Numerical Solutions of ODEs by Gaussian (Kalman) Filtering

Hans Kersting
joint work with Michael Schober, Philipp Hennig, Tim Sullivan and Han C. Lie

SIAM CSE, Atlanta
March 1, 2017

MAX-PLANCK-GESELLSCHAFT

Emmy Noether Group on Probabilistic Numerics
Department of Empirical Inference
Max Planck Institute for Intelligent Systems
Tübingen, Germany

Emmy Noether-Programm
DFG Deutsche Forschungsgemeinschaft

Contents

Numerical methods such as

- linear algebra (least-squares)
- optimization (training & fitting)
- integration (MCMC, marginalization)
- solving differential equations (RL, control)

output approximate solutions for unknown quantities.

Numerical methods such as

- linear algebra (least-squares)
- optimization (training & fitting)
- integration (MCMC, marginalization)
- solving differential equations (RL, control)

output approximate solutions for unknown quantities.
Probabilistic numerics aimes to produce probability measures instead,

Numerical methods such as

- linear algebra (least-squares)
- optimization (training & fitting)
- integration (MCMC, marginalization)
- solving differential equations (RL, control)

output approximate solutions for unknown quantities.
Probabilistic numerics aimes to produce probability measures instead, which are supposed to capture our epistemic uncertainty over the solution.

Numerical methods perform inference

an old observation [Poincaré 1896, Diaconis 1988, O'Hagan 1992]

A numerical method
estimates a function's latent property
given the result of computations.

quadrature	estimates $\int_a^b f(x)\,dx$	given $\{f(x_i)\}$
linear algebra	estimates x s.t. $Ax = b$	given $\{As = y\}$
optimization	estimates x s.t. $\nabla f(x) = 0$	given $\{\nabla f(x_i)\}$
analysis	estimates $x(t)$ s.t. $x' = f(x,t)$,	given $\{f(x_i, t_i)\}$

- computations yield "data" / "observations"
- non-analytic quantities are "latent"
- even deterministic quantities can be uncertain.

Numerical methods perform inference

an old observation [Poincaré 1896, Diaconis 1988, O'Hagan 1992]

A numerical method
estimates a function's latent property
given the result of computations.

quadrature	estimates $\int_a^b f(x)\,dx$	given $\{f(x_i)\}$
linear algebra	estimates x s.t. $Ax = b$	given $\{As = y\}$
optimization	estimates x s.t. $\nabla f(x) = 0$	given $\{\nabla f(x_i)\}$
analysis	estimates $x(t)$ s.t. $x' = f(x,t)$,	given $\{f(x_i, t_i)\}$

- computations yield "data" / "observations"
- non-analytic quantities are "latent"
- even deterministic quantities can be uncertain.

Probabilistic numerics uses this link between statistics and numerics to

Numerical methods perform inference

an old observation [Poincaré 1896, Diaconis 1988, O'Hagan 1992]

A numerical method
estimates a function's latent property
given the result of computations.

quadrature	estimates $\int_a^b f(x)\,dx$	given $\{f(x_i)\}$
linear algebra	estimates x s.t. $Ax = b$	given $\{As = y\}$
optimization	estimates x s.t. $\nabla f(x) = 0$	given $\{\nabla f(x_i)\}$
analysis	estimates $x(t)$ s.t. $x' = f(x,t)$,	given $\{f(x_i,t_i)\}$

- computations yield "data" / "observations"
- non-analytic quantities are "latent"
- even deterministic quantities can be uncertain.

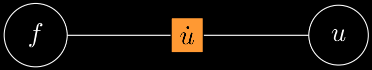
Probabilistic numerics uses this link between statistics and numerics to

(i) perform numerical computation in a statistically interpretable framework, and

(ii) enable an all–inclusive uncertainty quantification (for computations which include both numerical and statistical parts).

ODEs: Initial Value Problems (IVP)

$$\frac{\partial u}{\partial t}(t) = f(u(t), t), \quad u(0) = u_0 \in \mathbb{R}^n$$



I. In engineering, for example:

I. In engineering, for example:

1. modelling mechanical oscillations,
2. heating/cooling of engine parts, and
3. model predictive control.

