Physics Informed Neural Networks for an Inverse Problem in Peridynamic Models

F.V. Difonzo
joint work with L. Lopez (Università di Bari) and S.F. Pellegrino (Politecnico di Bari)

Consiglio Nazionale delle Ricerche

Istituto per le Applicazioni del Calcolo "Mauro Picone"

PINN-PAD: Physics Informed Neural Networks in PADova
Padova, February 22-23, 2024

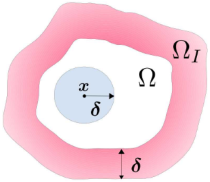
Peridynamic models

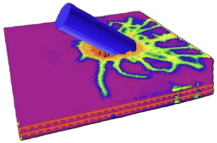
Basic concepts:

- The state of a system at any point depends on the state in a **neighborhood** of points
- Interactions occur at finite distance, even without contact
- Solutions can be non-differentiable, singular, discontinuous

Advantages:

- Multiscale behaviors
- Cracks and fractures
- Predictive capability



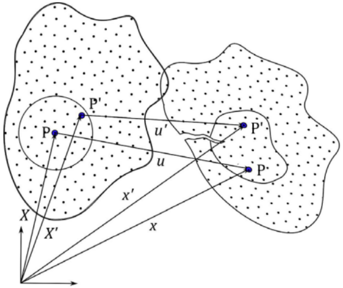


Peridynamic Model

We consider the following PDE in *linear peridynamic*[1] formulation:

$$\partial_{tt}\theta(x,t) = \int_\Omega C(|x-y|)[\theta(x,t)-\theta(y,t)]\,\mathrm{d}y,$$

which describes the dynamic response of an infinite bar composed of a linear microelastic material.



- The general initial-value problem is well-posed[2]
- long-range forces $\rightarrow$ solution showing a dispersive behavior
- nonnegative even kernel $\rightsquigarrow$ the regularity of the solution
- *nonlocal operator* in the integrand $\rightsquigarrow$ deformation of θ

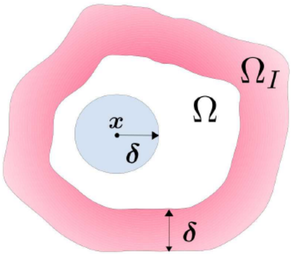
[1] Silling 2000.

[2] Emmrich and Puhst 2015.

Nonlocal operator

$$\mathcal{L}\theta(x) = \int_{\Omega} C(|x-y|)[\theta(x,t)-\theta(y,t)]\,\mathrm{d}y$$

- **integral form**: allows long-range forces and reduce regularity requirements
- $\mathcal{L}\theta \to \Delta\theta$ in a **suitable sense** for vanishing nonlocality $\rightsquigarrow$ $\partial_{tt}\theta(x,t) - \partial_{xx}\theta(x,t) = 0$[3]
- the **kernel** $C(|x-y|)$ is application dependent
- δ is the **horizon** and measures the nonlocality



[3]Oterkus, Madenci, and Agwai 2014.

Peridynamic formulation: kernel properties

In order to maintain the consistency with Newton's third law, the micromodulus function must be even:

$$C(\xi) = C(-\xi), \quad \xi \in \mathbb{R}.$$

Moreover, due to the dispersive effects C must be such that[4]

$$\int_{\mathbb{R}} (1 - \cos(k\xi))\, C(\xi)\, \mathrm{d}\xi > 0,$$

for every wave number $k \neq 0$.
Additionally, since the interaction between two material particles should become negligible as the distance between particles become very large, we can assume that

$$\lim_{\xi \to \pm\infty} C(\xi) = 0.$$

[4] Weckner and Abeyaratne 2005.

Aim

- Solving the inverse problem to learn the shape of the kernel function C by a PINN
  - $\rightarrow$ careful selection of activation functions in all the layers
  - $\rightarrow$ correct interaction with kernel initializers
  - $\rightarrow$ geometric knowledge relative to the data
  - $\rightarrow$ the peridynamic operator is bounded on a compact support $[-\delta, \delta]$

Kernels

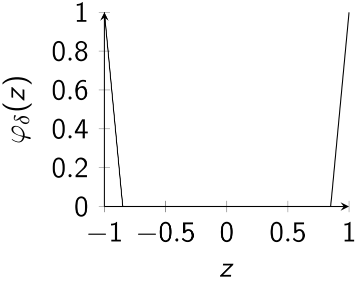
We will focus on a Gauss-type kernel[5] of the form

$$C(\xi) = \lambda e^{-\mu\xi^2}, \qquad \lambda,\, \mu > 0.$$

and on a distributed kernel function with shape[6]

$$C(\xi) = \begin{cases} \frac{|\xi|-\lambda+\delta}{\delta}, & |\xi| \geq \lambda - \delta \\ 0, & |\xi| < \lambda - \delta \end{cases}$$

proposed in nonlocal unsaturated soil model contexts.



