Data assimilation methods with Neural Galerkin schemes

Joubine Aghili, Joy Atokple, Marie Billaud-Friess, Guillaume Garnier, Olga Mula, Norbert Tognon

January 27, 2024

Outline of the presentation

The Forward Problem

The Forward Problem

We use methods from [Bruna et al., 2022].

$\mathcal{X} \subset \mathbb{R}^d$

$$\begin{cases} \partial_t u(t,x) &= f(t,x,u), & (t,x) \in [0,\infty) \times \mathcal{X}, \\ u(0,x) &= u_0(x) & x \in \mathcal{X}, \end{cases}$$

If $u_0 \in L^2(x) \implies u(t) \in L^2(x)$

Examples:

- Advection-diffusion-reaction equation

$$f(t,x,u) = b(t,x) \cdot \nabla u + a(t,x) : \nabla\nabla u + G(t,x,u).$$

- KdV equation

$$f(t,x,u) = -\partial_x^3 u(t,x) - 6u(t,x)\partial_x u(t,x).$$

The forward problem

Goal : Find an approximation $\widetilde{u}$ of the solution u under the form

$$\boxed{\widetilde{u}(t,x) = U(\theta(t),x), \quad t>0, \ \ x \in \mathcal{X}, \ \ \theta \in \Theta.}$$

Idea: If $\widetilde{u} \approx u$, then

$$r_t(\theta,\eta,x) := \nabla_\theta \mathrm{U}(\theta(t),x)^T \cdot \eta - f(t,x,U(\theta)), \ \ \forall x \in \mathcal{X}.$$

Minimize: We search for a good $\theta(t) \in \Theta$ such that for all $t > 0$

$$\dot{\theta}(t) \in \arg\min_{\eta \in \dot{\Theta}} J_t(\theta,\eta), \qquad \text{where}$$

$$J_t(\theta,\eta) := \int_{\mathcal{X}} |r_t(\theta,\eta,x)|^2 d\nu(x).$$

The forward problem

Key point : The Euler–Lagrange Equation

$$\boxed{\nabla_\eta J_t(\theta(t), \eta) = 0}.$$

This equation can be written as a system of ODEs

$$\boxed{\begin{cases} M_\theta(t)\dot{\theta}(t) = F_\theta(t, \theta), \ t > 0 \\ \theta(0) = \theta_0, \end{cases}}$$

where

$$\theta_0 \in \arg\min_{\theta \in \Theta} \int_{\mathcal{X}} |u_0(x) - U(\theta, x)|^2 \, d\nu(x),$$

$$M_\theta(t) := \int_{\mathcal{X}} \nabla_\theta U(\theta, x)^T \cdot \nabla_\theta U(\theta, x) d\nu(x),$$

$$F_\theta(t, \theta) := \int_{\mathcal{X}} \nabla_\theta U(\theta, x) f(t, x, U(\theta)) d\nu(x).$$

Main consequence : We transform the PDE as an ODE on θ.

The forward problem

We solve the KdV equation given by :

$$\partial_t u = -\partial_x^3 u - 6u\partial_x u, \quad t \in [0,4],\ x \in [-20,40],$$

where $u(0,x)$ is given.
Solving the ODE system requires dealing with four issues:

- Estimating the operators $M(\theta)$ and $F(t,\theta)$ after proper specification of the measure ν: For giving samples $\{x_j\}_{j=1}^n$ from uniform probability measure, we get

$$M(\theta) \approx \frac{1}{n}\sum_{j=1}^{n} \nabla_\theta U(\theta,x_j) \cdot \nabla_\theta U(\theta,x_j)^T,$$

$$F(t,\theta) \approx \frac{1}{n}\sum_{j=1}^{n} \nabla_\theta U(\theta,x_j) f(t,x_j,U(\theta)).$$

The forward problem

- Designing a discrete time-integrator:
  - Fixed point time integrator: RK4,Euler Methods,...
  - Adaptative method: RK45, DOPRI5,...
- Choosing the parametrization $U(\theta)$: the neural network architecture, **Shallow Network** (One-hidden-layer)

$$U(\theta, x) = \sum_{i=1}^{m} c_i \varphi(x, \omega_i, b_i),$$

where

$$\varphi(x, \omega, b) = e^{-\omega^2 |x-b|^2},$$

and the parameter $\theta = \{(c_i, w_i, b_i\}_{i=1}^{m}$.

- Choosing the Python Librairy: Jax, Pytorch, Tensorflow, ...

