Probabilistic ODE Solvers
with Runge-Kutta Means

Michael Schober*, David Duvenaud‡, Philipp Hennig*

*Research Group Elementary Intelligence
Department of Empirical Inference
Max Planck Institute for Intelligent Systems
Tübingen, Germany

‡Computational and Biological Learning Lab
Department of Engineering
Cambridge University
Can we assign a probability distribution over the solution to an ordinary differential equation (initial value problem)?

\[ x(t_0) = x_0 \quad \quad x'(t) = f(x(t), t) \]
The Probabilistic View on Computation
computing as the collection of information

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 and Statistical Estimators

several classic numerical algorithms identified precisely as maximum a-posteriori estimators

quadrature
Gaussian quadrature \[\text{[Diaconis, 1988, O'Hagan, 1991]}\]
Gaussian process regression

linear algebra
conjugate gradients \[\text{[Hennig, 2015]}\]
Gaussian conditioning

nonlinear optimization
BFGS \[\text{[Hennig & Kiefel, 2013]}\]
autoregressive filtering

ordinary differential equations
Runge-Kutta \[\text{[Schober et al., 2014]}\]
Gauss-Markov extrapolation
Runge-Kutta methods are linear extrapolators of high convergence order [Hairer et al., 1987].

\[
\begin{align*}
0 & \quad 1 \quad Y_1 = f(1x_0, t_0 + 0) \\
c_1 & \quad 1 \quad w_{11} \quad Y_2 = f(1x_0 + w_{11}Y_1, t_0 + c_1) \\
c_2 & \quad 1 \quad w_{21} \quad w_{22} \quad Y_{s+1} = f(1x_0 + \sum_i^s w_{si}Y_i, t_0 + c_s) \\
h & \quad 1 \quad b_1 \quad b_2 \quad b_3 \quad \hat{x}(t_0 + h) = 1x_0 + \sum_i b_iY_i
\end{align*}
\]
Runge-Kutta methods are linear extrapolators of high convergence order [Hairer et al., 1987]

\[ x(t_0 + h) = x_0 + b_1 Y_1 + b_2 Y_2 + b_3 Y_{s+1} \]

\[
\begin{align*}
Y_1 &= f(1x_0, t_0 + 0) \\
Y_2 &= f (1x_0 + w_{11}Y_1, t_0 + c_1) \\
Y_{s+1} &= f (1x_0 + \sum_i w_{si} Y_i, t_0 + c_s) \\
\hat{x}(t_0 + h) &= 1x_0 + \sum_i b_i Y_i
\end{align*}
\]

\[ \begin{array}{c|cccc}
0 & 1 \\
c_1 & 1 & w_{11} \\
c_2 & 1 & w_{21} & w_{22} \\
h & 1 & b_1 & b_2 & b_3 \\
\end{array} \]
Runge-Kutta methods are linear extrapolators of high convergence order [Hairer et al., 1987]

\[ Y_1 = f(x_0, t_0 + 0) \]
\[ Y_2 = f(x_0 + w_{11}Y_1, t_0 + c_1) \]
\[ Y_{s+1} = f(x_0 + \sum_i w_{si}Y_i, t_0 + c_s) \]
\[ \hat{x}(t_0 + h) = x_0 + \sum_i b_i Y_i \]
Runge-Kutta methods are linear extrapolators of high convergence order [Hairer et al., 1987]
Gaussian process solvers are also linear extrapolators

- Linear extrapolation suggests Gaussian process model
- Gaussian process solvers previously studied
  [Skilling (1991), Chkrebtii et al. (2014), Hennig & Hauberg (2014)]
Some properties of Gaussian measures

The only two equations you really need (in this group)

- closure under affine transformations \((x \in \mathbb{R}^N, y \in \mathbb{R}^M)\)

\[
p(x) \sim \mathcal{N}(m, P), \quad p(y|x) \sim \mathcal{N}(Hx + \nu, R)
\]

\[
\Rightarrow p\left(\begin{bmatrix} x \\ y \end{bmatrix}\right) \sim \mathcal{N}\left(\begin{bmatrix} m \\ Hm + \nu \end{bmatrix}, \begin{bmatrix} P & PH^T \\ PH & HPH^T + R \end{bmatrix}\right)
\]

- inference involves only linear algebra operations

\[
p\left(\begin{bmatrix} x \\ y \end{bmatrix}\right) \sim \mathcal{N}\left(\begin{bmatrix} m_1 \\ m_2 \end{bmatrix}, \begin{bmatrix} P_1 & C \\ C^T & P_2 \end{bmatrix}\right)
\]

\[
p(x | y) \sim \mathcal{N}(m_1 + CP_2^{-1}(y - m_2), P_1 - CP_2^{-1}C^T)
\]

\[
\Rightarrow \text{sequential Gaussian inference at linear cost (‘filtering’)}
\]
Gaussian process solvers implicitly define a Butcher tableau

\[
x(t) = x(t_0) + c_1 \frac{h}{w_{11}} + c_2 \frac{h}{w_{21} w_{22}} + h \sum_{i=1}^{3} b_i x_{0}(t_0 + i h)
\]

\[
\begin{array}{c|cccc}
0 & 1 \\
c_1 & 1 & w_{11} \\
c_2 & 1 & w_{21} & w_{22} \\
h & 1 & b_1 & b_2 & b_3 \\
\end{array}
\]

\[
y_1 = f(\mu_{|\cdot x_0}(t_0 + 0), t_0 + 0)
\]

\[
\mu_{|\cdot x_0}(t_0) := [k(t_0, t_0)] [k(t_0, t_0)]^{-1} (x_0)
\]

\[
= 1 x_0
\]
Gaussian process solvers implicitly define a Butcher tableau.