[5]Weckner and Abeyaratne 2005.

[6]Berardi, Difonzo, and Pellegrino 2023.

Radial Basis Functions

As activation function for the first layer, whose input is x, a Radial Basis Function (RBF) is selected. An RBF can be defined as

$$\phi(x) = \phi(\|x - c\|),$$

where ϕ is the RBF function, x is the input to the RBF, c is the center or prototype point, $\|x - c\|$ represents the distance between x and c. When used as activation functions in neural networks, they give rise to Radial Basis Function Neural Networks (RBFNNs)[7].

[7] Fasshauer 2007.

Radial Basis Functions

We have considered two families of RBFs given by

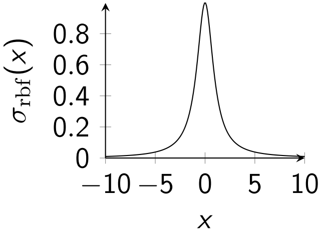
- Inverse quadratic:

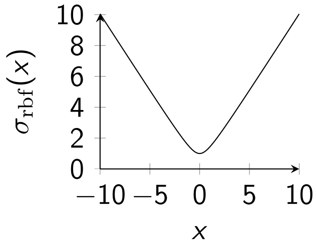
$$\sigma_{\mathrm{rbf}}(x) := \frac{\rho}{1+\gamma(x-\mu)^2}$$

- Multiquadric:

$$\sigma_{\mathrm{rbf}}(x) := \rho\sqrt{1+\gamma(x-\mu)^2}$$

$\rightsquigarrow$ $\rho, \gamma, \mu > 0$ could be trainable.





RBF-iPINN

- *Spatial NN* and *Temporal NN* in series

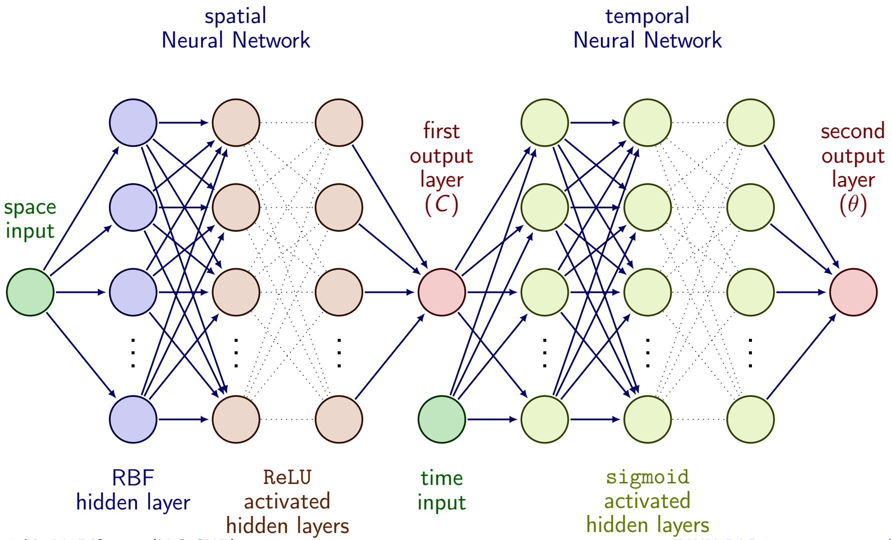
  Spatial NN: the spatial variable x is the sole input of a hidden layer of 20 neurons, activated by an RBF, followed by 8 layers with 20 neurons each, activated by `ReLu` function; kernel initializer of type `glorot_normal`; nonnegative kernel constraint and a kernel regularizer of type `l1_l2`, with weights $l_1 = l_2 = 0.01$

  Concatenation: The output of this sequence of layers is then concatenated with t, providing the input for

  Temporal NN: 8 layers, each containing 20 neurons and activated by a `sigmoid` function; `random_uniform` kernel initializer.

