Solving PDEs using NNs -

Applied Mathematics MSc Individual Project III

Miskei Ferenc István

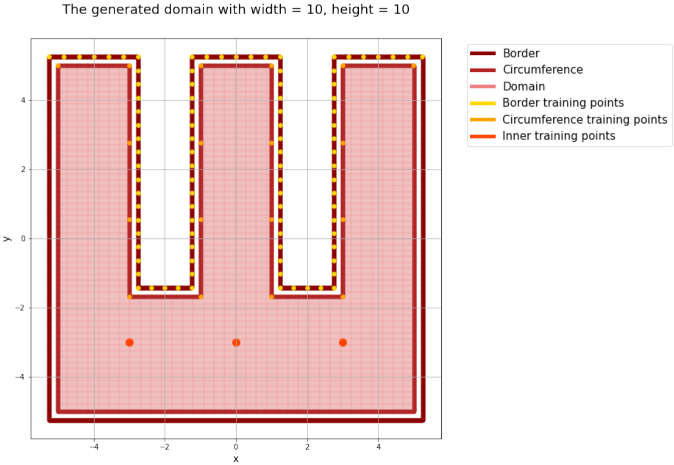
22 December 2022

Basic setup

- Trying to create <u>numerical solutions</u> to PDEs using NNs
- Main problem: Lack of training data
- New approach: Use of fundamental solutions

Basic setup

- Laplace's equation with non-homogeneous Dirichlet type boundary condition: $\Delta u = 0, (\mathbf{x} \in \Omega), u(\mathbf{x}) = g(\mathbf{x}), (\mathbf{x} \in \partial\Omega)$.



Definition of the method

Train a neural network – whose architecture will be specified later – such that

$$\mathrm{NN} : \underbrace{\left(G(\mathbf{z}_k, \mathbf{y}_j)\right)_{k=1}^{K}}_{\in \mathbb{R}^K} \mapsto \underbrace{\left(G(\mathbf{x}_i, \mathbf{y}_j)\right)_{i=1}^{I}}_{\in \mathbb{R}^I} \quad (\forall \mathbf{y}_j \in Y). \tag{1}$$

Then, we define the numerical approximation as

$$\tilde{u} \colon \mathbb{R}^K \to \mathbb{R}^I, \quad (\tilde{u}(\mathbf{x}_i))_{i=1}^{I} := \mathrm{NN}\left((g(\mathbf{z}_k))_{k=1}^{K}\right) \tag{2}$$

- Essentially an interpolation problem, where we impose $\Delta u = 0$ by only picking functions that already satisfy this criterion

Thoretical result

Theorem

Suppose that the points $Y = \{\mathbf{y}_j\}_{j=1}^{J}$ equally spaced around the domain, and their distance from the domain is above a particular positive constant. Then, the approximation

$$\tilde{u}(\mathbf{x}) := \sum_{j=1}^{J} a_j G(\mathbf{x} - \mathbf{y}_j) := \sum_{j=1}^{J} a_j G_{\mathbf{y}_j}(\mathbf{x})$$

provides the convergence rate $\mathcal{O}(h)$, where $h = \mathrm{dist}(\mathbf{z}_j, \mathbf{z}_{j+1})$.

Experiments on the first problem

The exact values to be estimated are $u(\mathrm{x}_1) = -21$, $u(\mathrm{x}_2) = 0$ and $u(\mathrm{x}_3) = 39$, and the relative error is calculated as:

$$e_p = \frac{\|(u(\mathrm{x}_1) - \tilde{u}(\mathrm{x}_1), u(\mathrm{x}_2) - \tilde{u}(\mathrm{x}_2), u(\mathrm{x}_3) - \tilde{u}(\mathrm{x}_3))\|_p}{\|(u(\mathrm{x}_1), u(\mathrm{x}_2), u(\mathrm{x}_3))\|_p}$$

opt	lr	ep	e_1	e_2	e_∞
Adam	0.1	$1k$	0.7358	0.7345	0.7851
Adam	0.01	$1k$	8.4109×10^{-2}	7.8732×10^{-2}	7.3069×10^{-2}
Adam	0.01	$5k$	6.9011×10^{-2}	7.0638×10^{-2}	7.5214×10^{-2}
Adam	0.01	$10k$	7.9641×10^{-2}	7.2514×10^{-2}	7.5266×10^{-2}
Adam	0.001	$1k$	100+	100+	100+
Adam	0.001	$5k$	7.005×10^{-3}	7.285×10^{-3}	7.6783×10^{-3}
Adam	0.001	$10k$	5.032×10^{-3}	4.342×10^{-3}	4.018×10^{-3}
SGD	0.001	$1k$	1.2421	0.9722	0.6726
SGD	0.001	$5k$	1.5755×10^{-2}	1.4383×10^{-2}	1.4402×10^{-2}
SGD	0.001	$10k$	1.1992×10^{-2}	1.21523×10^{-2}	1×10^{-2}