Ordinary Differential Equations

Applications all over the place

I. In engineering, for example:

1. modelling mechanical oscillations,
2. heating/cooling of engine parts, and
3. model predictive control.

II. In AI, for example:

Ordinary Differential Equations

Applications all over the place

I. In engineering, for example:

1. modelling mechanical oscillations,
2. heating/cooling of engine parts, and
3. model predictive control.

II. In AI, for example:

1. Nesterov's Accelerated Gradient Descent
2. dynamically changing data, and
3. demand forecasting.

Challenge in AI: Most quantities involving the ODE can be uncertain:

1. initial value,
2. partial knowledge of vector field f
3. imprecise function evaluations, and
4. accumulated numerical errors.

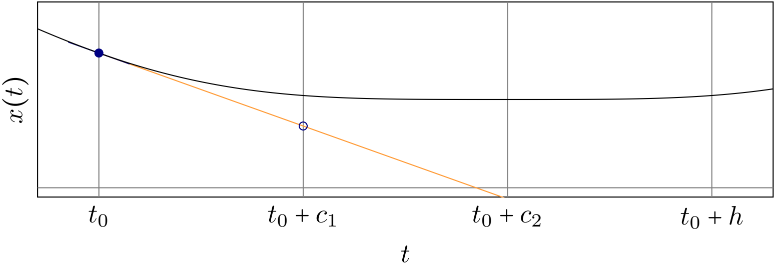
Numerical solutions of IVPs

plots: Runge-Kutta of order 3

How classical solvers extrapolate forward from time t_0 to $t_0 + h$:

- Estimate $\dot{x}(t_i)$, $t_0 \leq t_1 \leq \cdots \leq t_n \leq t_0 + h$ by evaluating $y_i \approx f(t, \hat{x}(t_i))$, where $\hat{x}(t)$ is itself an estimate for $x(t)$
- Use this data $y_i := \dot{x}(t_i)$ to estimate $x(t_0 + h)$, i.e.

$$\hat{x}(t_0 + h) \approx x(t_0) + h \sum_{i=1}^{b} w_i y_i.$$



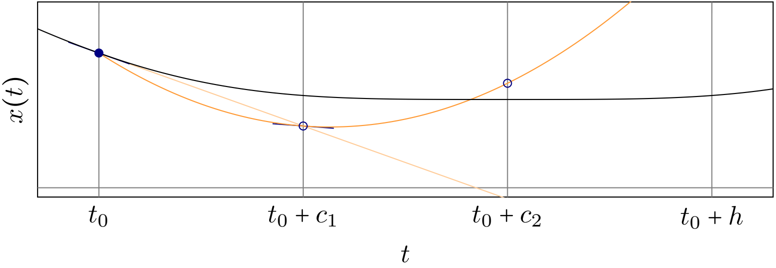
Numerical solutions of IVPs

plots: Runge-Kutta of order 3

How classical solvers extrapolate forward from time t_0 to $t_0 + h$:

- Estimate $\dot{x}(t_i)$, $t_0 \leq t_1 \leq \cdots \leq t_n \leq t_0 + h$ by evaluating $y_i \approx f(t, \hat{x}(t_i))$, where $\hat{x}(t)$ is itself an estimate for $x(t)$
- Use this data $y_i := \dot{x}(t_i)$ to estimate $x(t_0 + h)$, i.e.

$$\hat{x}(t_0 + h) \approx x(t_0) + h \sum_{i=1}^{b} w_i y_i.$$



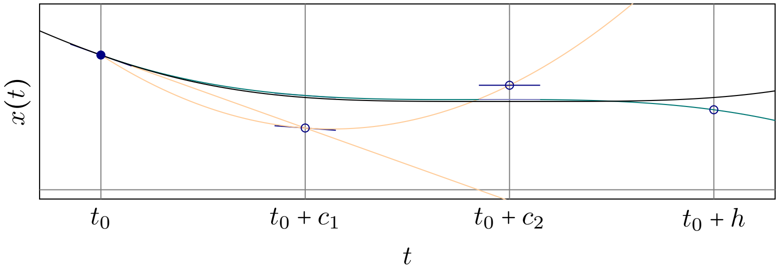
Numerical solutions of IVPs

plots: Runge-Kutta of order 3

How classical solvers extrapolate forward from time t_0 to $t_0 + h$:

- Estimate $\dot{x}(t_i)$, $t_0 \leq t_1 \leq \cdots \leq t_n \leq t_0 + h$ by evaluating $y_i \approx f(t, \hat{x}(t_i))$, where $\hat{x}(t)$ is itself an estimate for $x(t)$
- Use this data $y_i := \dot{x}(t_i)$ to estimate $x(t_0 + h)$, i.e.

$$\hat{x}(t_0 + h) \approx x(t_0) + h \sum_{i=1}^{b} w_i y_i.$$



Numerical solutions of IVPs

plots: Runge-Kutta of order 3

How classical solvers extrapolate forward from time t_0 to $t_0 + h$:

- Estimate $\dot{x}(t_i)$, $t_0 \leq t_1 \leq \cdots \leq t_n \leq t_0 + h$ by evaluating $y_i \approx f(t, \hat{x}(t_i))$, where $\hat{x}(t)$ is itself an estimate for $x(t)$
- Use this data $y_i := \dot{x}(t_i)$ to estimate $x(t_0 + h)$, i.e.

$$\hat{x}(t_0 + h) \approx x(t_0) + h \sum_{i=1}^{b} w_i y_i.$$

Uncertainty in these calculations:

- We can only observe x indirectly via $\hat{x}$.
- The observations of $\dot{x}(t) = f(t, \hat{x}(t))$ is inaccurate, since $\hat{x}(t) \approx x(t)$.
- There is uncertainty on our source of information $\hat{x}$, since it is both partial (i.e. discrete) and 'noisy'.
- The quantification of uncertainty on $\hat{x}$ is crucial to quantify uncertainty on x.

The Filtering Problem from Stochastic Calculus

Assume we have an unobservable *state* X_t of a dynamical system given by the SDE:

$$dX_t = b(t, X_t)dt + \sigma(t, X_t)dB_t.$$

We can only observe the *observations process* Z_t, a noisy transform of X_t, given by the SDE:

$$dZ_t = c(t, X_t)dt + \gamma(t, X_t)d\tilde{B}_t, \quad Z_0 = 0.$$

Filtering Problem: What is the L^2-best estimate $\hat{X}_t$ of X_t, based on observations $\{Z_{s_i} | s_i \leq t\}$?

IVPs as Filtering Problems:

- State is the unknown belief over $x(t)$
- Observation process is $\dot{x}(t)$ + 'noise'
- 'noise' process is due to the inaccurate evaluation position $\hat{x}(t)$ in $\dot{x}(t) \approx f(t, \hat{x}(t))$

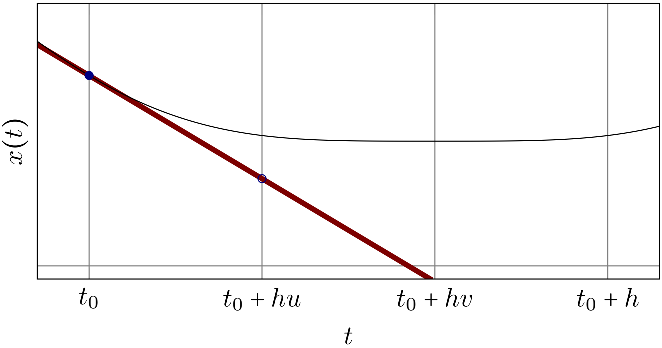
Hence,

(i) IVPs can be recast as Stochastic Filtering Problems,

(ii) and solved by Gaussian (Kalman) filtering.

