
PINA is an open-source software powered by PyTorch and Lightning, designed for solving differential equations with neural networks. It supports Physics Informed Neural Networks and Neural Operators, offering flexibility for users to craft models tailored to their needs. PINA is modular and adaptable to different hardware setups, including GPUs and TPUs.

SOLVE YOUR DIFFERENTIAL EQUATION STEP BY STEP

1. **Problem definition**
   - Specify the mathematical problem aimed to be solved and the specific physical conditions to be satisfied.

2. **Sample the domain**
   - Prepare the input of the model by discretizing the physical domain, or import data from numerical solvers.

3. **Model & Solver selection**
   - Build a Model as a PyTorch Module and choose the Solver strategy to optimize the model and solve the problem.

4. **Training**
   - Optimize the Model with the specific Solver strategy with all the additional Pytorch Lightning features.

TRASLATE YOUR PROBLEM INTO PINA LANGUAGE

- **Spatial, Time-dependent, Inverse, and Parametric problems** are already available!
- We implemented the most common differential operators for easiness of usage:
  - grad
  - div
  - laplacian
- Triangular, cartesian, and elliptic shapes can be used to define the problem domain. Moreover, boolean operations have been implemented to make possible the definition of complex domains.

A HIERARCHICAL PERSPECTIVE: SOLVER, MODEL, LAYER

- **Physics-informed paradigm**, the solver minimizes the physical residual:
  - PINN, GPINN, CausalPINN, CompetitivePINN, SAFPINN, ...
- **Supervised learning paradigm**, the solver minimizes the difference between data and network output:
  - SupervisedSolver, GAROM, Message Passing Neural PDE, Reduced Order Modeling Solvers, ...
- **Abstract Interfaces** to easily build new solvers:
  - SolverInterface, PINNInterface, ...

**Solver**
- The “strategy” we want to apply to solve the problem — e.g. PINN

**Model**
- Standard and customizable deep learning architectures:
  - FeedForward, MultilayerFeedForward, ResidualFeedForward, ...

**Layer**
- Many PyTorch implementations of deep learning Layers:
  - ContinuousConvBlock, ResidualBlock, EnhancedLinear, ...

**Layer**
- The minimal brick that can be used to build a Model — e.g. SpectralConv

HIGHLIGHTS

- **PINA** is built upon PyTorch Lightning
  - CPU GPU and TPU training support
  - Loggers and Checkpoints for monitoring training
  - Gradient Clipping, SWA, Gradient accumulation ...
  - Callbacks for Solvers and Trainer

- **LabelTensor**
  - Extension of PyTorch Tensor class to handle labels
  - Easily extract variables with strings and compute differential operator in symbolic notation
  - Compatible with Pytorch main Tensor operations

- **High-level design**
  - Modular components and object-oriented structure
  - Abstract interfaces to easily add new components
  - Natively support Pytorch Model compilation

- **Documentation online**
  - Fully documented package
  - Installation and contribution guidelines
  - Tutorials for getting start with the software

- **Tested for several operating systems**
  - Fully compatible and tested for Python ≥ 3.7
  - Running on Windows, macOS, and Ubuntu

**Easy installation**
- `pip install pina-mathlab`
- New release the 1st of any month

CHECK THE GITHUB PAGE &
LEAVE US A STAR ⭐️