- the overall output of the RBF-iPINN is returned as an array that lists two tensors, the first carrying the kernel C, and the other carrying the dependent variable θ.

RBF-iPINN structure



RBF-iPINN

Remark

Let us notice that selecting the `ReLU` activation function for all the layers of the architecture could result in a loss of compatibility potential of the PINN.[8] This consideration, also supported by several experiments, justifies the choice of the `sigmoid` activation function in the temporal NN. We witness that, however, other selections than `sigmoid` function do not perform satisfactorily enough.

[8] Leng and Thiyagalingam 2023.

Loss function

$$
\begin{aligned}
\text{pde loss: } \mathcal{L}_{\text{pde}} &:= \|\mathcal{P}(f) - 0^*\|_2, \\
\text{data fitting loss: } \mathcal{L}_{\text{data}} &:= \|f - f^*\|_2, \\
\text{symmetry loss: } \mathcal{L}_{\text{sym}} &:= \|f(x,t) - f(-x,t)\|_1 \to \text{ small errors.}
\end{aligned}
$$

Then we consider a weighted sum of the contributions given above as

$$\mathcal{L} = w_{\text{pde}}\mathcal{L}_{\text{pde}} + w_{\text{data}}\mathcal{L}_{\text{data}} + w_{\text{sym}}\mathcal{L}_{\text{sym}},$$

where

$$w_{\text{pde}} = 2,\ w_{\text{data}} = 1,\ w_{\text{sym}} = 2.$$

Learning rate

The learning rate α has been selected to be decreasing with the epoch in a quadratic way. More precisely, we implemented the following scheduler:

$$\alpha_0 = 10^{-4},\ \alpha_1 = 0.7\alpha_0$$
$$\alpha_i = \left(1 - \left(\frac{i}{N}\right)^2\right)\alpha_0 + \left(\frac{i}{N}\right)^2 \alpha_1, \quad i = 0, \dots, N,$$

where N is the number of epochs chosen for the training. Thus, starting with a learning rate of α_0 at epoch 0, it progressively gets reduced over the epochs, until it reaches the value α_1 at epoch N.

Experiment setup

A main feature of the numerical computation is the evaluation of the integral

$$\begin{aligned}\int_{\mathbb{R}} C(|x-y|)[\theta(x,t)-\theta(y,t)]\mathrm{d}y &= \theta(x,t)\int_{\mathbb{R}} C(|x-y|)\mathrm{d}y - \int_{\mathbb{R}} C(|x-y|)\theta(y,t)\mathrm{d}y \\ &= \theta(x,t)\int_{\mathbb{R}} C(|x-y|)\mathrm{d}y - C(|x|)*\theta(x,t),\end{aligned}$$

where the second term in the right-hand side above is the convolution product between the kernel C and the unknown function θ. It has to be noticed here that the kernel function C is compactly supported, with support $[-\delta,\delta]$.

Experiment setup

Now, in order to numerically compute such convolution product, let $[0, X]$ be the space interval and let $0 < x_1 < x_2 < \ldots < x_{N-1} < x_N = X$ be the uniform spatial discretization of the interval $[0, X]$ with stepsize $h > 0$. the convolution product above can be numerically treated by determining the exact number of components in the vector $[C(x_i)]_{i=1}^{n}$ so that only points x_i such that

$$|x_i - x_j| < \delta, \ i, j = 1, \ldots, N,$$

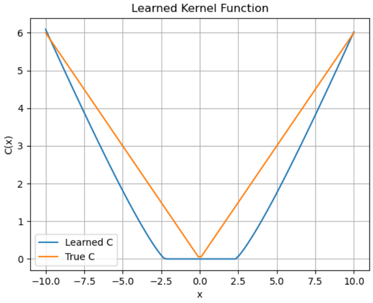
come into play when computing $C(\|x\|) * \theta(x, t)$. Since $x_i = i \cdot h$, then we deduce that the only indices involved in the convolution product are $i, j = 1, \ldots, N$ such that

$$|i - j| < \frac{\delta}{h}.$$

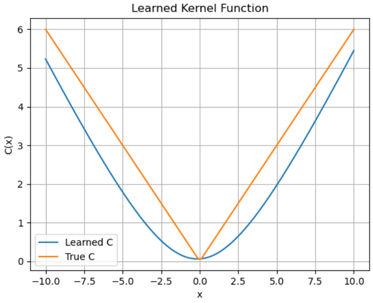
Experiment 1

Here we consider a dataset with $t \in [0, 20]$, $x \in [-10, 10]$ with spatial stepsize $h = 2 \cdot 10^{-1}$ and $\delta = 10$, and for which the analytical expression of the kernel is

