Universität Bremen

Center for Industrial Mathematics (ZeTeM)

Faculty 03

Mathematics / Computer science

Introduction to TorchPhysics

Parameter identification problems

Janek Gödeke, Nick Heilenkötter, Tom Freudenberg

Renningen, 21.11.2025

Parameter Identification

So far: Solved ODEs/PDEs for given parameters

- Given scalar $c \in \mathbb{R}$, find solution u: $\partial_t^2 u(t,x) = c \cdot \partial_x^2 u(t,x)$

Now: Parameter-Identification

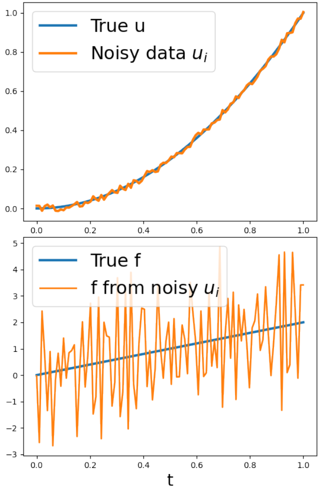
- Given solution u, find parameter c, or parameter function $f(x)$
- Solution given as noisy measurements $(x_1, u_1), ..., (x_n, u_n)$:

$$|u_i - u(x_i)| \lesssim \delta \quad \text{(noise-level } \delta).$$

Parameter Identification - Ill-Posedness

Caution with naive reconstruction strategies, e.g.:

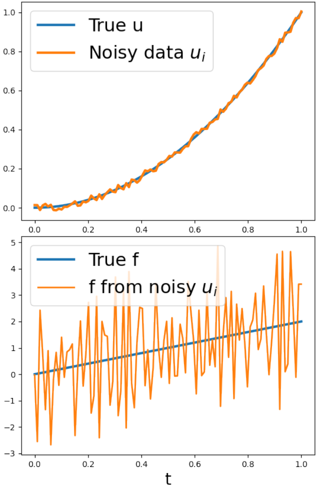
- Difference quotients for $\frac{d}{dt}u(t) = f(t)$?
- Noise in data u_i can get amplified!
  (Fig.: 2.4 % noise on u_i)



Parameter Identification - Ill-Posedness

Caution with naive reconstruction strategies, e.g.:

- Difference quotients for $\frac{d}{dt}u(t) = f(t)$?
- Noise in data u_i can get amplified!
  (Fig.: 2.4 % noise on u_i)
- Need noise-resilient reconstruction strategies



Parameter Identification with PINNs[1]

Find $c \in \mathbb{R}$ of wave equation:

$$\begin{cases} \partial_t^2 u = c\,\partial_x^2 u, & \text{in } I_x \times I_t, \\ u = 0 & \text{in } \partial I_x \times I_t, \\ \partial_t u(\cdot, 0) = 0 & \text{in } I_x, \\ u(\cdot, 0) = \sin(x) & \text{in } I_x. \end{cases}$$

Given: Data (t_i, x_i, u_i)

[1] Raissi et al., *PINNs: A deep learning framework for solving forward and inverse problems (...)*, 2019

Parameter Identification with PINNs[1]

Find $c \in \mathbb{R}$ of wave equation:

$$\begin{cases} \partial_t^2 u = c\,\partial_x^2 u, & \text{in } I_x \times I_t, \\ u = 0 & \text{in } \partial I_x \times I_t, \\ \partial_t u(\cdot, 0) = 0 & \text{in } I_x, \\ u(\cdot, 0) = \sin(x) & \text{in } I_x. \end{cases}$$

Given: Data (t_i, x_i, u_i)

- Trainable parameter c

[1] Raissi et al., *PINNs: A deep learning framework for solving forward and inverse problems (...)*, 2019

Parameter Identification with PINNs[1]

Find $c \in \mathbb{R}$ of wave equation:

$$
\begin{cases}
\partial_t^2 u = c\,\partial_x^2 u, & \text{in } I_x \times I_t, \\
u = 0 & \text{in } \partial I_x \times I_t, \\
\partial_t u(\cdot, 0) = 0 & \text{in } I_x, \\
u(\cdot, 0) = \sin(x) & \text{in } I_x.
\end{cases}
$$

Given: Data (t_i, x_i, u_i)

- Trainable parameter c
- Train NN u_θ with data loss

$$
\textbf{Data loss} = \frac{1}{N}\sum_{i=1}^{N} \left| u_\theta(t_i, x_i) - u_i \right|^2
$$

[1] Raissi et al., *PINNs: A deep learning framework for solving forward and inverse problems (...)*, 2019

