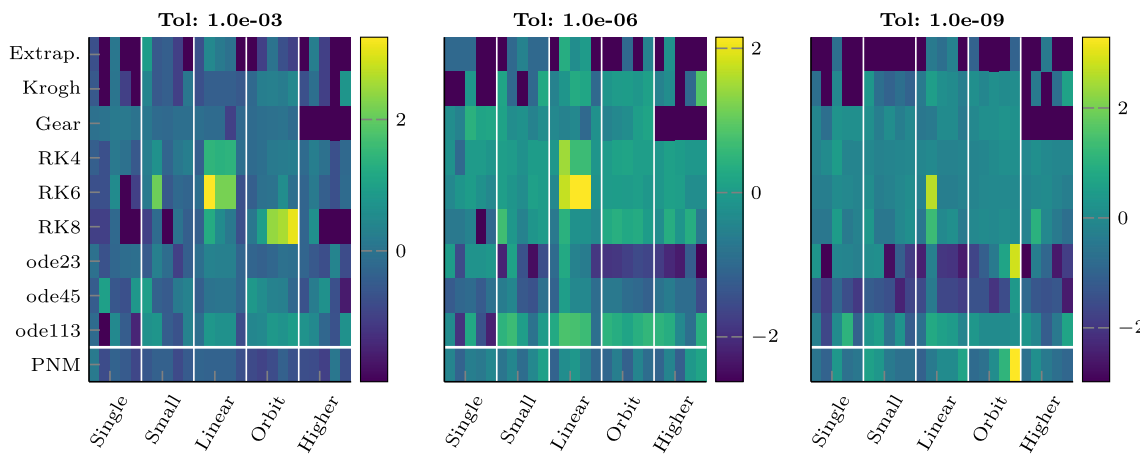


**Fig. 14** $\log_{10}(\max\{\xi_n[h_n\epsilon]^{-1} \,|\, n = 1, \ldots, N\})$, the maximum error per unit step in logarithmic scale, for all tested methods and individual problems

$$+ 5v^3 - 15v^2 + 5v\Big]\Big[725760v(1-u)\Big]^{-1}$$

$(\mathbf{C}_{t_3})_{01} = 0$

$$(\mathbf{C}_{t_3})_{02} = \sigma^2 h^7 \Big[(v-1)(3u^6v+3u^5v^2-16u^5v+6u^5+3u^4v^3$$
$$- 16u^4v^2 + 14u^4v - 4u^4 + 3u^3v^4 - 16u^3v^3$$
$$+ 14u^3v^2 - 4u^3v + 3u^2v^5 - 16u^2v^4 + 14u^2v^3$$
$$+ 76u^2v^2 - 40u^2v - 6uv^5 + 29uv^4 - 24uv^3$$
$$- 85uv^2 + 50uv + 3v^5 - 14v^4 + 15v^3 + 20v^2$$
$$- 15v)\Big]\Big[120960v(u-1)\Big]^{-1}$$

$$(\mathbf{C}_{t_3})_{03} = \sigma^2 h^6 \Big[3u^6v^2 - 6u^6v + 3u^5v^3 - 18u^5v^2 + 32u^5v$$
$$- 10u^5 + 3u^4v^4 - 18u^4v^3 + 40u^4v^2 - 30u^4v$$
$$+ 8u^4 + 3u^3v^5 - 18u^3v^4 + 40u^3v^3 - 30u^3v^2$$
$$+ 8u^3v + 3u^2v^6 - 18u^2v^5 + 40u^2v^4 + 50u^2v^3$$
$$- 192u^2v^2 + 80u^2v - 9uv^6 + 41uv^5 - 69uv^4$$
$$- 81uv^3 + 271uv^2 - 117uv + 6v^6 - 28v^5$$
$$+ 50v^4 + 2v^3 - 78v^2 + 39v\Big]\Big[60480v(u-1)\Big]^{-1}$$

