AI4PDE and PINN

Basic Problems,Numerical Methods and PINN on Different Architectures

Yunfeng Liao

12 July,2024

Presentation Overview

Sources
What are PDE and AI4PDE?
Salient Numerical Methods:Finite Difference and Finite Element
Ways to Solve PDE with DL
PINN On Different Architectures: FFN,RNN,CNN

Sources

Articles

- (Review) Scientific Machine Learning through Physics-Informed Neural Networks: Where we are and What's next. Ruiyang Zhanga et al.
- Physics-Informed Neural Networks: A Deep Learning Framework for Solving Forward and Inverse Problems Involving Nonlinear Partial Differential Equations. M. Raissi et al.
- Physics-Informed Multi-LSTM Networks for Metamodeling of Nonlinear Structures. Ruiyang Zhanga et al.
- PhyGeoNet: Physics-Informed Geometry-Adaptive Convolutional Neural Networks for Solving Parameterized Steady-State PDEs on Irregular Domain. Han Gao et al.

Tutorials

- Partial Differential Equations. Evans.
- 微分方程数值解法. 李荣华,刘播.
- 数值分析. 李庆扬等.
- Finite Element Methods and Their Applications. Zhangxin Chen.
- Introductory Variational Calculus on Manifolds. Ivo Terek.
- From Fourier Analysis to Wavelets. Jonas Gomes, Luiz Velho.

Section Overview

What are PDE and AI4PDE?

Basic Knowledge of PDE

PDE

A pde problem consists of the PDE itself, along with boundary conditions (B.C.) and initial conditions (I.C.). $\mathbf{u}(\mathbf{x}), \mathbf{x} \in \Omega$ is to be solved.

$$F(t, \mathbf{u}, \frac{\mathrm{d}^{\mathrm{n}}}{\mathrm{d}t^{\mathrm{n}}}\mathbf{u}, \nabla \mathbf{u}, \nabla\nabla \mathbf{u} ...) = 0, \mathbf{x} \in \Omega \tag{1}$$

$$u(\mathbf{x}, t) = f(\mathbf{x}, t), \mathbf{x} \in \partial\Omega \tag{2}$$

$$u(\mathbf{x}, 0) = g(\mathbf{x}), \mathbf{x} \in \Omega \tag{3}$$

Terminology

In fact, the B.C. $u(x, t) = f(x, t)$ is called **Dirichlet B.C.** while $\partial_{x^i} u = f(x, t)$ and $mu + n\partial_{x^i} u = f(x, t)$ are named **Neumann B.C.** and **Robin B.C.**,respectively.

Basic Knowledge of PDE

Terminology

Usually, a PDE is called **time-dependent** or **non-autonomous system** if t is explicitly included in F and otherwise **time-independent** or **autonomous system**.

$$F(t, \mathbf{u}, \frac{\mathrm{d}^n}{\mathrm{d}t^n}\mathbf{u}, \nabla\mathbf{u}, \nabla\nabla\mathbf{u}\ldots) = 0, \mathbf{x} \in \Omega \tag{4}$$

However, one can take t as one spatial dimension, turning a time-dependent one into an independent one.

Terminology

A PDE is **linear** if F is linear.

Basic Knowledge of PDE

Common PDEs are:

- 2nd-order **Elliptic**. $\nabla \cdot (g \nabla u) = h(x, t)$. It is **homogeneous** if h vanishes. The electric potential $\triangle u = -\frac{1}{\varepsilon_0}\rho(\mathbf{x}, t)$ and Helmholtz equation $(\triangle + k^2)u = 0$.
- 2nd-order **Hyperbolic**. Wave function $\triangle u - \frac{1}{v^2} u_{tt} = 0$.
- 2nd-order **Parabolic**. Heat equation. $\triangle u - \rho u_t = 0$.
- Navier-Stokes equation. $\rho \frac{D}{Dt} V = \rho f - \nabla p + \mu \triangle V$

Salient Numerical Methods:Finite Difference and Finite Element

Finite Difference

Finite difference method solves PDE by replacing derivatives with differences where the spatial and temperoal intervals are denoted as h, τ, respectively.

$$u_t(x,t) = f(t,u) \Rightarrow \frac{u^n(x) - u^{n-1}(x)}{\tau} = f(u^{n-1}) \tag{5}$$

where $u^n(x) := u(n\tau, x)$

Terminology

A **scheme** is the way how to compute the difference. A scheme is **explicit** if one can figure out u iteratively as the above formula (**Forward Euler Method**) and otherwise **implicit** such as $u^{n-1} \leftarrow u^n$(**Backward Euler Method**).