\[
x(t) = x_0 + w_{11} y_1
\]

\[
\begin{array}{c|ccc}
0 & 1 \\
c_1 & 1 & w_{11} \\
c_2 & 1 & w_{21} & w_{22} \\
h & 1 & b_1 & b_2 & b_3 \\
\end{array}
\]

\[
y_1 = f\left(\mu|_{x_0}(t_0 + 0), t_0 + 0\right)
\]

\[
y_2 = f\left(\mu|_{x_0, y_1}(t_0 + c_1), t_0 + c_1\right)
\]

\[
\mu|_{x_0, y_1}(t_0 + c_1) := \left[ k(t_0 + c_1, t_0) \right. \\
\left. k^\partial(t_0 + c_1, t_0) \right]^{-1} \left[ \frac{\partial k(t_0, t_0)}{\partial k(t_0, t_0)} \frac{\partial^2 k(t_0, t_0)}{\partial k^\partial(t_0, t_0)} \right]^{-1} (x_0)
\]

\[
= w_{10} x_0 + w_{11} y_1
\]
Gaussian process solvers

implicitly define a Butcher tableau

\[
\begin{array}{l}
\begin{array}{c|ccc}
0 & 1 & & \\
c_1 & 1 & w_{11} & \\
c_2 & 1 & w_{21} & w_{22} \\
h & 1 & b_1 & b_2 & b_3
\end{array}
\end{array}
\]

\[
y_1 = f \left( \mu | x_0 (t_0 + 0), t_0 + 0 \right)
\]

\[
y_2 = f \left( \mu | x_0, y_1 (t_0 + c_1), t_0 + c_1 \right)
\]

\[
y_{s+1} = f \left( \mu | x_0, y_i (t_0 + c_s), t_0 + c_s \right)
\]

\[
\mu | x_0, y_i (t_0 + c_s) := \left[ k(t_0 + c_s, t_0) \quad k^\Theta(t_0 + c_s, t_0 + c_i) \right] K^{-1} \begin{pmatrix} x_0 \\ y_i \end{pmatrix}
\]

\[
= w_{20} x_0 + \sum_{i=1}^{s} w_{2i} y_i
\]
Gaussian process solvers
implicitly define a Butcher tableau

\[ y_1 = f \left( \mu_{|x_0}(t_0 + 0), t_0 + 0 \right) \]
\[ y_2 = f \left( \mu_{|x_0,y_1}(t_0 + c_1), t_0 + c_1 \right) \]
\[ y_{s+1} = f \left( \mu_{|x_0,y_i}(t_0 + c_s), t_0 + c_s \right) \]
\[ \hat{x}(t_0 + h) = \mu_{|x_0,y_i}(t_0 + h) \]
\[ \mu_{|x_0,y_i}(t_0 + h) := \begin{bmatrix} k(t_0 + h, t_0) & k^3(t_0 + h, t_0 + c_i) \end{bmatrix} K^{-1} \begin{bmatrix} x_0 \\ y_i \end{bmatrix} \]
\[ = b_0 x_0 + \sum_{i=1}^{s} b_i y_i \]
Gauss-Markov-Runge-Kutta methods

a GP solver whose mean matches RK exactly

- RK choose \((c, w, b)\) such that \(\|\hat{x}(t_0 + h) - x(t_0 + h)\| = \mathcal{O}(h^p)\)
- polynomial form suggests integrated Wiener (polynomial spline) process
  \[
p(x(t)) = \mathcal{GP}(x(t); 0, k_s(t, t'))\]
  where
  \[
k_s(t, t') = \int \cdots \int_t^\tau \int \cdots \int_{t'}^\tau \min(\tilde{t}, \tilde{t}') \, d\tilde{t} \, d\tilde{t}'
  \]
- \(\tau \rightarrow -\infty\): improper prior \(p(x(t))\), proper posterior after \(s\) observations.
- \(k\)th-times integrated Wiener process gives \(k\)-order RK solver!
- Inherets RK guarantees. Gives closed-form solution for tableau (used to use numerical search!)
- a Markov (state-space) model, so inference is \(\mathcal{O}(s)\) (as opposed to usual \(\mathcal{O}(s^3)\) cost)
Calibrating Uncertainty within the parametrized class

- posterior mean $\mu | y = k K^{-1} y$ invariant under $k \rightarrow \theta^2 k$
- posterior covariance $k | y = k - k K^{-1} k$ scaled by $\theta^2$
- initial ideas for uncertainty calibration in paper (more to come)
Multi-Step Extension

[A. Nordsieck, 1962]

- probabilistic interpretation questions RK beyond $s$ steps
- ‘obvious’ solution is to continue filtering process
- result very similar, though not identical, to multi-step methods
Some Conceptual Open Questions

precise interpretation of posterior measure still evolving

How precise can the connection to multi-step methods be?

- order / stability conditions currently not fully understood
- flexibility is also a design criterion
- what about stiff problems?

What, precisely, does the posterior mean?

- width of Gaussian posterior should be inferred from regularity of ‘observed’ gradients. How, precisely, should this be done? (We have one particular solution)
- is the Gaussian family enough? How expensive is it to move beyond Gauss?
What we’ve done so far:

- Numerical methods can be interpreted as performing statistical inference from noise-free data
- In some cases, e.g. Runge-Kutta, this link can be made precise
- Inherits convergence guarantees, but also get extensibility & uncertainty estimates

What we’re working on next:

- Understand the connection to multi-step methods
- Construct a robust probabilistic IVP solver
- Continue finding model-based interpretations of numerical solvers.