$$(\mathbf{C}_{t_3})_{04} = \sigma^2 h^5 \Big[3u^6v + 3u^5v^2 - 12u^5v + 4u^5 + 3u^4v^3$$
$$- 12u^4v^2 + 12u^4v - 4u^4 + 3u^3v^4 - 12u^3v^3$$
$$+ 12u^3v^2 - 4u^3v + 3u^2v^5 - 12u^2v^4 + 12u^2v^3$$
$$+ 76u^2v^2 - 40u^2v + 3uv^6 - 12uv^5 + 12uv^4$$
$$+ 16uv^3 - 140uv^2 + 72uv - 3v^6 + 13v^5 - 19v^4$$
$$+ 5v^3 + 45v^2 - 27v\Big]\Big[20160v(1-u)\Big]^{-1}$$

$(\mathbf{C}_{t_3})_{11} = 0$

$(\mathbf{C}_{t_3})_{12} = 0$

$(\mathbf{C}_{t_3})_{13} = 0$

$(\mathbf{C}_{t_3})_{14} = 0.$

The last four equations are a consequence of the noise-free observation $z_3$ at $t_0 + c_4 h = t_0 + h = t_3$. The remaining entries are given by the expressions

$$(\mathbf{C}_{t_3})_{22} = \sigma^2 h^5 [(v-1)^2(u^4 + u^3v + u^2v^2 + uv^3 - 10uv$$
$$- 2v^3 + 2v^2 + 6v)][2520v]^{-1}$$

$$(\mathbf{C}_{t_3})_{23} = \sigma^2 h^4 [(v-1)(u^5v - 3u^5 + u^4v^2 - 5u^4v + 4u^4$$
$$+ u^3v^3 - 5u^3v^2 + 4u^3v + u^2v^4 - 5u^2v^3$$
$$- 16u^2v^2 + 40u^2v - 5uv^4 + 15uv^3 + 37uv^2$$
$$- 77uv + 5v^4 - 15v^3 - 11v^2 + 33v)]$$
$$[2520v(u-1)]^{-1}$$

$$(\mathbf{C}_{t_3})_{24} = \sigma^2 h^3 [(v-1)(-u^5 - u^4v + 2u^4 - u^3v^2 + 2u^3v$$
$$- u^2v^3 + 2u^2v^2 + 20u^2v - uv^4 + 2uv^3$$
$$+ 5uv^2 - 50uv + 2v^4 - 5v^3 + 25v)]$$
$$[840v(u-1)]^{-1}$$

$$(\mathbf{C}_{t_3})_{33} = \sigma^2 h^3 [u^5v - 2u^5 + 2u^4v^2 - 6u^4v + 4u^4 + 2u^3v^3$$
$$- 6u^3v^2 + 4u^3v + 2u^2v^4 + 4u^2v^3 - 36u^2v^2$$
$$+ 40u^2v + uv^5 - 12uv^4 - 12uv^3 + 104uv^2$$
$$- 96uv - 2v^5 + 16v^4 - 8v^3 - 48v^2 + 48v]$$
$$[630v(1-u)]^{-1}$$

$$(\mathbf{C}_{t_3})_{34} = \sigma^2 h^2 [u^5 + 3u^4v - 4u^4 + 3u^3v^2 - 4u^3v + 3u^2v^3$$
$$+ 16u^2v^2 - 40u^2v + 3uv^4 + uv^3 - 95uv^2$$
$$+ 135uv + v^5 - 10v^4 + 14v^3 + 54v^2 - 81v]$$
$$[420v(u-1)]^{-1}$$

$$(\mathbf{C}_{t_3})_{44} = \sigma^2 h [u^4 + u^3v + u^2v^2 + 10u^2v + uv^3 + 20uv^2$$
$$- 60uv + v^4 - 5v^3 - 15v^2 + 45v][70v(1-u)]^{-1}$$

which defines the entire matrix since $\mathbf{C}_{t_3}$ is a symmetric matrix.

