Backward and Inverse
Opportunities for Probnum in Machine Learning

Philipp Hennig

Heilbronn Workshop – 27 March 2022

Eberhard Karls Universität Tübingen

Faculty of Science
Department of Computer Science
Chair for the Methods of Machine Learning

Many thanks to our supporter and sponsors:

Max-Planck-Gesellschaft · erc · ellis · CyberValley · DFG · Federal Ministry of Education and Research · Baden-Württemberg Ministerium für Wissenschaft, Forschung und Kunst · Microsoft

The Men in Black

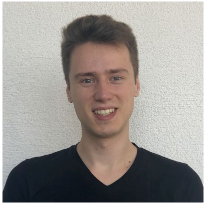
This talk contains work by many people, including many in the room today

Filip Tronarp

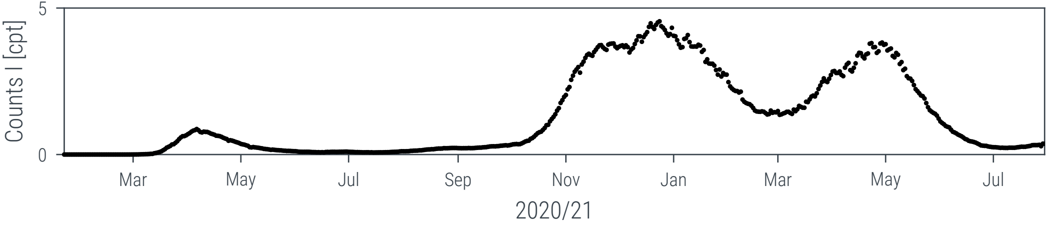
Nico Krämer

Nathanael Bosch

Jonathan Schmidt



Marvin Pförtner



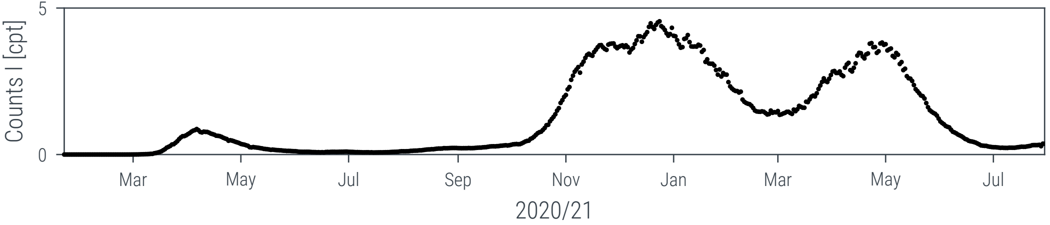
Is this

- a machine learning task? (regress on $I(t)$)
  doesn't work withouth mechanistic knowledge

A very 2021 inference task

Mixed Information Sources

Schmidt, Krämer, Hennig, 2021, NeurIPS 2021



Is this

- a machine learning task? (regress on $I(t)$)
  doesn't work withouth mechanistic knowledge
- a simulation problem? (solve SIRD ODE)
  we don't know β, though!
- an *inverse* problem (estimate β)
  we really care about $I(t)$, though!

$$\frac{d}{dt}\begin{bmatrix} S(t) \\ I(t) \\ R(t) \\ V(t) \\ D(t) \end{bmatrix} = \begin{bmatrix} -\beta(t)S(t)I(t)/P - v(t) \\ \beta(t)S(t)I(t)/P - \gamma I(t) - \eta I(t) \\ \gamma I(t) \\ v(t) \\ \eta I(t) \end{bmatrix}$$

What is an Inverse Problem?

Mechanistic Knowledge and mixed information sources

$$x'(t) = f(x(t), u(t)), \quad p(\mathbf{y} \mid x) = \prod_n^N \mathcal{N}(y_n; H(x(t_n)), \Sigma_n), \quad p(x) = \mathcal{GP}(m_x, k_x), \quad p(u) = \mathcal{GP}(m_u, k_u)$$

An *inverse problem* seems to be

- another word for an *inference* problem (inferring latent quantities from observations) (wikipedia).
- about inferring the objext x in $y = D(x)$, where D is a known operator (here: the ODE integral) from data y.