Finite Difference:Runge-Kutta Method

Notice that

$$u_t(x,t) = f(t,u) \Leftrightarrow u(t+\tau) - u(t) = \int_t^{t+\tau} f(s,u(s))ds \tag{6}$$

It is expected to approximate f via a linear combination of q samples $k_i := f(t_i, u(t_i))$ within $[t, t+\tau]$, i.e.

$$f \approx \sum_{i=1}^{q} c_i k_i, \sum c_i = 1 \tag{7}$$

q is the **state** of the Runge-Kutta method and the highest non-vanishing order p in Taylor expansion is called the **order**.

Finite Difference:Runge-Kutta Method

We must determine the length of the sampling interval in one go. The scheme can be:

$$[t, t + a_0\tau], [t, t + ta_1\tau]...[t, t + a_q\tau], a_0 = 0, a_q = 1, 0 < a_i < 1. \tag{8}$$

To evaluate f via $k_j = f(t + a_j\tau, u(t) + k_1 b_{1j}\tau + ... + k_{j-1} b_{j-1,j}\tau)$ in jth step, one give the sampled j points within $[t, t + a_j\tau]$ a proper weight b_{1j} s.t.

$$a_i = \sum_j b_{ij} \tag{9}$$

Apply Taylor series with (6)(7)(9), and determine the coefficients a_i, b_{ij}, c_i, despite not unique as $m > 1$. ($m = 1$ is simply Euler method)

Finite Element

Examples

Consider the problem:

$$\Delta u = -f, x \in \Omega; u = 0, x \in \partial\Omega \tag{10}$$

We can show that to look for u satisfying (10) is to look for a critical-point of the functional S (also called **action**). That is, we've **turned a PDE problem into an optimization problem**.

$$S[u] = \frac{1}{2}(\nabla u, \nabla u) - (f, u) \tag{11}$$

where $(u, v) := \int_\Omega uvdw$.
However, finite element method can only work out PDEs that have *variation forms* albeit quite abundant.

Section Overview

Ways to Solve PDE with DL

Birdview

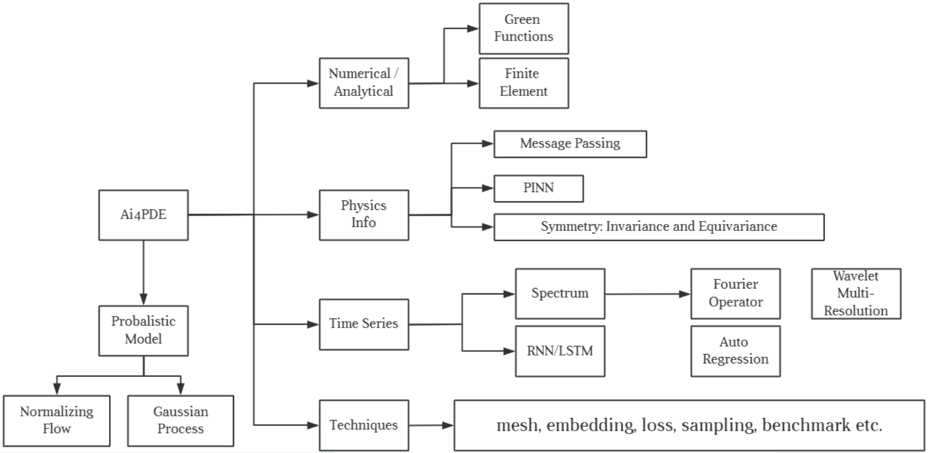


Fig. 1: Ways to Solve PDE with DL

Section Overview

PINN On Different Architectures: FFN,RNN,CNN

Problem Setup

$$u_t + F(u, t, x; \lambda) = 0 \tag{12}$$

with B.C. and I.C. The tasks are:

- Predict the system behavior with **continuous** observations
- Reconstruct the physical system with sparse **observations** $u(x, t)$
- **Inverse Problem**. Find out the parameter λ in the system

PINN can be data-driven or unsupervised.

