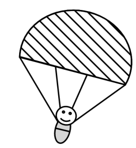
2.1 Solving a ODE with TorchPhysics



Use TorchPhysics to solve the ODE for falling with a parachute

$$\begin{aligned} \partial_t^2 u(t) &= D(\partial_t u(t))^2 - g, \\ u(0) &= H, \\ \partial_t u(0) &= 0, \end{aligned} \tag{1}$$

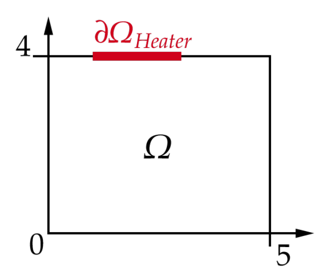
which we also considered yesterday. To open the prepared code template:

1. Open Google Colab
2. Select *open Notebook* and then the tab *GitHub*
3. Search: TomF98/torchphysics
4. Select the branch: *Workshop* and then Exercise2_1.ipynb

As a guideline, the example of the morning lecture can be found here.

⋆) **Bonus**: Extend your implementation, to learn the solution for multiple values of $D \in [0.01, 1.]$ and then also for different $H \in [50, 100]$ and $g \in [5, 10]$. Similar to the exercises of yesterday. **Hint:** Create seperate samplers for the respective parameters. Then multiply ("$*$") the time sampler with the parameter sampler in order to obtain a sampler which samples tuples (t, D).

2.2 Solving a PDE with TorchPhysics



Use TorchPhysics to solve the following heat equation:

$$\partial_t u(t,x) = D\Delta_x u(t,x), \quad \text{on } I \times \Omega, \tag{2}$$

$$u(0,x) = u_0, \quad \text{on } \Omega, \tag{3}$$

$$u(t,x) = h(t), \quad \text{at } I \times \partial\Omega_{Heater}, \tag{4}$$

$$\nabla_x u(t,x) \cdot \vec{n}(x) = 0, \quad \text{at } I \times (\partial\Omega \setminus \partial\Omega_{Heater}). \tag{5}$$

The above system describes an isolated room Ω, with a heater at the wall $\partial\Omega_{Heater} = \{(x,y) | 1 \leq x \leq 3, y = 4\}$. We set $I = [0, 20]$, $D = 1$, the initial temperature to $u_0 = 16\,°\mathrm{C}$ and the temperature of the heater to:

$$h(t) = \begin{cases} (16 + 24\frac{t}{5})\,°\mathrm{C}, & \text{if } t \leq 5, \\ 40\,°\mathrm{C}, & \text{if } t > 5. \end{cases}$$

a) **A PDE in TorchPhysics**: Implement the above equation with TorchPhysics. A template for this problem can be found under: Exercise2_2.ipynb.

b) **Domain Operations**: Next, we assume that the room contains a pillar (a circle) at position $(2, 2)$ with radius 0.5, remove this part from your domain. Here, also the boundary condition (5) holds.
Hint: Inside TorchPhysics, the difference of two domains `A` and `B` can be computed with `A - B`.

⋆) **Bonus**: Add a window at $\partial\Omega_{\text{Window}} = \{(x, y)|2 \leq x \leq 4, y = 0\}$ with fixed temperature of 16 °C.

⋆) **Bonus**: Let the network learn all solutions for $D \in [0.1, 5]$, like in the problem before.

2.3 Solving an inverse Problem with TorchPhysics

We are given a noisy dataset $\{(u_i, x_i, t_i)\}_{i=1}^N$ which corresponds to the solution of the wave equation

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

with $I_x = [0, 2\pi]$ and $I_t = [0, 20]$. Here, we aim to determine the unknown parameter c with the PINN approach. Follow the template given in Exercise2_3.ipynb to solve this exercise.

⋆) **Bonus**: In the notebook we added 1 % noise to the data and picked only half for the training. First, try out what results can be achieved with 5 % and 10 % noise. Second, if you keep 1 % noise but only use 10 % of the available data. Lastly, combine the case of 10 % noise with only 10 % of the available data and check the accuracy of the learned c and u.

2.4 Solving a PDE with the Deep Ritz Method

Instead of using PINNs, we now try a different approach: the Deep Ritz Method. Use TorchPhysics and the Deep Ritz Method to solve the following problem:

$$\begin{aligned}
\Delta u(x) &= f(x), \text{ in } \Omega, \\
u &= 0, \text{ on } \partial\Omega
\end{aligned} \tag{6}$$

Here, $f(x) = 8\pi^2 \sin(2\pi x_1) \sin(2\pi x_2)$ and $\Omega = [0, 1] \times [0, 1]$. The general idea of using a neural network to learn the solution is the same. We only change how the physics (the PDE) is utilized in the training. Instead of using the above (strong) formulation, the Deep Ritz Method minimizes the energy functional:

$$\int_\Omega \tfrac{1}{2}\|\nabla u(x)\|^2 - f(x)u(x)\mathrm{d}x + \lambda \int_{\partial\Omega} u(x)^2 \mathrm{d}x. \tag{7}$$

One can mathematically show, for a weight $\lambda > 0$, that a solution of (6) is a minimum of (7) and vice versa. The implementation inside TorchPhysics is similar to the PINN case and is the focus of this exercise and can be found in Exercise2_4.ipynb.