In both cases, it seems the tough part, arguably, is the ill-posedness. But the data y is already probabilistic, too!

Solving Inverse Problems with Backprop

automatic differentiation in simulation

$$\boldsymbol{u}^{(i)} = \boldsymbol{u}^{(i-1)} + \alpha \nabla_u L(u^{(i)})$$

`while` $L >$ `tol`

$$\hat{x} \leftarrow \texttt{ODEsolve}(f, u^{(i)})$$

$$L(u^{(i)}) = \log p(y \mid \hat{x}(u^{(i)}))$$

- Define some loss $L(u)$, e.g.

$$L(u) := \sum_i - \log p(y_i \mid \hat{x}(u)) = \sum_i (y_i - H(\hat{x}(u)))^2 + \text{const.}.$$

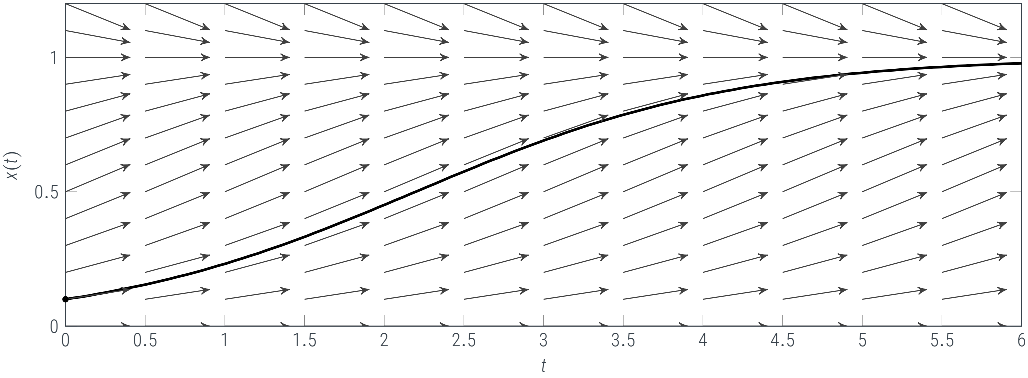
- Compute the gradient $\nabla_u L(u^{(i)})$ with automatic differentiation. Examples:
  - `numppyro` tutorial, using jax's dopri5
  - `Turing.jl` tutorial, using diffeq.jl solvers

Note how the user is discouraged from even thinking about the ODE solver.

The Probabilistic View on ODEs

[Schober, Duvenaud & P.H., 2014. Schober & P.H., 2016. Kersting & P.H., 2016, Tronarp, Kerstin, P.H., 2019, Bosch, Tronarp, P.H., 2021, ...]

$$x'(t) = f(x(t), t), \quad x(t_0) = x_0$$



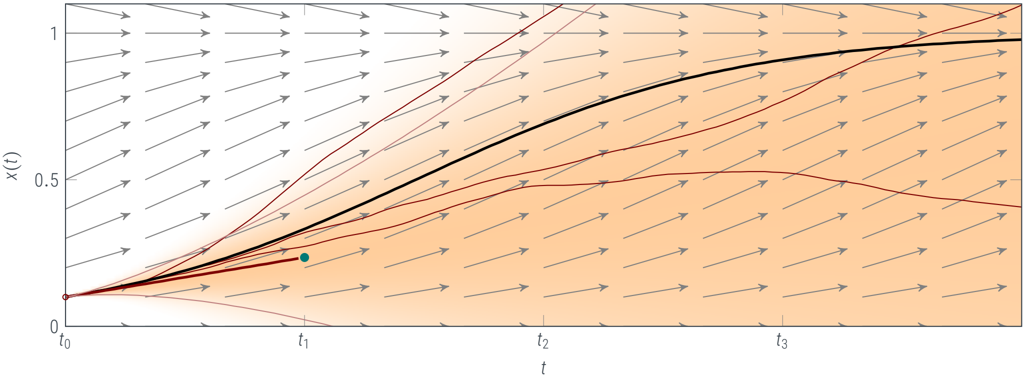

```
scipy.integrate.solve_ivp(f,t_span,x_0)   ⇒   probnum.diffeq.probsolve_ivp(f,t_span,x_0)
```