PINN

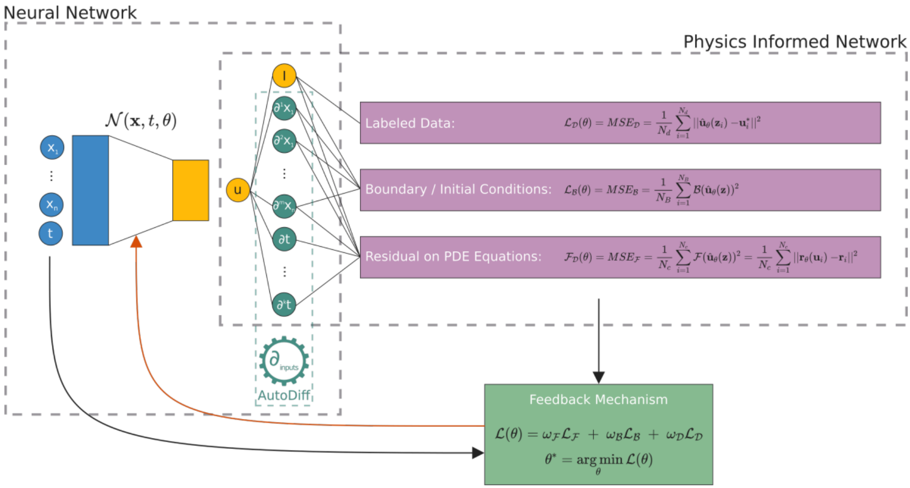


Fig. 2: PINN

Loss Terms

Add the penalty terms to reflect the physics scenario without changing inner structures so as to turn it into an optimization problem **[2]**. Let $\mathbf{u}_\theta$ denote the outputs.

- **Labeled Data Loss**. $MSE_D = \frac{1}{N_d}\sum|\mathbf{u}_\theta(\mathbf{x}_i) - \mathbf{u}_i|^2$.
- **B.C. Loss** $MSE_B = \frac{1}{N_B}\sum|\mathbf{u}_\theta(\mathbf{x}_i, t) - f(\mathbf{x}, t)|^2$.
- **I.C. Loss** $MSE_I = \frac{1}{N_I}\sum|\mathbf{u}_\theta(\mathbf{x}_i, 0) - g(\mathbf{x})|^2$.
- **PDE Loss** $MSE_P = \frac{1}{N_C}\sum|\partial_t \mathbf{u}_\theta + F|^2$.

Remark

Taking B.C. restrictions as the penalty is named **Soft B.C. Enforcement**. One can choose to embed this constraint into the network itself, which is **Hard B.C. Enforcement [4]**.

Combined With R-K Method

By the introduced R-K method (6)(7)(9), one can compute:

$$u^{n+c_i} = u^n - \Delta t \sum_j b_{ij} F(u^{n+c_j}; \lambda) \tag{13}$$

$$u^{n+1} = u^n - \Delta t \sum_j a_j F(u^{n+c_j}; \lambda) \tag{14}$$

where $u^{n+c_j} \equiv k_j$. Now we wanna make the network learn something like the simulation of **q-stage** R-K method. Therefore, NN is supposed to give a q-length tuple (u^{n+c_i}) as ith estimation.

There's a trick in constructing the loss terms.

Loss Term

Intuition

To train it effectively, one must use a pair of consecutive data like $(x^n, t^n, u^n), (x^{n+1}, t^{n+1}, u^{n+1}))$ and let NN also gives a pair.We claim that a direct integration as $|u^{n+c^j} - u^n|^2$ makes no sense. On contrary, we only take the output as intermediaries and figure out u^{n+1} using R-K method.

$$u^n_{(i)} := u^{n+c_i} + \Delta t \sum_j b_{ij} F(u^{n+c_j}; \lambda) \tag{15}$$

$$u^{n+1}_{(i)} := u^n_{(i)} - \Delta t \sum_j a_j F(u^{n+c_j}; \lambda) = u^{n+c_i} + \Delta t \sum_j (b_{ij} - a_j) F(u^{n+c_j}; \lambda) \tag{16}$$

$$Loss = SSE_n + SSE_{n+1} \tag{17}$$

$$= \sum_{Data} |u^n_{(i)}(\tilde{x}^n, \tilde{t}^n) - \tilde{u}(\tilde{x}^n, \tilde{t}^n)\Big|^2 + |u^{n+1}_{(i)}\left(\tilde{x}^{n+1}, \tilde{t}^{n+1}\right) - \tilde{u}\left(\tilde{x}^{n+1}, \tilde{t}^{n+1}\right)\Big|^2 \tag{18}$$

Assess

Pros

- Efficiency in increasing the order of R-K. Only the output layer scale adds up relatively.
- No need to care about system parameters λ if data-driven. (if unsupervised, a proper loss may be learned)

PINN Over CNN

PhyGeoNet: Physics-Informed Geometry-Adaptive Convolutional Neural Networks for Solving Parameterized Steady-State PDEs on Irregular Domain. Han Gao et al.