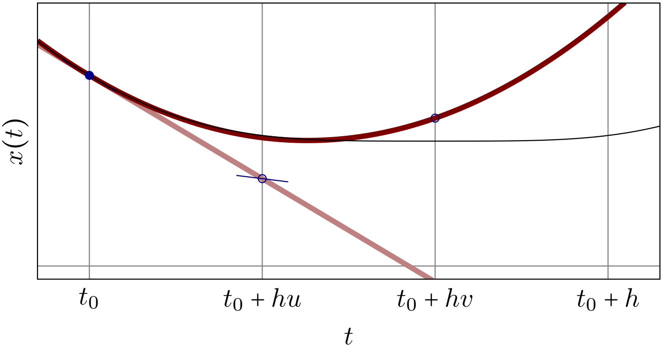
IVPs by Gaussian filtering

plots by M. Schober



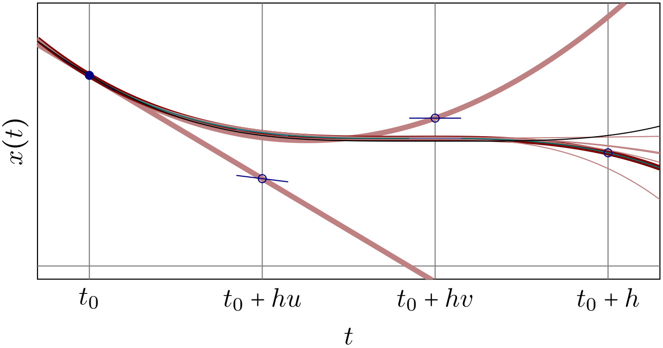
IVPs by Gaussian filtering

plots by M. Schober



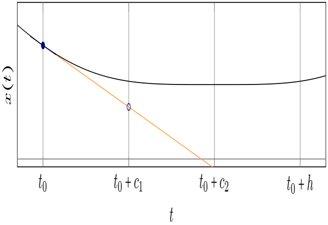
IVPs by Gaussian filtering

plots by M. Schober

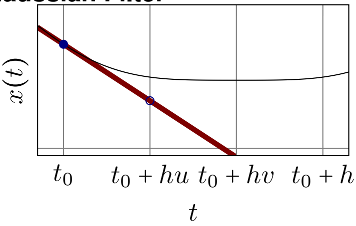


IVPs by Numerical Solver versus Gaussian Filtering

Numerical Solver

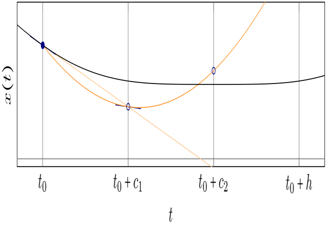


Gaussian Filter

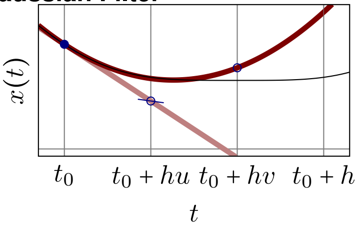


IVPs by Numerical Solver versus Gaussian Filtering

Numerical Solver

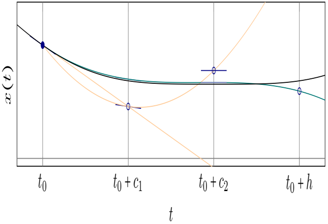


Gaussian Filter

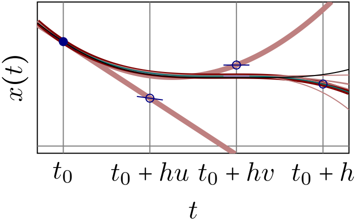


IVPs by Numerical Solver versus Gaussian Filtering

Numerical Solver

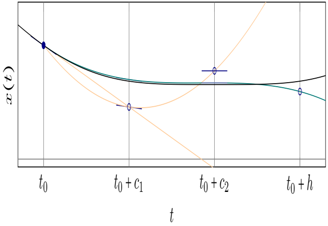


Gaussian Filter

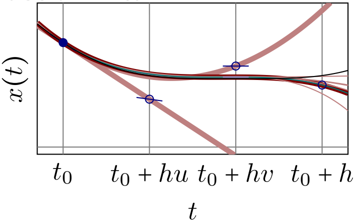


IVPs by Numerical Solver versus Gaussian Filtering

Numerical Solver



Gaussian Filter



The computation of the numerical mean and the posterior mean of Gaussian filtering share the same analytic structure [Schober et al., 2014]