The Probabilistic View on ODEs

EBERHARD KARLS UNIVERSITÄT TÜBINGEN

[Schober, Duvenaud & P.H., 2014. Schober & P.H., 2016. Kersting & P.H., 2016, Tronarp, Kerstin, P.H., 2019, Bosch, Tronarp, P.H., 2021, ...]

$$x'(t) = f(x(t), t), \quad x(t_0) = x_0$$

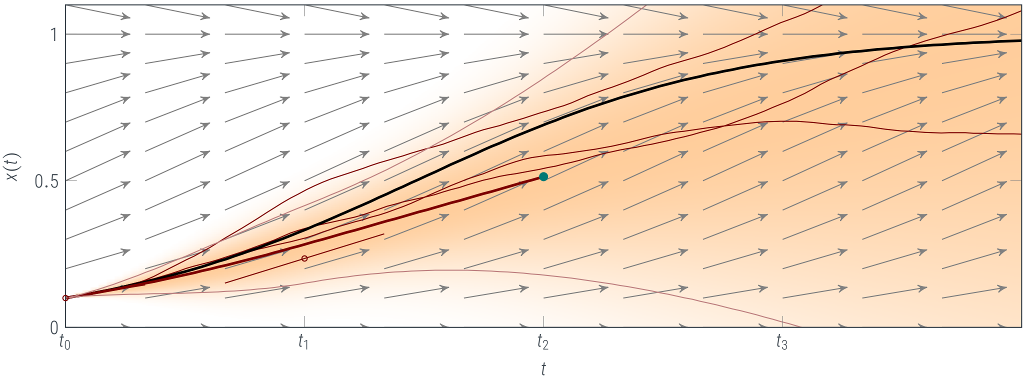


`scipy.integrate.solve_ivp(f,t_span,x_0)` $\Rightarrow$ `probnum.diffeq.probsolve_ivp(f,t_span,x_0)`

The Probabilistic View on ODEs

[Schober, Duvenaud & P.H., 2014. Schober & P.H., 2016. Kersting & P.H., 2016, Tronarp, Kerstin, P.H., 2019, Bosch, Tronarp, P.H., 2021, ...]

$$x'(t) = f(x(t), t), \quad x(t_0) = x_0$$

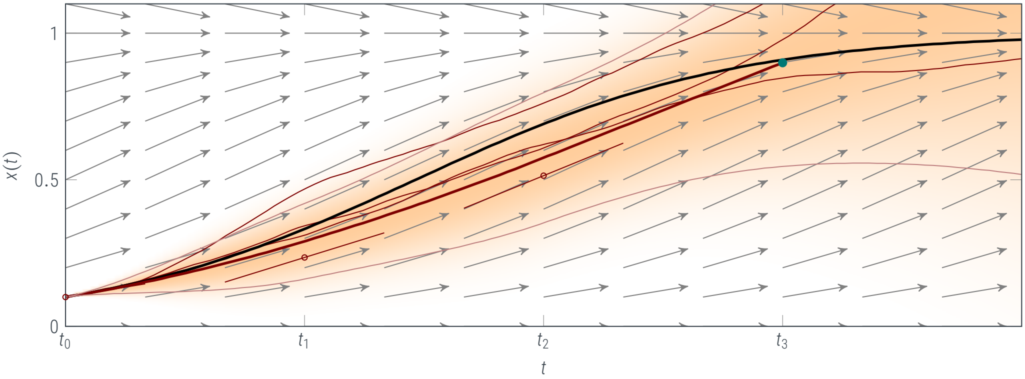


`scipy.integrate.solve_ivp(f,t_span,x_0)` $\Rightarrow$ `probnum.diffeq.probsolve_ivp(f,t_span,x_0)`

The Probabilistic View on ODEs

[Schober, Duvenaud & P.H., 2014. Schober & P.H., 2016. Kersting & P.H., 2016, Tronarp, Kerstin, P.H., 2019, Bosch, Tronarp, P.H., 2021, ...]