Results in Pytorch

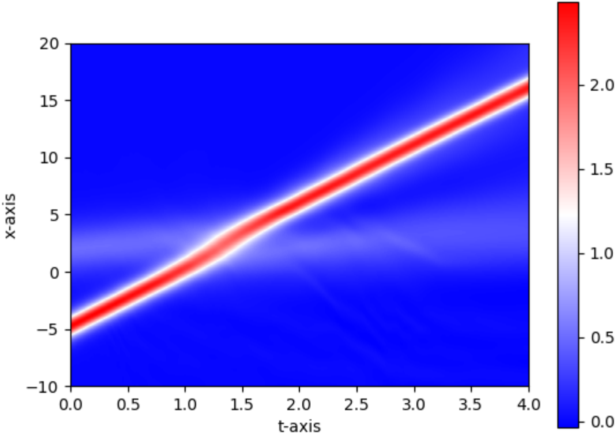

Figure: Solution to the KdV equation

Results in Pytorch

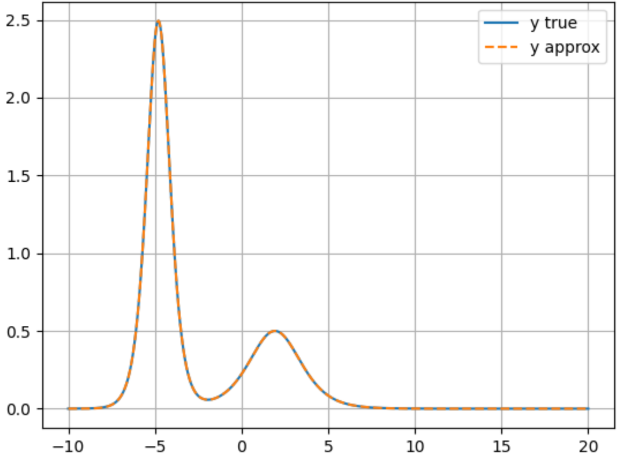

Figure: Solution to the KdV equation in $t = 0$.

Results in Pytorch

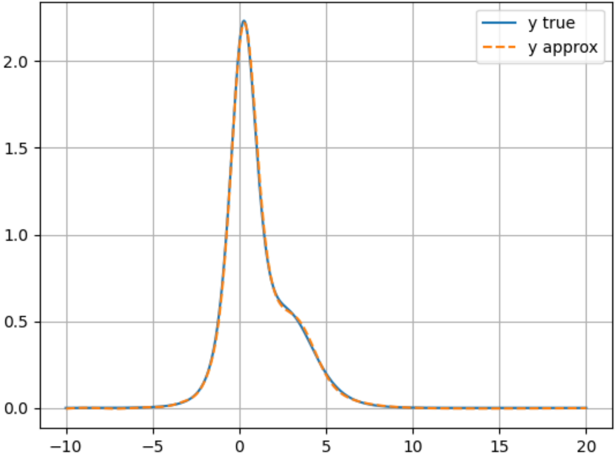

Figure: Solution to the KdV equation in $t = 1.0$.

Results in Pytorch

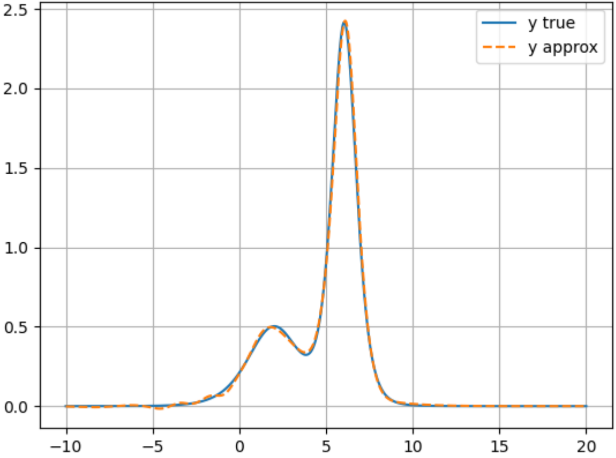

Figure: Solution to the KdV equation in $t = 2.0$.