The proof that this specific choice of $\mathbf{c}$ is analogous to the proofs given in Schober et al. (2014a) and can be checked with laborious algebra.

We would like to point to one more detail: Although it can easily be checked that $(\mathbf{m}_{t_3})_0$ is of the required form to produce the RK prediction, this does not suffice to show that this choice of evaluation knots produces Runge–Kutta. The space of suitable parameters to produce a $q$th-order method is constrained by the expansion to match the Taylor coefficients. In the case of the IWP($q$), where each subsequent evaluation increases the order of the polynomial approximation, this entails that each partial RK-step needs to be a RK method of its own right to produce an overall RK method of high order. One can think about this as a bigger set of constraints that needs to be fulfilled. As a consequence, this also entails that there is no meaningful interpretation of RK methods with $v \neq 2/3$ in the case of the IWP(3) as has erroneously been conjectured in Schober et al. (2014a).

For complete details, see Hairer et al. (1987) and Schober et al. (2014a).

## References

Albrecht, P.: Explicit, optimal stability functionals and their application to cyclic discretization methods. Computing **19**(3), 233–249 (1978)

Andria, G.D., Byrne, G.D., Hill, D.R.: Integration formulas and schemes based on g-splines. Math. Comput. **27**(124), 831–838 (1973)

Brown, P., Byrne, G., Hindmarsh, A.: Vode: a variable-coefficient ode solver. SIAM J. Sci. Stat. Comput. **10**(5), 1038–1051 (1989)

Butcher, J.: General linear method: a survey. Appl. Numer. Math. **1**(4), 273–284 (1985)

Byrne, G.D., Chi, D.N.H.: Linear multistep formulas based on g-splines. SIAM J. Numer. Anal. **9**(2), 316–324 (1972)

Byrne, G.D., Hindmarsh, A.C.: A polyalgorithm for the numerical solution of ordinary differential equations. ACM Trans. Math. Softw. **1**(1), 71–96 (1975)

Chkrebtii, O.A., Campbell, D.A., Calderhead, B., Girolami, M.A.: Bayesian solution uncertainty quantification for differential equations. Bayesian Anal. **11**(4), 1239–1267 (2016)

Cockayne, J., Oates, C., Sullivan, T., Girolami, M.: Bayesian Probabilistic Numerical Methods. ArXiv e-prints (2017)

Conrad, P.R., Girolami, M., Särkkä, S., Stuart, A., Zygalakis, K.: Statistical analysis of differential equations: introducing probability measures on numerical solutions. Stat. Comput. **27**(4), 1065–1082 (2017)

Cox, R.: Probability, frequency and reasonable expectation. Am. J. Phys. **14**(1), 1–13 (1946)

Crane, P., Fox, P.: A comparative study of computer programs for integrating differential equations. Bell Telephone Laboratories, New York (1969)

Crouzeix, M., Lisbona, F.: The convergence of variable-stepsize, variable-formula, multistep methods. SIAM J. Numer. Anal. **21**(3), 512–534 (1984)

Deuflhard, P.: Order and stepsize control in extrapolation methods. Numer. Math. **41**(3), 399–422 (1983)

Deuflhard, P., Bornemann, F.: Scientific Computing with Ordinary Differential Equations. Springer, New York (2002)

Diaconis, P.: Bayesian numerical analysis. Stat. Decis. Theory Relat. Top. **IV**(1), 163–175 (1988)

Gear, C.: Numerical solution of ordinary differential equations: Is there anything left to do? SIAM Rev. **23**(1), 10–24 (1981)

Gear, C.W.: Runge-Kutta starters for multistep methods. ACM Trans. Math. Softw. **6**(3), 263–279 (1980)

Giné, E., Nickl, R.: Mathematical Foundations of Infinite-Dimensional Statistical Models, vol. 40. Cambridge University Press, Cambridge (2015)