$$x'(t) = f(x(t), t), \quad x(t_0) = x_0$$

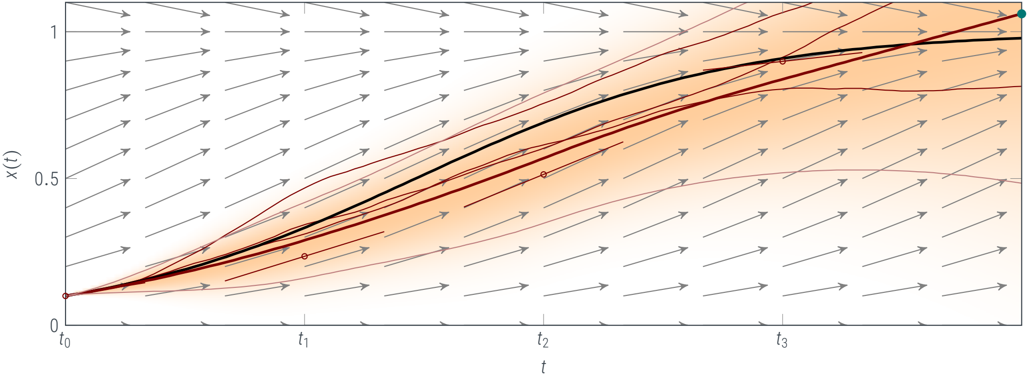


`scipy.integrate.solve_ivp(f,t_span,x_0)` $\Rightarrow$ `probnum.diffeq.probsolve_ivp(f,t_span,x_0)`

The Probabilistic View on ODEs

[Schober, Duvenaud & P.H., 2014. Schober & P.H., 2016. Kersting & P.H., 2016, Tronarp, Kerstin, P.H., 2019, Bosch, Tronarp, P.H., 2021, ...]

$$x'(t) = f(x(t), t), \quad x(t_0) = x_0$$

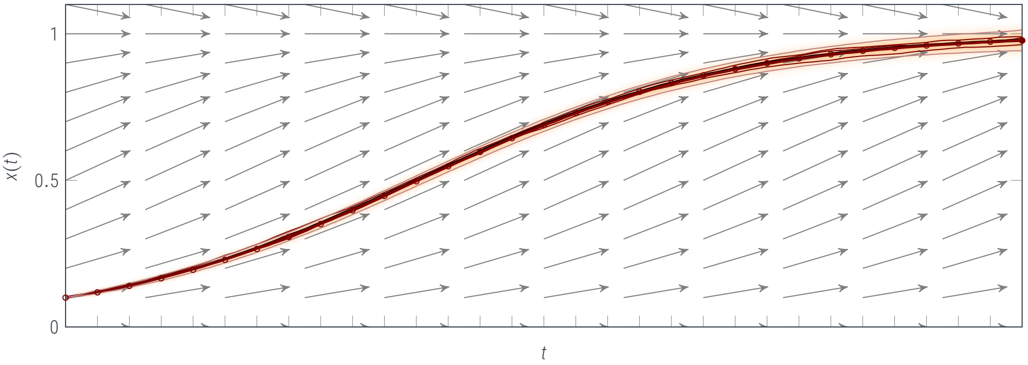


`scipy.integrate.solve_ivp(f,t_span,x_0)` $\Rightarrow$ `probnum.diffeq.probsolve_ivp(f,t_span,x_0)`

The Probabilistic View on ODEs

[Schober, Duvenaud & P.H., 2014. Schober & P.H., 2016. Kersting & P.H., 2016, Tronarp, Kerstin, P.H., 2019, Bosch, Tronarp, P.H., 2021, …]

$$x'(t) = f(x(t), t), \quad x(t_0) = x_0$$

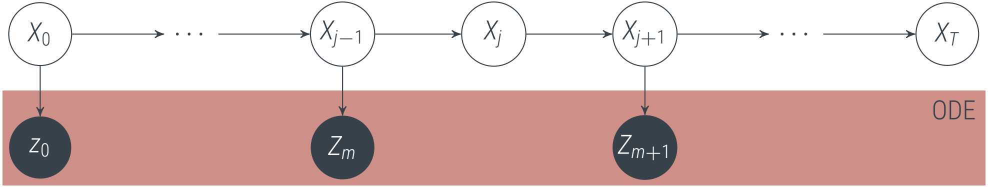


`scipy.integrate.solve_ivp(f,t_span,x_0)` $\Rightarrow$ `probnum.diffeq.probsolve_ivp(f,t_span,x_0)`