Results in Pytorch

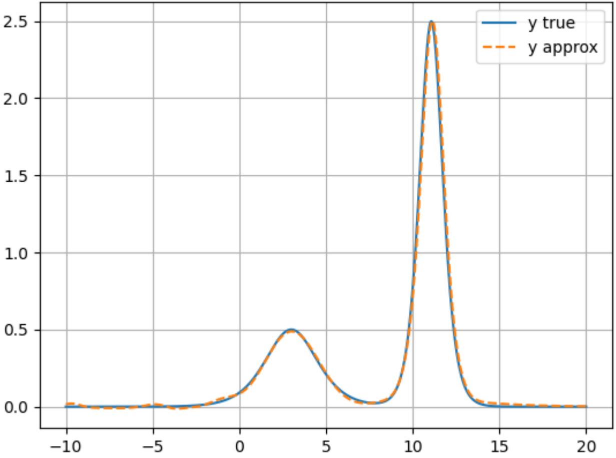

Figure: Solution to the KdV equation in $t = 3.0$.

Results in Pytorch

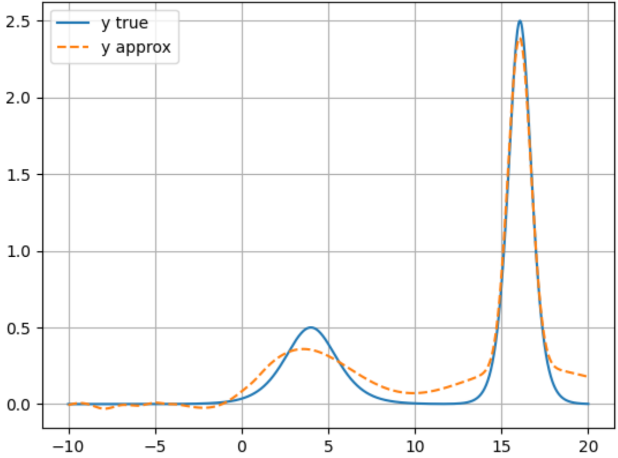

Figure: Solution to the KdV equation in $t = 4.0$.

The Inverse Problem

Inverse Problems- Method 1

Assume that the function f in (1) depends on a time-dependent parameter denoted $\mu(t) \in \mathcal{P} \subset \mathbb{R}^p$.

$$\begin{cases} \partial_t u(t,x) = f(t,x,u,\mu), & (t,x) \in [0,\infty) \times \mathcal{X}, \\ u(0,x) = u_0(x) & x \in \mathcal{X}. \end{cases} \quad (1)$$

Example: Parametric KdV equation

$$\partial_t u(t,x) = -\partial_x^3 u(t,x) - \mu(t) u(t,x) \partial_x u(t,x).$$

Inverse Problems- Method 1

Assumption: We do not know $\mu(t)$. Instead, we are given observations

$$\boxed{\dot{y}_i(t) = \dot{u}(t, x_i) \quad i = 1, \cdots, m}$$

We now consider the ansatz

$$\boxed{\widetilde{u}_\mu(t, x) = \mathrm{U}(\theta(t), x, \mu(t)), x \in \mathcal{X}}$$

We search for the derivative of the parameters $\theta(t)$ and $\mu(t)$ such that for all $t > 0$

$$\boxed{(\dot{\theta}(t), \dot{\mu}(t)) \in \underset{(\eta,\xi)\in\dot{\Theta}\times\dot{\mathcal{P}}}{\arg\min} \; G(t, \eta, \xi)}.$$

Inverse Problems- Method 1

The functional $G : [0, \infty) \times \dot{\Theta} \times \dot{\mathcal{P}} \to \mathbb{R}$ is defined as

$$G(t, \eta, \xi) := \int_{\mathcal{X}} \left| \nabla_\theta U(\theta, x, \mu)^T \cdot \eta + \nabla_\mu U(\theta, x, \mu)^T \xi - f(t, x, U(\theta), \mu) \right|^2 d\nu(x) + \lambda \sum_{i=1}^{m} \left| \dot{y}_i(t) - \nabla_\theta U(\theta, x, \mu)^T \cdot \eta - \nabla_\mu U(\theta, x, \mu)^T \cdot \xi \right|^2 .$$

Key point : The Euler–Lagrange Equation

$$\boxed{\begin{cases} \nabla_\eta G(t, \dot{\theta}, \dot{\mu}) = 0 \\ \nabla_\xi G(t, \dot{\theta}, \dot{\mu}) = 0 \end{cases}}$$

