Hybrid PDE-Data Modeling

Matthys Jacobus Steynberg

Friedrich-Alexander-Universität-Erlangen-Nürnberg

Friedrich-Alexander-Universität
DYNAMICS, CONTROL, MACHINE LEARNING AND NUMERICS

August 27, 2024

Joint work with Enrique Zuazua and Lorenzo Liverani

Outline

Outline

Data-Driven Modeling

Given some data, build a model for generalization and/or prediction

Data can be

1. Time-dependent (e.g. time series)
2. Dynamic (e.g. coming from sensors)
3. Irregularly spaced

$\Rightarrow$ Strong need for flexible architectures for data-driven modeling

Current Approaches

- **Fully Data-Driven.** NODEs, RNN, Reservoir Computing (RC) etc. $\rightsquigarrow$ No connection to physics, but very flexible
- **(Possibly hybrid) Physics-Based.** PINNs, Hamiltonian NN, Universal differential equations (UDE) etc. $\rightsquigarrow$ Require knowledge of the system, but are rigid and difficult to train

Problem Motivation

Suppose to have recorded some data

- We can construct a synthetic model with one of the approaches described above
- At the same time, if we assume that the data is the realization of a physical law depending on some structural parameters (diffusivity, Young modulus, Reynold number...), we might try to fit the parameters so that the solution is as close as possible to the observed data

$\rightsquigarrow$ But the two obtained models might be **very different** in general!

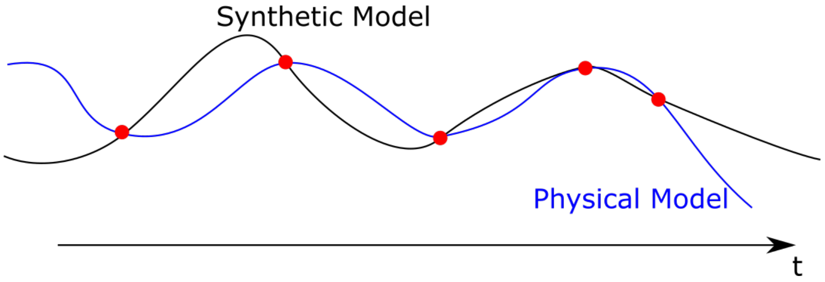
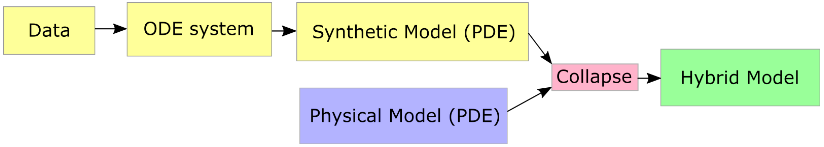


Figure: The two models perfectly interpolate the data points, but behave in completely different ways

In medio stat virtus $\rightsquigarrow$ The "best" model should be close to both the physical and the synthetic model $\Rightarrow$ Try to collapse the two into a single hybrid model

Outline

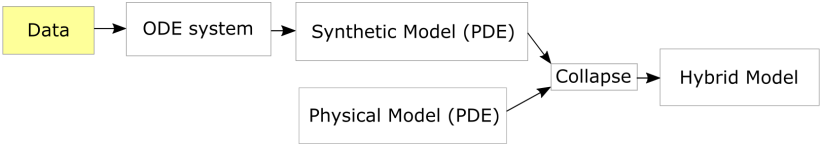
Strategy Overview



Advantages:

1. Highly adaptable: Data can be very rough, unbalanced in space or irregular in time
2. Physics based: The model is not purely data-driven, but exploits theoretical information