Simulation as Filtering

probabilistic ODE solvers can be realised as Kalman filters

Tronarp, Kersting, Särkkä, PH, Statistics & Computing **29**(6): 1297–1315



$$z_0 \mid X(t_0) \sim \delta(x(t_0) - x_0) \qquad Z_m \mid X(t_m) \sim \delta(X^{(1)}(t_m) - f(X^{(0)}(t_m)))$$

- Use a **tractable** (linear Gaussian) *stochastic* differential equation as a prior for the **intractable** solution of the *nonlinear ordinary* differential equation

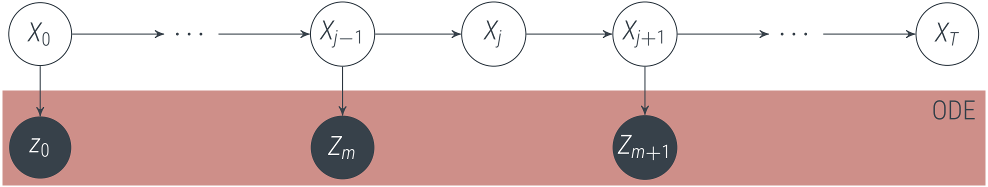
$$dX(t) = FX(t)\,dt + L\,dW(t) \qquad \text{with} \qquad X^{(i)}(t) = \frac{d^i}{dt^i}x(t), i = 1, \dots, \nu$$

- Consider *information operators* Z_i to link evaluations of the vector field f to x
- run the *extended Kalman filter (EKF)* to propagate uncertainty through f.

Simulation as Filtering

probabilistic ODE solvers can be realised as Kalman filters

Tronarp, Kersting, Särkkä, PH, Statistics & Computing **29**(6): 1297–1315



procedure $\textsc{ExtendedFilter}(m_{t-1}, P_{t-1}, A, Q, H, R, y)$

$m_t^- = Am_{t-1}$ // predictive meanm with $A = \int \exp(F(\Delta t))$

$P_t^- = AP_{t-1}A^\intercal + Q$ // predictive covariance, with $Q = \int_0^{\Delta t} e^{F\tau} LL^\intercal e^{F\tau} \, d\tau$

$r = y - Hm_t^-$ // residual, with $y = 0, H = \partial h_{10} / \partial X|_{X=m_t^-}$

$S = HP_t^- H^\intercal + R$ // innovation covariance

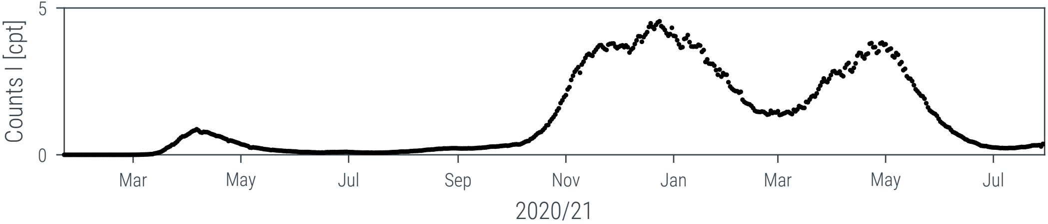
$K = P_t^- H^\intercal S^{-1}$ // gain

$m_t = m_t^- + Kr$ // updated mean

$P_t = (I - KH)P_t^-$ // updated covariance

return $(m_t, P_t), (m_t^-, P_t^-)$

end procedure

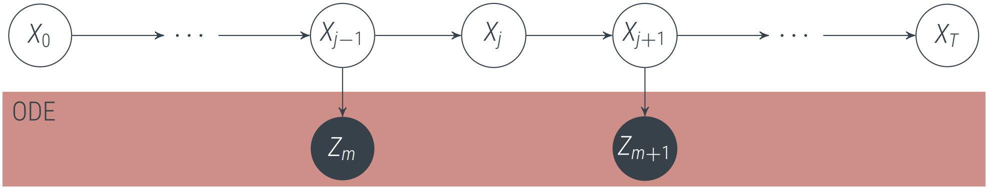


$$\frac{d}{dt}\begin{bmatrix} S(t) \\ I(t) \\ R(t) \\ V(t) \\ D(t) \end{bmatrix} = \begin{bmatrix} -\beta(t)S(t)I(t)/P - v(t) \\ \beta(t)S(t)I(t)/P - \gamma I(t) - \eta I(t) \\ \gamma I(t) \\ v(t) \\ \eta I(t) \end{bmatrix}$$