Innovations

- How to transform an irregular boundary into a rectangle so as to use a CNN?
- How to implement hard-BC-enforcement?

Assess

Compared with simple feed-forward network, CNN outperforms in the aggregation within the perception field, and the translation equivariance.

PINN Over CNN

How to transform an irregular boundary into a rectangle so as to use a CNN?

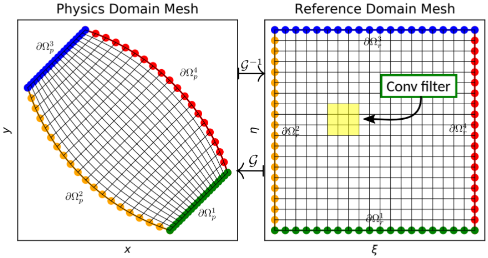


Fig. 3: Elliptic Coordinate Transform

$$x = a\cosh\xi\cos\eta, y = a\sinh\xi\sin\eta \tag{19}$$

PINN Over CNN

The coordinate transform above gives:

$$\nabla^2 u = \frac{2}{a^2(\cosh 2\xi - \cosh 2\eta)}(\nabla')^2 u \tag{20}$$

where $\nabla^2 = \partial_x^2 + \partial_y^2, (\nabla')^2 = \partial_\xi^2 + \partial_\eta^2$
Therefore, assume that the transform acting on the boundary is obtained as a prior, i.e.

$$\xi(\mathbf{x}) = \xi, \forall \mathbf{x} \in \partial\Omega_p; \mathbf{x}(\xi) = x, \forall \xi \in \partial\Omega_r \tag{21}$$

Then to yield G, one only need to solve $\nabla^2 u = 0$ on chart (ξ, η) **numerically**, and the property of elliptic transformation tells $\nabla^2 u = 0$ on chart (x, y). Since the Laplacian operator is coordinate-independent, we claim that the two harmonic fields are the same.

PINN Over CNN

On the chart (ξ, η), one can figure out things like $\frac{\partial x}{\partial \xi}$ easily. And for PDEs on chart (x, y), one can thereby translate the derivatives into those in (ξ, η) through the chain rule.

Drawbacks

- Challenging to tackle with more complicated $\partial\Omega$.
- Hard to determine the transform over $\partial\Omega$ in higher dimensions.

PINN Over CNN

How to implement hard-BC-enforcement?