Grewal, M.S., Andrews, A.P.: Kalman Filtering: Theory and Practice Using MATLAB. Wiley, New York (2001)

Griewank, A., Walther, A.: Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation, 2nd edn. No. 105 in Other Titles in Applied Mathematics. SIAM, Philadelphia (2008)

Grigorievskiy, A., Lawrence, N., Särkkä, S.: Parallelizable Sparse Inverse Formulation Gaussian Processes (SpInGP). ArXiv e-prints (2016)

Hairer, E., Nørsett, S., Wanner, G.: Solving Ordinary Differential Equations I-Nonstiff Problems. Springer, Berlin (1987)

Hartikainen, J., Särkkä, S.: Kalman filtering and smoothing solutions to temporal Gaussian process regression models. IEEE International Workshop on Machine Learning for Signal Processing (MLSP) **2010**, 379–384 (2010)

Hauberg, S., Schober, M., Liptrot, M., Hennig, P., Feragen, A.: A random riemannian metric for probabilistic shortest-path tractography. In: Navab, N., Hornegger, J., Wells, W., Frangi, A. (eds.) Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015. MICCAI 2015. Lecture Notes in Computer Science, vol 9349. Springer, Cham (2015)

Hennig, P., Osborne, M.A., Girolami, M.: Probabilistic numerics and uncertainty in computations. Proc. R. Soc. Lond. A Math. Phys. Eng. Sci. **471**(2179), 20150142 (2015)

Hull, T., Enright, W., Fellen, B., Sedgwick, A.: Comparing numerical methods for ordinary differential equations. SIAM J. Numer. Anal **9**(4), 603–637 (1972)

Jazwinski, A.H.: Stochastic Processes and Filtering Theory. Academic Press, London (1970)

Jeffreys, H.: Theory of Probability, 3rd edn. Oxford University Press, Oxford (1969)

Kalman, R.E.: A new approach to linear filtering and prediction problems. J. Fluids Eng. **82**(1), 35–45 (1960)

Karatzas, I., Shreve, S.E.: Brownian Motion and Stochastic Calculus. Springer, Berlin (1991)

Kersting, H.P., Hennig, P.: Active uncertainty calibration in Bayesian ODE solvers. In: Janzing, I. (eds.) Uncertainty in Artificial Intelligence (UAI), vol. 32. AUAI Press (2016)

Kimeldorf, G.S., Wahba, G.: A correspondence between bayesian estimation on stochastic processes and smoothing by splines. Ann. Math. Stat. **41**(2), 495–502 (1970)

Krogh, F.T.: On testing a subroutine for the numerical integration of ordinary differential equations. J. ACM **20**(4), 545–562 (1973)

Lancaster, P., Rodman, L.: Algebraic riccati equations. Clarendon press, Oxford (1995)

Lefever, R., Nicolis, G.: Chemical instabilities and sustained oscillations. J. Theor. Biol. **30**(2), 267–284 (1971)

Loscalzo, F.R.: An introduction to the application of spline functions to initial value problems. In: Greville, T.N.E. (ed.) Theory and Applications of Spline Functions, pp. 37–64. Academic Press, New York (1969)

Loscalzo, F.R., Talbot, T.D.: Spline function approximations for solutions of ordinary differential equations. SIAM J. Numer. Anal. **4**(3), 433–445 (1967)

Mazzia, F., Sestini, A., Trigiante, D.: B-Spline linear multistep methods and their continuous extensions. SIAM J. Numer. Anal. **44**(5), 1954–1973 (2006)

Mazzia, F., Sestini, A., Trigiante, D.: The continuous extension of the b-spline linear multistep methods for BVPs on non-uniform meshes. Appl. Numer. Math. **59**(3–4), 723–738 (2009). Selected Papers from NUMDIFF-11

Nordsieck, A.: On numerical integration of ordinary differential equations. Math. Comput. **16**(77), 22–49 (1962)