Not forward/inverse, but *mixed information*

blurring the boundaries of the black box

[Tronarp, Kersting, Särkkä, Hennig, 2019; Schmidt, Krämer, Hennig, , NeurIPS 2021]



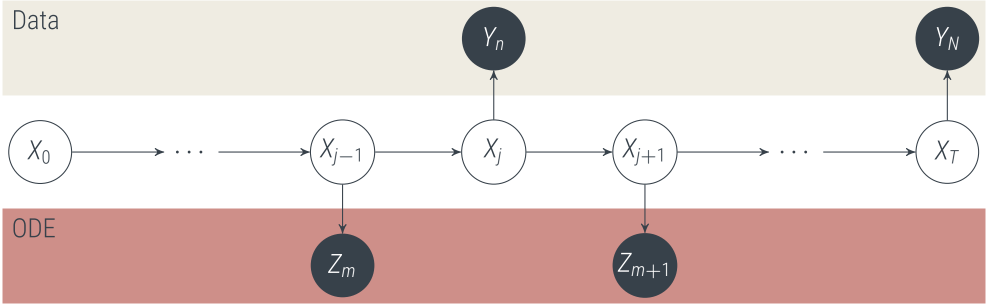
to solve ODE $\frac{d}{dt}x(t) = f(x(t), t)$, model with SDE $dX(t) = F_X X(t)\,dt + L_X dW_X(t)$ and observation model (information operator)

$$Z_m \mid X(t_m) \sim \delta(E_X^{(1)} X(t_m) - f(E_X^{(0)} X(t_m)))$$

Not forward/inverse, but *mixed information*

blurring the boundaries of the black box

[Tronarp, Kersting, Särkkä, Hennig, 2019; Schmidt, Krämer, Hennig, , NeurIPS 2021]



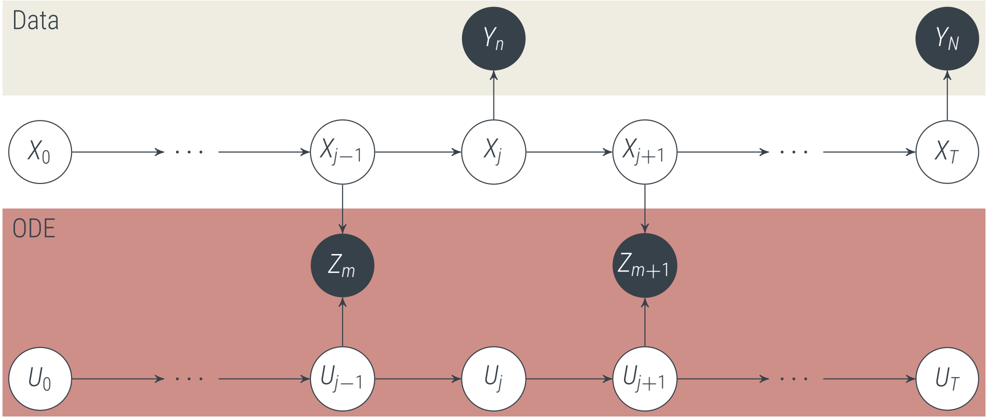
natively (within same "forward" solve) combine with physical observations of the trajectory

$$Y_n \mid X(t_n) \sim \mathcal{N}(HE_X^0 X(t_n), R)$$

Not forward/inverse, but *mixed information*

blurring the boundaries of the black box

[Tronarp, Kersting, Särkkä, Hennig, 2019; Schmidt, Krämer, Hennig, , NeurIPS 2021]