Inverse Problems- Method 1

This leads to a system of ODEs of size n+p

$$\begin{pmatrix} M_{\theta\theta} + \lambda M_{\theta\theta}^{X_m} & M_{\theta\mu} \\ M_{\theta\mu}^T & M_{\mu\mu} + \lambda M_{\mu\mu}^{X_m} \end{pmatrix} \begin{pmatrix} \dot{\theta} \\ \dot{\mu} \end{pmatrix} = \begin{pmatrix} F_\theta + \lambda \sum_{i=1}^{m} \dot{y_i}(t) \cdot \nabla_\theta U(\theta, x_i, \mu) \\ F_\mu + \lambda \sum_{i=1}^{m} \dot{y_i}(t) \nabla_\mu U(\theta, x_i, \mu) \end{pmatrix}$$

- the matrix in $\mathbb{R}^{n\times n}$ defined by

$$M_{\theta\theta}(t) = \int_{\mathcal{X}} \nabla_\theta \mathrm{U}(\theta, x, \mu) \cdot \nabla_\theta \mathrm{U}(\theta, x, \mu)^T d\nu(x)$$

and similarly for $M_{\mu\mu}(t)$ in $\mathbb{R}^{p\times p}$,

Inverse Problems- Method 1

- the matrix in $\mathbb{R}^{n\times n}$ defined by

$$M_{\theta\theta}^{X_m}(t) = \sum_{i=1}^{m} \nabla_\theta \mathrm{U}(\theta, x_i, \mu) \cdot \nabla_\theta \mathrm{U}(\theta, x_i, \mu)^T$$

  and similarly for $M_{\mu\mu}^{X_m}(t)$ in $\mathbb{R}^{p\times p}$,
- the matrix in $\mathbb{R}^{n\times p}$ defined by

$$M_{\theta\mu}(t) = \int_{\mathcal{X}} \nabla_\mu \mathrm{U}(\theta, x, \mu) \cdot \nabla_\theta \mathrm{U}(\theta, x, \mu)^T d\nu(x),$$

- and the vector in $\mathbb{R}^p$ defined by

$$F_\theta(t) = \int_{\mathcal{X}} \nabla_\theta \mathrm{U}(\theta, x, \mu) f(t, x, \mathrm{U}(\theta, \mu), \mu) d\nu(x),$$

  and similarly for $F_\mu(t)$ in $\mathbb{R}^p$.

Difficulties of Method 1

- The problem is ill-defined.
- How to add the $\mu(t)$ inside the Neural Network ?
- The initialisation is not clear.

Inverse Problems-Method 2

As an alternative to our first ansatz $U(\theta(t), x, \mu(t))$, we consider an approximation of $u(t)$ of the form

$$\boxed{\widetilde{u}_\mu(t,x) = U(\theta(t,\mu(t)),x), x \in \mathcal{X}} \tag{2}$$

$$J(t,\xi,\eta_1,\eta_2) = \frac{1}{2}\int_{\mathcal{X}} |\nabla_\theta U(\theta(t,\mu),x)\cdot(\eta_1+\xi\cdot\eta_2)) - f(t,x,U(\theta(t,\mu),\mu))|\, d\nu(x) + \frac{\lambda}{2}\sum_{i=1}^{m} |\dot{y}_i(t) - \nabla_\theta U(\theta(t,\mu),x)\cdot(\eta_1+\xi\cdot\eta_2))|^2$$

The second method

$$\frac{d\theta}{dt}(t) = \eta_1(t, \mu(t)) + \xi(t)\eta_2(t, \mu(t)) \tag{3}$$

$$\frac{d}{dt}\theta(t) \in \underset{\xi,\eta_1,\eta_2}{\arg\min}\, J(t, \xi, \eta_1, \eta_2) \tag{4}$$

(ξ, η_1, η_2) satisfy the following system of equations

$$\begin{aligned} \nabla_\xi J(t, \xi, \eta_1, \eta_2) &= 0 \\ \nabla_{\eta_1} J(t, \xi, \eta_1, \eta_2) &= 0 \\ \nabla_{\eta_2} J(t, \xi, \eta_1, \eta_2) &= 0 \end{aligned}$$

Difficulties of Method 2

- Results in a system of nonlinear PDEs

Inverse Problems-Method 3

Idea: Compute the most probable parameters μ w.r.t the data, and move in the direction of the derivative.

$$\begin{cases} M(\theta)\dot{\theta} = F(t, \theta, \mu) \\ \mu(t) \in \arg\min_{\mu} \left\{ \sum_{i=1}^{n} |f(t, x_i, U(\theta, x_i), \mu) - \dot{y}_i(t)|^2 \right\} \end{cases}$$