Data collection

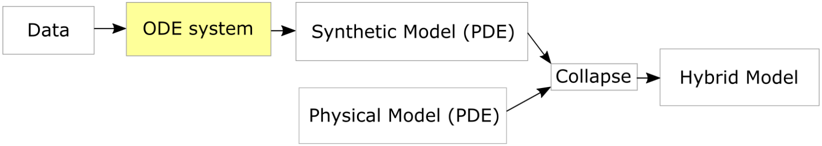


We suppose to have M sensors which can move in space and record data (e.g. a temperature) at N equispaced intervals in time. The j-th sensor records $u_j(t_i)$ at time t_i, so that the final form of the data is

$$\mathrm{Data}_j = (x_j(t_i), u_j(t_i))_{i=1}^{N}$$

for $j = 1, \ldots, M$

From discrete to continuous (time)

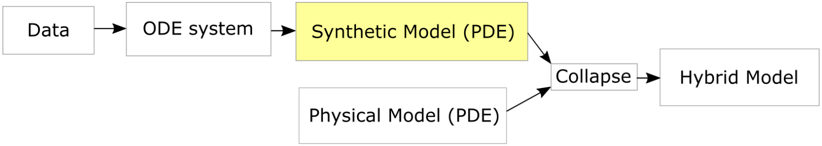


We build a NODE system

$$\begin{cases} \dot{\hat{x}}_j = f_j(\hat{x}_j, \hat{u}_j, t; \Theta) \\ \dot{\hat{u}}_j = g_j(\hat{x}_j, \hat{u}_j, t; \Theta) \end{cases}$$

to track the discrete data trajectories at every time t

From discrete to continuous (space)



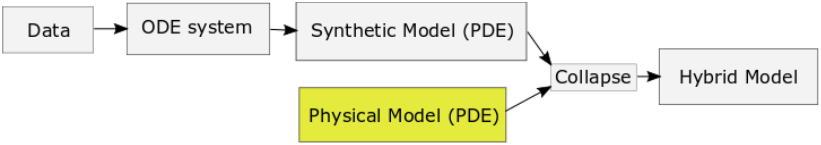
To interpolate at every space point, we exploit the connection between ODEs and Transport equations. We consider

$$\rho_t + f(t,x,v)\rho_x + g(t,x,v)\rho_v = 0.$$

This equation is linear and is reminiscent of kinetic models. We see the function $\rho(t,x,v)$ as a probability density on $\mathbb{R}$ for v. So we have

$$u(t,x) = \int_{\mathbb{R}} v\rho(t,x,v)dv$$

Physical model



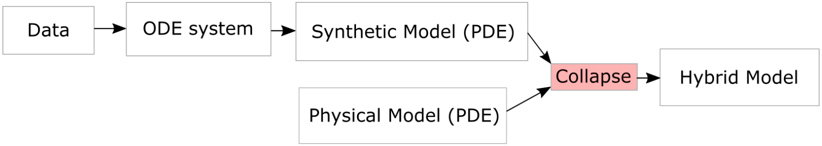
Assume that the underlying system is some PDE, for example a 1D heat equation with unknown diffusivity $d(x)$,

$$\begin{cases} u_t - \partial_x(d(x)\partial_x u) = 0, \quad x \in [0,1], t \in (0,T] \\ u(0,t) = u(1,t) = 0, \\ u(x,0) = u_0(x). \end{cases}$$

Discretizing this using finite elements with basis $\Phi = (\phi_1, \ldots, \phi_n)$,

$$M\dot{U}(t) + K(d)U(t) = 0,$$

Collapse



The aim of the collapse strategy is to minimize the gap between the physical model and the synthetic one by training both in tandem. Thus, the goal is to minimize a loss functional of the form

$$L(d, \Theta) = \alpha_1 L_{phy}(d) + \alpha_2 L_{syn}(\Theta) + \alpha_3 \mathrm{Int}(d, \Theta)$$

Loss connected to the physical model:

$$L_{phy}(d) = \frac{1}{M}\sum_{j=1}^{M}\int_0^T |\left\langle U(t), \Phi(x_j^D)\right\rangle - u_j^D(t)|^2 dt$$

Loss connected to the synthetic model:

$$L_{syn}(\Theta) = \frac{1}{M}\sum_{j=1}^{M}\int_0^T \left(|\hat{x}_j(t) - x_j^D(t)|^2 + |\hat{u}_j(t) - u_j^D(t)|^2\right) dt$$