We interpret $(u, \dot{u}, u^{(2)}, \ldots, u^{(q-1)})$ as a draw from a q-times-integrated Wiener process $(X_t)_{t\in[0,T]} = (X_t^{(1)}, \ldots, X_t^{(q)})^T_{t\in[0,T]}$ given by a linear SDE:

$$dX_t = FX_t dt + QdW_t,$$
$$X_0 = \xi, \quad \xi \sim \mathcal{N}(m(0), P(0))$$
$$\Longrightarrow \; X_t = \mathcal{GP}(A(t)m(0), A(t)P(0)A(t)^\top + Q), \quad A(t) = \exp(hF) \text{ and } Q(t) = \ldots$$

Filtering–based probabilistic ODE solvers

We interpret $(u, \dot{u}, u^{(2)}, \ldots, u^{(q-1)})$ as a draw from a q-times-integrated Wiener process $(X_t)_{t\in[0,T]} = (X_t^{(1)}, \ldots, X_t^{(q)})_{t\in[0,T]}^T$ given by a linear SDE:

$$dX_t = FX_t dt + QdW_t,$$

$$X_0 = \xi, \quad \xi \sim \mathcal{N}(m(0), P(0))$$

$$\implies X_t = \mathcal{GP}(A(t)m(0), A(t)P(0)A(t)^\top + Q), \quad A(t) = \exp(hF) \text{ and } Q(t) = \ldots$$

Calculation of Posterior by Gaussian filtering

Filtering–based probabilistic ODE solvers

We interpret $(u, \dot{u}, u^{(2)}, \dots, u^{(q-1)})$ as a draw from a q-times-integrated Wiener process $(X_t)_{t\in[0,T]} = (X_t^{(1)}, \dots, X_t^{(q)})^T_{t\in[0,T]}$ given by a linear SDE:

$$dX_t = FX_t dt + QdW_t,$$

$$X_0 = \xi, \quad \xi \sim \mathcal{N}(m(0), P(0))$$

$$\implies X_t = \mathcal{GP}(A(t)m(0), A(t)P(0)A(t)^\intercal + Q), \quad A(t) = \exp(hF) \text{ and } Q(t) = \dots$$

Calculation of Posterior by Gaussian filtering

Prediction step:

$$m^-_{t+h} = A(h)m_t,$$

$$P^-_{t+h} = A(h)P_tA(h)^T + Q(h),$$

Filtering–based probabilistic ODE solvers

Gaussian filtering [Schober et al., 2014]

We interpret $(u, \dot{u}, u^{(2)}, \dots, u^{(q-1)})$ as a draw from a q-times-integrated Wiener process $(X_t)_{t\in[0,T]} = (X_t^{(1)}, \dots, X_t^{(q)})^T_{t\in[0,T]}$ given by a linear SDE:

$$dX_t = FX_t dt + QdW_t,$$
$$X_0 = \xi, \quad \xi \sim \mathcal{N}(m(0), P(0))$$
$$\Longrightarrow X_t = \mathcal{GP}(A(t)m(0), A(t)P(0)A(t)^\top + Q), \quad A(t) = \exp(hF) \text{ and } Q(t) = \dots$$

Calculation of Posterior by Gaussian filtering

Prediction step:

$$m^-_{t+h} = A(h)m_t,$$
$$P^-_{t+h} = A(h)P_tA(h)^T + Q(h),$$

Vector field prediction at $t + h$:
Vector field y with uncertainty R
main source of uncertainty
cheaply quantified by Bayesian quadrature [Kersting and Hennig, 2016]

Filtering–based probabilistic ODE solvers

Gaussian filtering [Schober et al., 2014]

We interpret $(u, \dot{u}, u^{(2)}, \dots, u^{(q-1)})$ as a draw from a q-times-integrated Wiener process $(X_t)_{t\in[0,T]} = (X_t^{(1)}, \dots, X_t^{(q)})^T_{t\in[0,T]}$ given by a linear SDE:

$$dX_t = FX_t dt + QdW_t,$$
$$X_0 = \xi, \quad \xi \sim \mathcal{N}(m(0), P(0))$$
$$\implies X_t = \mathcal{GP}(A(t)m(0), A(t)P(0)A(t)^\top + Q), \quad A(t) = \exp(hF) \text{ and } Q(t) = \dots$$