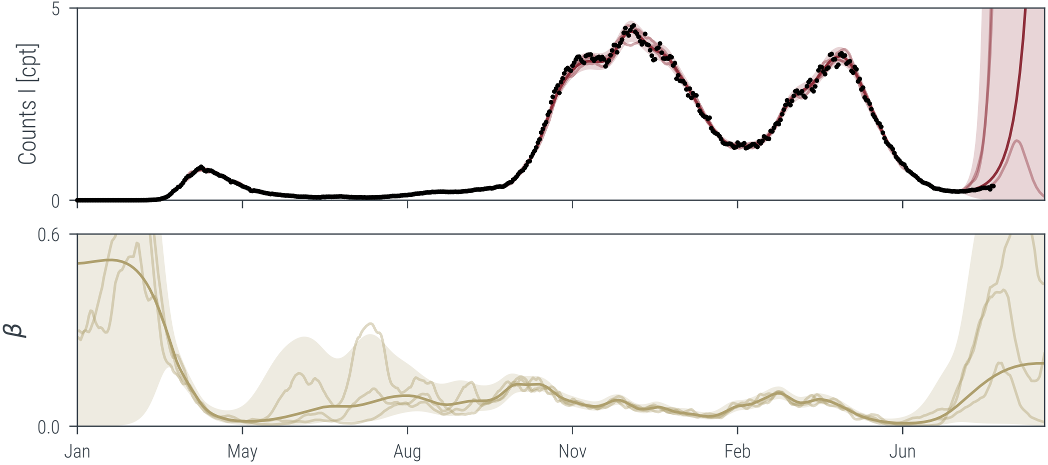
propagate uncertainty about ODE (e.g. from a latent force U) through the extended Kalman filter to solve $\frac{d}{dt}x(t) = f(x(t), u(t), t)$ with $dU(t) = F_U U(t)\, dt + L_U dW_U(t)$.

No more black box ODE solvers

Example: Covid modelling

EBERHARD KARLS UNIVERSITÄT TÜBINGEN

Schmidt, Krämer, Hennig, 2021, NeurIPS 2021

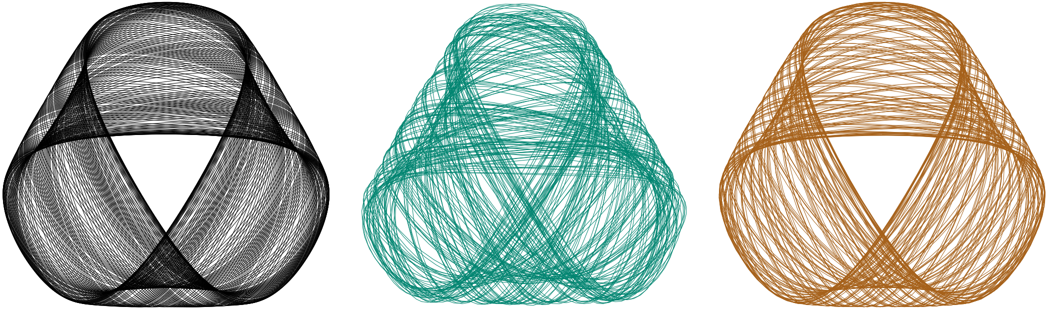


Addiotnal Information can be added, too

Information Operator for Hamiltonians and other conserved quantities

EBERHARD KARLS UNIVERSITÄT TÜBINGEN

Bosch, Tronarp, PH, AISTATS 2022 (arXiv 2110.10770)



Description	Equation	Information operator
First-order ODE	$\dot{y}(t) = f\left(y(t), t\right)$	$z(t, Y) := Y^{(1)} - f\left(Y^{(0)}, t\right)$
Second-order ODE	$\ddot{y}(t) = f\left(\dot{y}(t), y(t), t\right)$	$z(t, Y) := Y^{(2)} - f\left(Y^{(1)}, Y^{(0)}, t\right)$
Mass matrix DAE	$M\dot{y}(t) = f\left(y(t), t\right)$	$z(t, Y) := MY^{(1)} - f\left(Y^{(0)}, t\right)$
Invariances	$g\left(y(t), \dot{y}(t)\right) = 0$	$z(t, Y) := g\left(Y^{(0)}, Y^{(1)}\right)$
Chain rule	$\ddot{y}(t) = J_f\left(y(t)\right) \cdot \dot{y}(t)$	$z(t, Y) := Y^{(2)} - J_f\left(Y^{(0)}\right) \cdot Y^{(1)}$