$$C(x) = \frac{3}{5}|x|.$$



(a) $\gamma = 0.09$

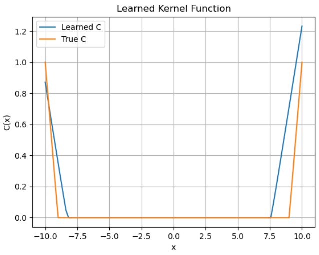


(b) $\gamma = 0.05$

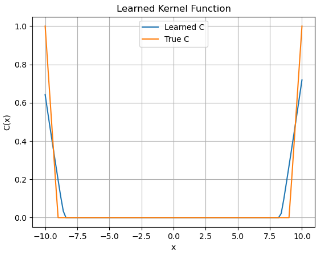
Experiment 2

Here we consider a dataset with $t \in [0, 20], x \in [-10, 10]$ with spatial stepsize $h = 2 \cdot 10^{-1}$ and $\delta = 1$, with kernel

$$C(x) = \begin{cases} \frac{\delta - x - 10}{\delta}, & x \leq -10 + \delta, \\ 0, & -10 + \delta < x \leq 10 - \delta, \\ \frac{\delta + x - 10}{\delta}, & x > 10 - \delta. \end{cases}$$



(a) $\gamma = 0.09$



(b) $\gamma = 0.05$

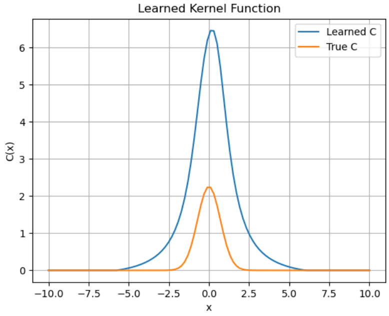
Experiment 3

Here we consider a dataset with $t \in [0, 20], x \in [-10, 10]$ with spatial stepsize $h = 2 \cdot 10^{-1}$ and $\delta = 1$, with kernel

$$C(x) = \frac{4}{\sqrt{\pi}} e^{-x^2}.$$

Hyperparameters tuning:

- `l1_l2`$= 0.01, 0.1$;
- $\sigma_{\text{rbf}}(x) = \frac{\rho}{1+(x-\mu)^2}$;
- $\mathcal{L}_{\text{data}} = \|f - f^*\|_{\infty}$.



Experiment 3

Therefore, we have performed a further analysis by implementing a standard inverse PINN to learn parameters γ^* and σ^* in

$$C^*(x) := \gamma^* e^{-\sigma^* x^2}.$$

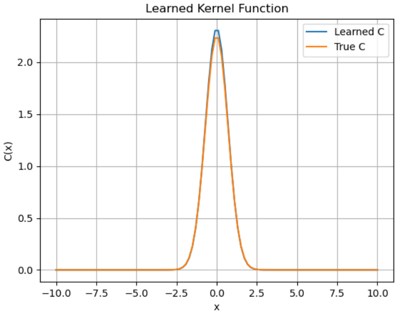
Initial guesses: $\gamma^* = 3,\ \sigma^* = 0.5$.

Hyperparameter tuning:

- 1000 epochs
- same learning rate scheduler with $\alpha_0 = 10^{-3}$

Results:

- learned $\gamma^* = 2.3302033$, true $\frac{4}{\sqrt{\pi}} \approx 2.2567583$
- learned $\sigma^* = 1.0218402$, true 1



Conclusions

- peridynamic formulation of a classical wave equation
- computation of the kernel function responsible for the nonlocal behavior of the model
- two serialized PINNs (spatial and temporal)
- Radial Basis Function (RBF) as activation function for the spatial NN

¿ optimal control problems?

¿ more complicated peridynamic models via PINNs and Radial Basis Functions, exploiting their inherently symmetric nature?

Conclusions

- peridynamic formulation of a classical wave equation
- computation of the kernel function responsible for the nonlocal behavior of the model
- two serialized PINNs (spatial and temporal)
- Radial Basis Function (RBF) as activation function for the spatial NN

¿ optimal control problems?

¿ more complicated peridynamic models via PINNs and Radial Basis Functions, exploiting their inherently symmetric nature?

THANK YOU