O'Hagan, A.: Some Bayesian numerical analysis. Bayesian. Stat. **4**, 345–363 (1992)

Øksendal, B.: Stochastic Differential Equations: An Introduction with Applications, 6th edn. Springer, Berlin (2003)

Owhadi, H., Scovel, C.: Toward machine Wald. In: Ghanem, R., Higdon, D., Owhadi, H. (eds.) Springer Handbook of Uncertainty Quantification, pp. 1–35. Springer (2016)

Paskov, S.H.: Average case complexity of multivariate integration for smooth functions. J. Complex. **9**(2), 291–312 (1993)

Poincaré, H.: Calcul des probabilités. Gauthier-Villars, Paris (1896)

Rauch, H.E., Striebel, C., Tung, F.: Maximum likelihood estimates of linear dynamic systems. AIAA J. **3**(8), 1445–1450 (1965)

Särkkä, S.: Recursive Bayesian Inference on Stochastic Differential Equations. Ph.D. thesis, Helsinki University of Technology (2006)

Särkkä, S.: Bayesian Filtering and Smoothing. Cambridge University Press, Cambridge (2013)

Särkkä, S., Solin, A., Hartikainen, J.: Spatiotemporal learning via infinite-dimensional Bayesian filtering and smoothing. IEEE Signal Process. Mag. **30**(4), 51–61 (2013)

Schober, M., Duvenaud, D., Hennig, P.: Probabilistic ODE Solvers with Runge–Kutta Means. In: Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N.D., Weinberger, K.Q. (eds.) Advances in Neural Information Processing Systems 27, pp. 739–747. Curran Associates, Inc. (2014a)

Schober, M., Kasenburg, N., Feragen, A., Hennig, P., Hauberg, S.: Probabilistic shortest path tractography in DTI using Gaussian Process ODE solvers. In: Medical Image Computing and Computer-Assisted Intervention–MICCAI 2014. Springer (2014b)

Shampine, L., Watts, H., Davenport, S.: Solving nonstiff ordinary differential equations-the state of the art. SIAM Rev. **18**, 376–411 (1976)

Skeel, R.: Analysis of fixed-stepsize methods. SIAM J. Numer. Anal. **13**(5), 664–685 (1976)

Skeel, R.D.: Equivalent forms of multistep formulas. Math. Comput. **33**(148), 1229–1250 (1979)

Skeel, R.D., Jackson, L.W.: Consistency of nordsieck methods. SIAM J. Numer. Anal. **14**(5), 910–924 (1977)

Skilling, J.: Bayesian solution of ordinary differential equations. In: Smith, C.R., Erickson, G.J., Neudorfer, P.O. (eds.) Maximum Entropy and Bayesian Methods. Fundamental Theories of Physics (An International Book Series on The Fundamental Theories of Physics: Their Clarification, Development and Application), vol 50. Springer, Dordrecht (1992)

Solin, A.: Stochastic Differential Equation Methods for Spatio-Temporal Gaussian Process Regression. Ph.D. thesis, Aalto University, Helsinki 2016)

Stuart, A.M.: Inverse problems: a Bayesian perspective. Acta Numer. **19**, 451559 (2010)

Sullivan, T.J.: Introduction to Uncertainty Quantification, vol. 63. Springer, Berlin (2015)

Teymur, O., Zygalakis, K., Calderhead, B.: Probabilistic linear multistep methods. Adv. Neural Inf. Process. Syst. (2016)

van der Pol, B.: Lxxxviii. on relaxation-oscillations. Lond. Edinb. Dublin Philos. Mag. J. Sci. **2**(11), 978–992 (1926)

Wahba, G.: Spline Models for Observational Data. No. 59 in CBMS-NSF Regional Conferences Series in Applied Mathematics. SIAM (1990)

Wald, A.: Sequential Analysis. Courier Corporation, Mineola (1973)