The interaction functional for hybrid training:

$$\mathrm{Int}(d, \Theta) = \frac{1}{K}\sum_{j=1}^{K}\int_0^T |\left\langle U(t), \Phi(\hat{x}_j(t))\right\rangle - \hat{u}_j(t)|^2 dt.$$

Training Approach

To train both models, we will do gradient descent while calculating the gradients using the adjoint method.

$$d^{l+1}(x) = d^l(x) - \beta_d \frac{\partial L}{\partial d}$$
$$\Theta^{l+1} = \Theta^l - \beta_\Theta \frac{\partial L}{\partial \Theta}$$

where

$$\frac{\partial L}{\partial d(x)} = -\int_0^T \sum_{ij} \Lambda_i(t) U_j(t) \nabla \Phi_i(x) \cdot \nabla \Phi_j(x) dt$$
$$\frac{\partial L}{\partial \Theta} = -\sum_{j=1}^{M} \int_0^T \left(\mu_j(t)^T \frac{\partial f(x_j(t), u_j(t), t)}{\partial \Theta} + \nu_j(t)^T \frac{\partial g(x_j(t), u_j(t), t)}{\partial \Theta} \right) dt$$

The adjoint connected to the FE solution $U(t)$:

$$\dot{\Lambda}(t) = K^T(d)\Lambda(t) + \frac{2\alpha_1}{M}\sum_{j=1}^{M}\left(\left\langle U(t), \Phi(x_j^D(t))\right\rangle - u_j^D(t)\right)\Phi(x_j^D) + \frac{2\alpha_3}{K}\sum_{j=1}^{K}\left(\langle U(t), \Phi(\hat{x}_j(t))\rangle - \hat{u}_j(t)\right)\Phi(\hat{x}_j)$$

The adjoint connected to the NODE solution $\hat{x}_j$:

$$\dot{\mu}_j(t) = \frac{2\alpha_2}{M}\left(\hat{x}_j(t) - x_j^D(t)\right) + \frac{2\alpha_3}{K}\left(\langle U(t), \Phi(\hat{x}_j(t))\rangle - u_j(t)\right)\left\langle U(t), \frac{\partial \Phi(\hat{x}_j)}{\partial \hat{x}_j}\right\rangle - \mu_j(t)^T\frac{\partial f(x_j(t), u_j(t), t)}{\partial x_j(t)} - \nu_j(t)^T\frac{\partial g(\hat{x}_j(t), \hat{u}_j(t), t)}{\partial u_j(t)}$$

The adjoint connected to the NODE solution $\hat{u}_j$:

$$\dot{\nu}_j(t) = \frac{2\alpha_2}{M}\left(\hat{u}_j(t) - u_j^D(t)\right) + \frac{2\alpha_3}{K}\left(\langle U(t), \Phi(\hat{x}_j)\rangle - \hat{u}_j\right) + \mu_j(t)^T\frac{\partial f(x_j(t), u_j(t), t)}{\partial x_j(t)} + \nu_j(t)^T\frac{\partial g(\hat{x}_j(t), \hat{u}_j(t), t)}{\partial u_j(t)}$$