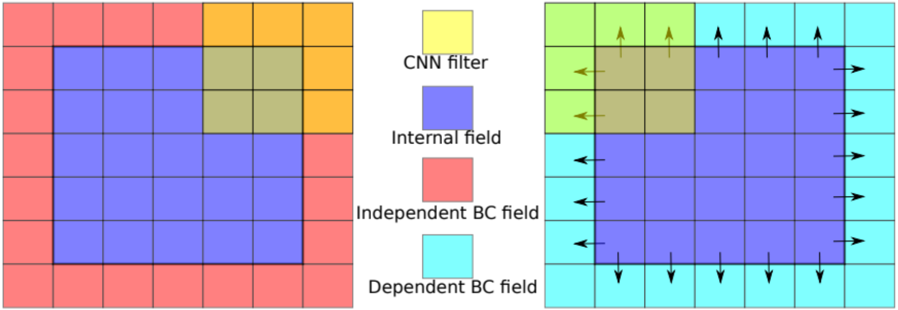


Fig. 4: Hard enforcement of BCs via padding

PINN Over CNN

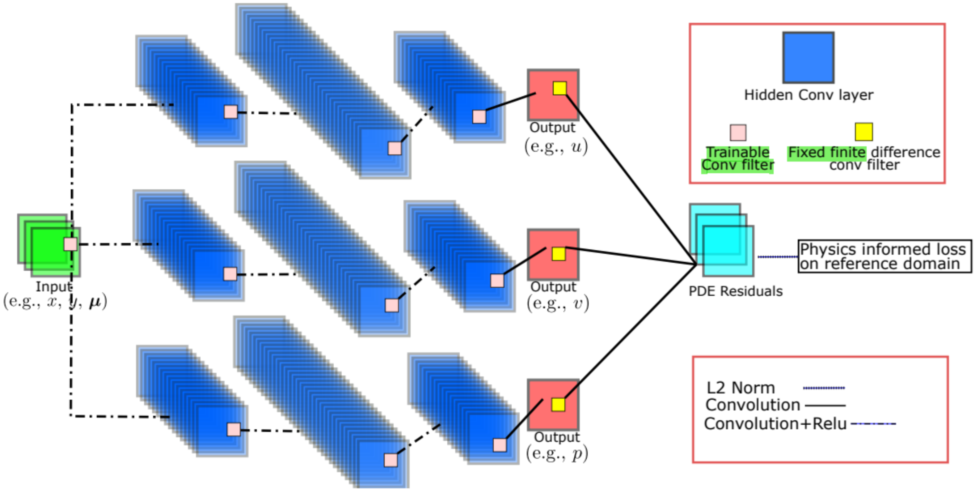


Fig. 5: Architecture of PINN Over CNN

PINN Over LSTM

The problem setup is two seismic wave models.

$$Mu'' + Cu' + \lambda Ku + (1-\lambda)Kr(\Phi) = -M\Gamma a_g \tag{22}$$

A simpler version is:

$$u'' + g = -\Gamma a_g \tag{23}$$

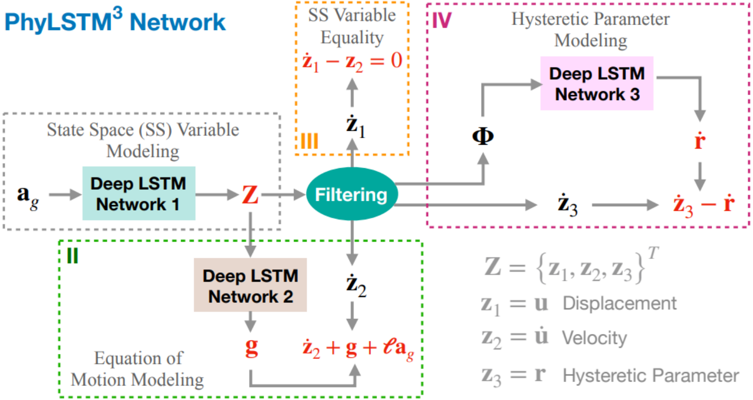
where u, u', u'' are the displacement,velocity and acceleration respectively. M, C, K, λ are the system parameters and only a_g can be observed.

Innovations

- Evaluate u, u', u'' separately and **take the identities as physical info**, i.e. $|\frac{d}{dt}u - u'|$.
- more capable of modeling the latent non-linearity **g** explicitly .

PINN Over LSTM

Filtering: $Z \to Z'$



PINN Over LSTM

Loss terms:

- Identity loss: $\sum|\frac{d}{dt}u_i - u_i'|^2$
- Data loss: $\sum|u_i - u_{(i)}|^2 + |u_i' - u_{(i)}'|^2$ where $u_{(i)}$ is the data.
- PDE loss: $\sum_i|u'' + g + \Gamma a_g|^2$

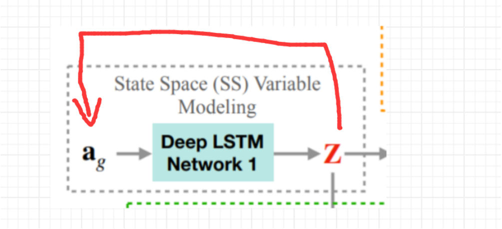
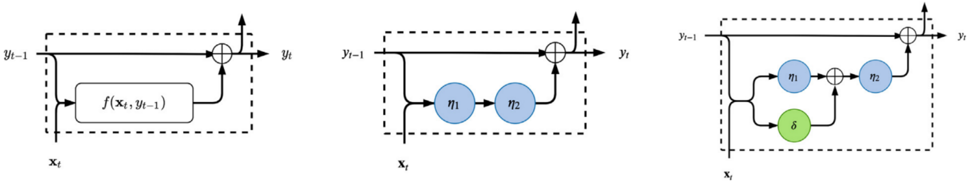


Fig. 6: Case without observations a_g: auto-regression

PINN Over LSTM

Motivation: Euler-method cells or resNet

$$y_n = y_{n-1} + f(x_n, y_{n-1}) = y_{n-1} + \eta_2(\eta_1 + \delta) \tag{24}$$



(a) General model. (b) Physics-informed example. (c) Hybrid example.

Estimating model inadequacy in ordinary differential equations with physics-informed neural networks. Felipe A.C. Viana et al.