Stationary Inverse Problems as GP Hyperparameter Inference

Infer the parameters θ of IVP ξ_θ measured with Gaussian noise at solution x

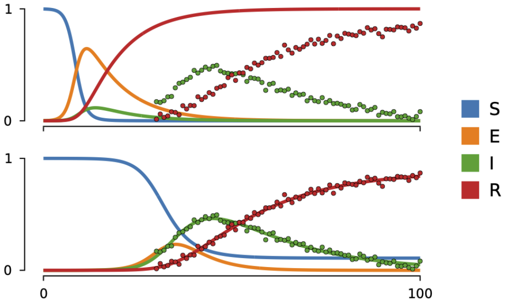
$$\frac{\mathrm{d}}{\mathrm{d}t}\xi_\theta(t) = f_\theta(\xi_\theta(t)), \qquad \phi_\theta(0) = x_0, \qquad p(\boldsymbol{y} \mid x) = \prod_i \mathcal{N}(y_i; H^\intercal x, R_\theta)$$

- We'd like to compute the marginal

$$p(\boldsymbol{y} \mid \theta) = \int p(\boldsymbol{y} \mid x)\delta(x - \xi_\theta)\,\mathrm{d}x$$

- Approximate δ with a Gaussian

$$\hat{p}(\boldsymbol{y} \mid \theta) = \int p(\boldsymbol{y} \mid x)\hat{\delta}_N(x - \xi_\theta)\,\mathrm{d}x$$

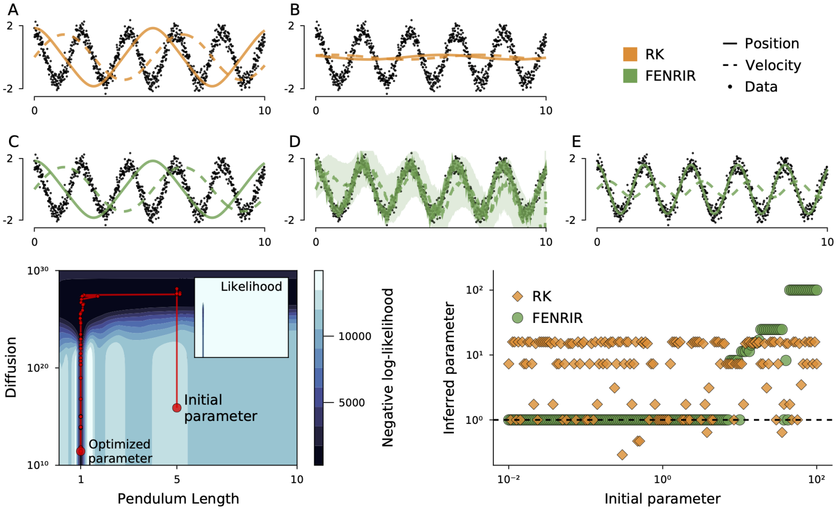


Prior Hyperparameters as Regularizers

Tronarp, Bosch, Hennig. *Fenrir: Physics-Enhanced Regression for Initial Value Problems*

EBERHARD KARLS UNIVERSITÄT TÜBINGEN

arXiv 2202.01287



EBERHARD KARLS
UNIVERSITÄT
TÜBINGEN

Summary

- *Propagation of Uncertainty* is great, but should not mislead us to keep the rigid structure of classical code
- instead, sometimes, *information* (the opposite of uncertainty) shouldn't be propagated, but *combined* efficiently
- because Probnum methods can deal with imprecise quantities natively, changing the order of the computation does not pose a conceptual problem for them. (That doesn't mean changing the order is always a good idea. But it's also not necessarily a bad idea).
- doing so can break the (artificial) separation between forward and inverse problems.

Re-casting computation as inference allows genuinely new, valuable functionality.

http://mml.inf.uni-tuebingen.de

Probabilistic Numerics – Computation as Machine Learning. P. Hennig, H. Kersting, M.A. Osborne, CUP, 2022

https://www.youtube.com/c/TübingenML

@PhilippHennig5

High-Dimensional ODEs/PDEs

Factorization assumptions allow scaling to millions of dimensions

Krämer, Bosch, Schmidt, PH, arXiv 2110.11812