Outline

Training data

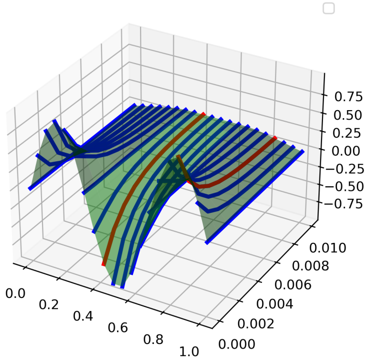


Figure: Data from the exact model

Training performance

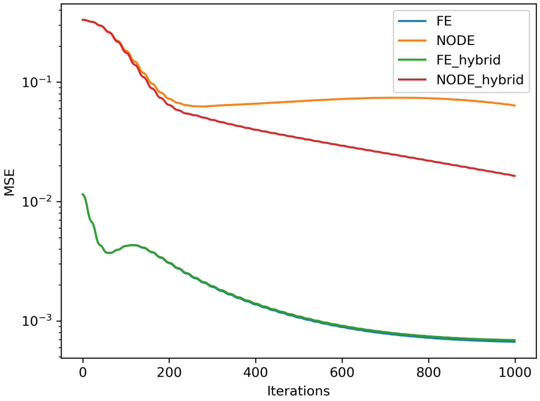
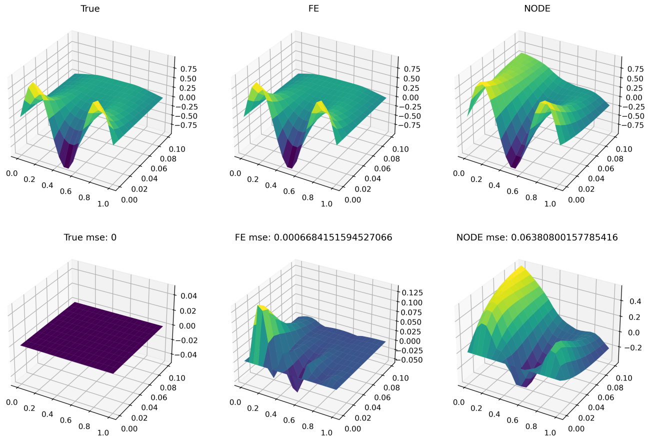
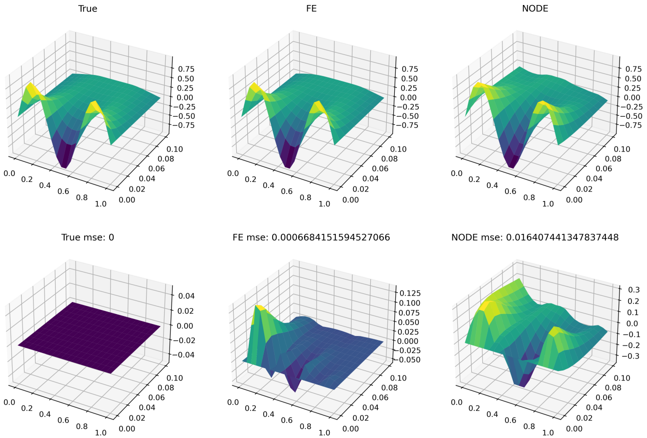


Figure: MSE over the entire domain.

Separate models



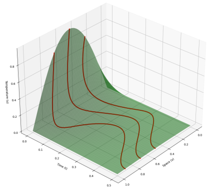
Hybrid models



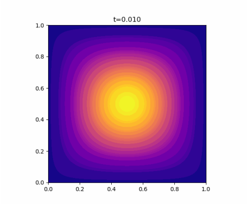
Outline

Further work

- Extend the code for general sensor trajectories.



- Extend the code for more complex physical models.



  - Understand what information is contained in the "gap".
  - Understand the implications of the interaction functional in the loss function.

Remarks and further work

- This formulation can be applied to hybridize any two data-driven and physics-based models.
- It can be used NODE training from sparse data by assuming a particular underlying system, similar to the idea behind PINNs and Universal Differential Equations.
- Our goal would be to extend this for parameter/flux identification.

Inverse coefficient problem

Nonlinear Diffusion in 2D/3D

$$\begin{cases} u_t - \nabla \cdot (\Phi(\theta, \nabla u)) = g, \\ + \text{ Initial and Boundary conditions} \end{cases}$$

1. Porous media equation
2. Image processing

Flux Identification in 1D Conservation Laws

$$u_t + (f(u))_x = 0$$

Conclusions

- Introduction of the collapse strategy by considering the connection between ODEs and PDEs using the transport equation.
- The implementation for fixed-position sensors on a heat-type equation.
- Planned further work and possible applications including flux-identification and nonlinear diffusion.

Thank you for listening!