Calculation of Posterior by Gaussian filtering

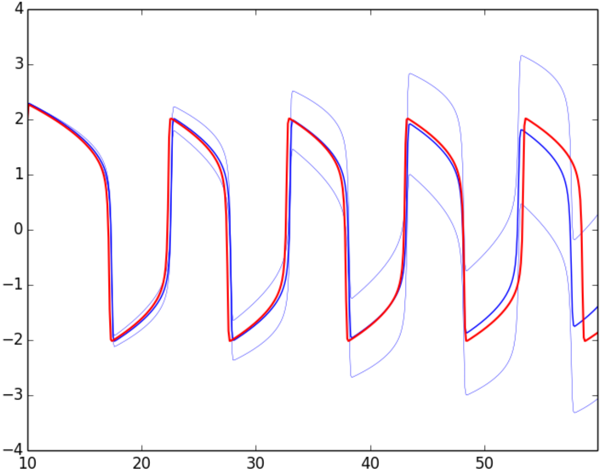
Prediction step:

$$m^-_{t+h} = A(h)m_t,$$
$$P^-_{t+h} = A(h)P_t A(h)^T + Q(h),$$

Vector field prediction at $t+h$:
Vector field y with uncertainty R
main source of uncertainty
cheaply quantified by Bayesian quadrature [Kersting and Hennig, 2016]

Update step:

$$z = y - e_n^T m^-_{t+h},$$
$$S = e_n^T P^-_{t+h} e_n + R,$$
$$K = P^-_{t+h} e_n S^{-1},$$
$$m_{t+h} = m^-_{t+h} + Kz,$$
$$P_{t+h} = P^-_{t+h} - K e_n^T P^-_{t+h},$$



We can compute a probabilistic output (above 95% confidence interval) at a low computational overhead.

Does this solver live up to classical expectations?

Current project: Theoretical Analysis

For an Integrated Wiener Process prior, we have the following convergence rates for the posterior mean:

Theorem

Under some technical assumptions, we have, for all modeled dimensions $i \in \{0, \dots, q\}$, *globally that*

$$\sup_n \left\| m(nh)_i - x^{(i)}(nh) \right\| \leq K h^{q-i}, \tag{1}$$

and locally that

$$\left\| m(h)_i - x^{(i)}(h) \right\| \leq K h^{q+1-i}, \tag{2}$$

where $K > 0$ *is a constant independent of* h *and* n*.*

Does this solver live up to classical expectations?

Current project: Theoretical Analysis

For an Integrated Wiener Process prior, we have the following convergence rates for the posterior mean:

Theorem

Under some technical assumptions, we have, for all modeled dimensions $i \in \{0, \ldots, q\}$*, globally that*

$$\sup_n \left\| m(nh)_i - x^{(i)}(nh) \right\| \leq K h^{q-i}, \tag{1}$$

and locally that

$$\left\| m(h)_i - x^{(i)}(h) \right\| \leq K h^{q+1-i}, \tag{2}$$

where $K > 0$ *is a constant independent of* h *and* n*.*

Proof: On arxiv soon!

Summary

The PN perspective on ODEs:

1. Unknown numerical quantities are modeled as random variables
2. uncertainty arises from initial values, imprecise function evaluations, partial knowledge of functions and accumulated numerical errors,
3. modeling these uncertainties yields a stochastic filtering problem.

We have a solver which can

(i) solve IVP at comparable cost of Runge–Kutta,
(ii) performs consistent UQ for all sources of uncertainty
(iii) output a whole probability measures, including confidence intervals,
(iv) filter out higher derivatives of the solution simultaneously, and
(v) learn (e.g. a periodic) vector field, while solving an ODE.

More information at probabilistic-numerics.org.

More information at probabilistic-numerics.org.

Thank you for listening!

Bibliography

Hand Kersting and P. Hennig. Active Uncertainty Calibration in Bayesian ODE Solvers. *Uncertainty in Artificial Intelligence (UAI)*, 2016.

M. Schober, D. Duvenaud, and P. Hennig. Probabilistic ODE Solvers with Runge-Kutta Means. *Advances in Neural Information Processing Systems (NIPS)*, 2014.