Parameter Identification with PINNs[1]

Find $c \in \mathbb{R}$ of wave equation:

$$\begin{cases} \partial_t^2 u = c\,\partial_x^2 u, & \text{in } I_x \times I_t, \\ u = 0 & \text{in } \partial I_x \times I_t, \\ \partial_t u(\cdot,0) = 0 & \text{in } I_x, \\ u(\cdot,0) = \sin(x) & \text{in } I_x. \end{cases}$$

Given: Data (t_i, x_i, u_i)

- Trainable parameter c
- Train NN u_θ with data loss
- Train c (and u_θ) with PDE loss

$$\textbf{PDE loss} = \frac{1}{M}\sum_{j=1}^{M} \left| \partial_t^2 u_\theta(\tilde{t}_j, \tilde{x}_j) - c\,\partial_x^2 u_\theta(\tilde{t}_j, \tilde{x}_j) \right|^2$$

[1] Raissi et al., *PINNs: A deep learning framework for solving forward and inverse problems (...)*, 2019

Parameter Identification with PINNs[1]

Find $c \in \mathbb{R}$ of wave equation:

$$\begin{cases} \partial_t^2 u = c\, \partial_x^2 u, & \text{in } I_x \times I_t, \\ u = 0 & \text{in } \partial I_x \times I_t, \\ \partial_t u(\cdot, 0) = 0 & \text{in } I_x, \\ u(\cdot, 0) = \sin(x) & \text{in } I_x. \end{cases}$$

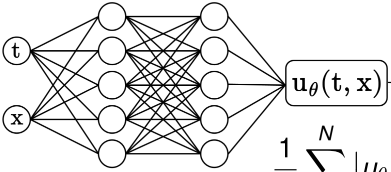
Given: Data (t_i, x_i, u_i)

- Trainable parameter c
- Train NN u_θ with data loss
- Train c (and u_θ) with PDE loss
- Train simultaneously

Data loss + PDE loss $= \frac{1}{N}\sum_{i=1}^{N} \left| u_\theta(t_i, x_i) - u_i \right|^2 + \frac{1}{M}\sum_{j=1}^{M} \left| \partial_t^2 u_\theta(\tilde{t}_j, \tilde{x}_j) - c\, \partial_x^2 u_\theta(\tilde{t}_j, \tilde{x}_j) \right|^2$

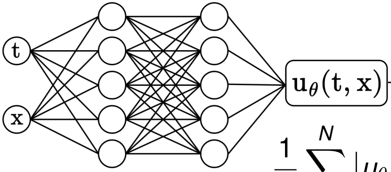
[1] Raissi et al., *PINNs: A deep learning framework for solving forward and inverse problems (...)*, 2019

Identify Parameter with TorchPhysics



$$\frac{1}{N}\sum_{i=1}^{N}\left|u_\theta(t_i, x_i) - u_i\right|^2 + \frac{1}{M}\sum_{j=1}^{M}\left|\partial_t^2 u_\theta(\tilde{t}_j, \tilde{x}_j) - c\,\partial_x^2 u_\theta(\tilde{t}_j, \tilde{x}_j)\right|^2$$

Identify Parameter with TorchPhysics

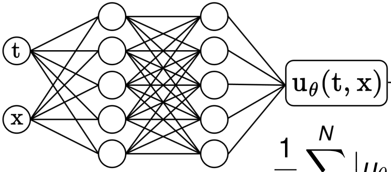


$$\frac{1}{N}\sum_{i=1}^{N}\left|u_\theta(t_i, x_i) - u_i\right|^2 + \frac{1}{M}\sum_{j=1}^{M}\left|\partial_t^2 u_\theta(\tilde{t}_j, \tilde{x}_j) - {\color{red}c}\,\partial_x^2 u_\theta(\tilde{t}_j, \tilde{x}_j)\right|^2$$

Implementation of **data condition**:

```
data_condition = tp.conditions.DataCondition(module=model_u,
                                             dataloader=data_loader,
                                             norm=2)
```

Identify Parameter with TorchPhysics



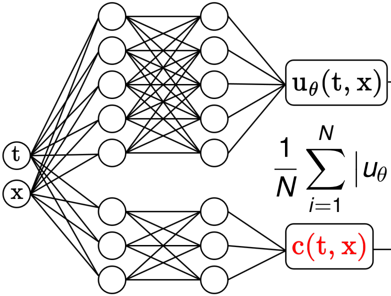
$$\frac{1}{N}\sum_{i=1}^{N}\left|u_\theta(t_i, x_i) - u_i\right|^2 + \frac{1}{M}\sum_{j=1}^{M}\left|\partial_t^2 u_\theta(\tilde{t}_j, \tilde{x}_j) - c\,\partial_x^2 u_\theta(\tilde{t}_j, \tilde{x}_j)\right|^2$$

c

Implementation of **PDE condition**:

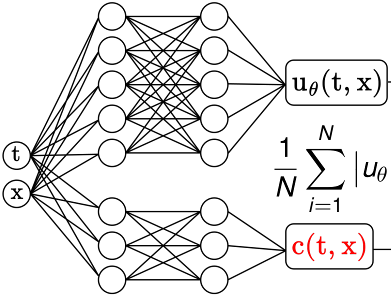
```
param_c       = tp.models.Parameter(init=1.0, space= C)
pde_condition = tp.conditions.PINNCondition(model_u, ...,
                                            parameter=param_c)
```

Identify Parameter Function with TorchPhysics



$$\frac{1}{N}\sum_{i=1}^{N}\left|u_\theta(t_i, x_i) - u_i\right|^2 + \frac{1}{M}\sum_{j=1}^{M}\left|\partial_t^2 u_\theta(\tilde{t}_j, \tilde{x}_j) - {\color{red}c(\tilde{t}_j, \tilde{x}_j)}\,\partial_x^2 u_\theta(\tilde{t}_j, \tilde{x}_j)\right|^2$$

Identify Parameter Function with TorchPhysics

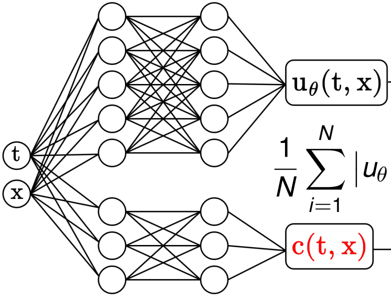


$$\frac{1}{N}\sum_{i=1}^{N}\left|u_\theta(t_i, x_i) - u_i\right|^2 + \frac{1}{M}\sum_{j=1}^{M}\left|\partial_t^2 u_\theta(\tilde{t}_j, \tilde{x}_j) - {\color{red}c(\tilde{t}_j, \tilde{x}_j)}\,\partial_x^2 u_\theta(\tilde{t}_j, \tilde{x}_j)\right|^2$$

Implementation of parallel model:

```
model_c = tp.models.FCN(...)
model   = tp.models.Parallel(model_u, model_c)
```

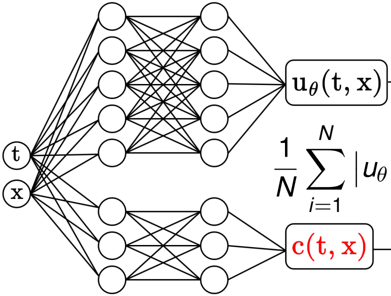
If data u_i is noisy...



$$\frac{1}{N}\sum_{i=1}^{N}\left|u_\theta(t_i, x_i) - u_i\right|^2 + \frac{1}{M}\sum_{j=1}^{M}\left|\partial_t^2 u_\theta(\tilde{t}_j, \tilde{x}_j) - {\color{red}c(\tilde{t}_j, \tilde{x}_j)}\,\partial_x^2 u_\theta(\tilde{t}_j, \tilde{x}_j)\right|^2$$

Hope for noise-resilience due to:

If data u_i is noisy...



$$\frac{1}{N}\sum_{i=1}^{N}\left|u_\theta(t_i, x_i) - u_i\right|^2 + \frac{1}{M}\sum_{j=1}^{M}\left|\partial_t^2 u_\theta(\tilde{t}_j, \tilde{x}_j) - c(\tilde{t}_j, \tilde{x}_j)\,\partial_x^2 u_\theta(\tilde{t}_j, \tilde{x}_j)\right|^2$$

Hope for noise-resilience due to:

- NN architectures for u_θ and c
- PDE loss term penalizes "chaotic" u_θ and c

Exercises: Inverse Problems

- Find parameter $c \in \mathbb{R}$ in wave equation:

$$\begin{cases} \partial_t^2 u = c\,\partial_x^2 u, & \text{in } I_x \times I_t, \\ u = 0 & \text{in } \partial I_x \times I_t, \\ \partial_t u(\cdot, 0) = 0 & \text{in } I_x, \\ u(\cdot, 0) = \sin(x) & \text{in } I_x, \end{cases}$$

  Template: `Exercise_8.ipynb`

- Find parameter function $D(t)$ in a ODE
  Template: `Exercise_9.ipynb`

Solution u for unknown c