Numerical Results

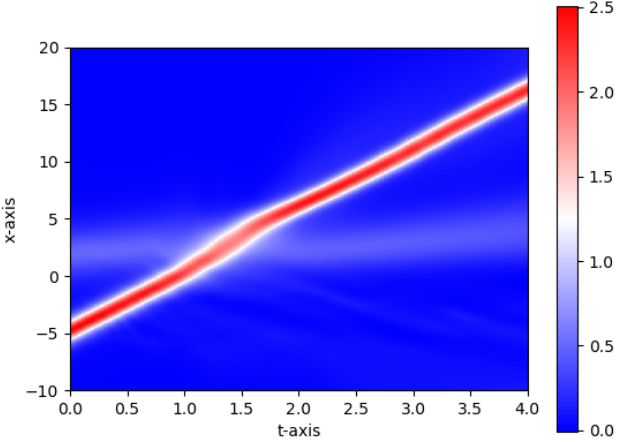

Figure: Inverse problem with 100 sensors, uniformly located

Numerical Results

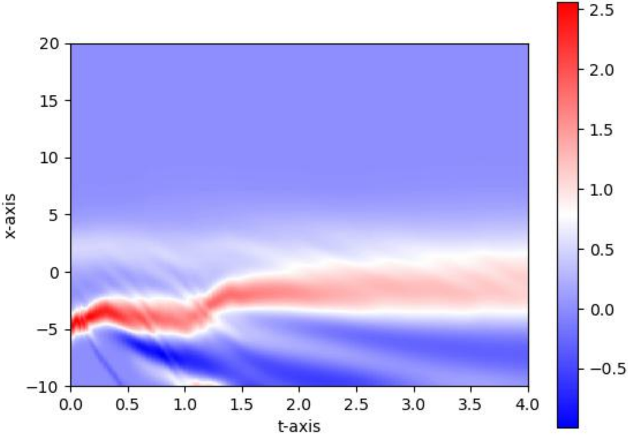

Figure: Inverse problem with 10 sensors, uniformly located

Numerical Results

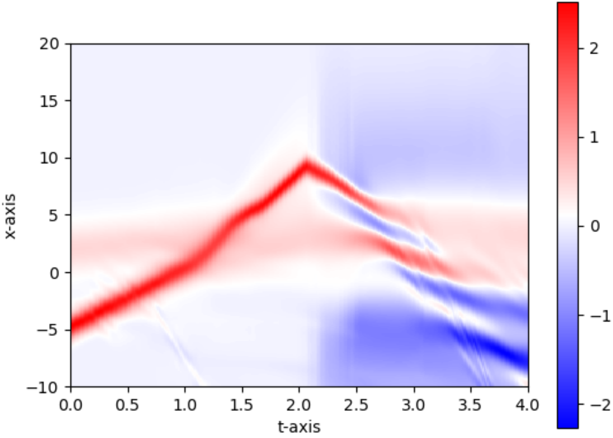

Figure: Inverse problem with 10 sensors, uniformly located between -5 and 0

Numerical Results

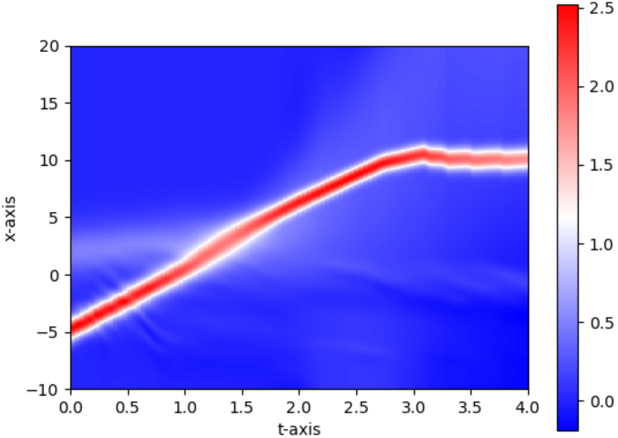

Figure: Inverse problem with 10 sensors, moving with the solution

Next Steps

Next Steps

- KdV 1D with varying velocity
- Allen Cahn 1D
- Adaptation of code to multiple dimensions
- Find a good strategy to solve the coupled PDE/ODE problem of the second inverse method.
- Find a data-driven strategy to make sensors move

Bibliography I

Bruna, J., Peherstorfer, B., and Vanden-Eijnden, E. (2022).
Neural galerkin scheme with active learning for high-dimensional evolution equations.
arXiv preprint arXiv:2203.01360.