Remarks

- all the experiments were done with constant h
- Thm does not apply here, we are trying to establish relative accuracy

Poisson's equation

1. ábra. Domain of the second problem

Auxiliary points

We are interested in the values $\{u(\mathbf{x}_i)\}_{i=1}^{I}$, where $X = \{\mathbf{x}_i\}_{i=1}^{I}$ are the $I = 15$ 'spring green' points in the inside. To approximate these values, let us define the following auxiliary sets:

$$Y = \{\mathbf{y}_j\}_{j=1}^{J} \subset \operatorname{ext}\Omega \quad Z = \{\mathbf{z}_k\}_{k=1}^{K} \subset \partial\Omega \quad W = \{\mathbf{w}_l\}_{l=1}^{L}.$$

Here, the "border distance" is 0.005, meaning that every point in Z is 0.005 units away from $\partial\Omega$.
Now, the task of the numerical approximation is to find a map A, for which

$$A\Big(\big(f(\mathbf{w}_l)\big)_{l=1}^{L}, \big(g(\mathbf{z}_k)\big)_{k=1}^{K}\Big) \approx \big(u(\mathbf{x}_i)\big)_{i=1}^{I}.$$

Setup of the method for the second problem

Take the input-output pairs as before:

$$\mathrm{NN}: \underbrace{\left(\mathbf{0}, G(\mathbf{z}_k, \mathbf{y}_j)_{k=1}^{K}\right)}_{\in\mathbb{R}^{L+K}} \mapsto \left(G(\mathbf{x}_i, \mathbf{y}_j)\right)_{i=1}^{I} \quad (\forall \mathbf{y}_j \in Y), \tag{3}$$

but also need to include the BC:

$$\mathrm{NN}: \left(\left(\Delta\psi(\|\mathbf{w}_l - \mathbf{y}_j\|_2)\right)_{l=1}^{L}, \left(\psi(\|\mathbf{z}_k - \mathbf{y}_j\|_2)\right)_{k=1}^{K}\right) \mapsto \left(\psi(\|\mathbf{x}_i - \mathbf{y}_j\|_2)\right)_{i=1}^{I} \tag{4}$$

where ψ is a radial basis function.

Exact solution

The exact solution chosen to be

$$u(x,y) = \underbrace{xy\sin(x+y^2)}_{\Delta(\dots):=f} + \underbrace{x^5 - 10x^3y^2 + 5xy^4}_{\Delta(\dots)=0},$$

and the exact solutions at the points of X are

$$\begin{aligned}\big(u(\mathsf{x}_i)\big)_{i=1}^{15} = \big(&-468.34, -420.25, -224.78, 31.685, 215.65,\\ &267.86, 225.76, 149.92, 73.672, 8.265,\\ &-49.81, -111.43, -188.82, -289.09, -398.99\big),\end{aligned}$$

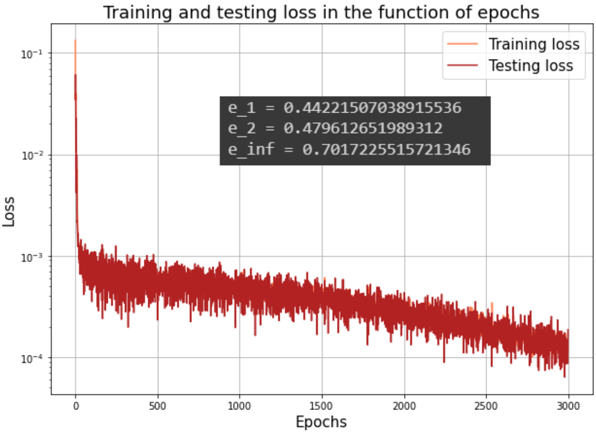
which visibly has a significant variance.

Experiments II-1

$K = 302$ boundary points, $L = 312$ inner training points
$\implies I \cdot (L + K) = 15 \cdot 612 = 9210$ parameters. Set $J = 9662$.
Single linear layer with no bias, $lr = 0.001$ 10,000 epochs, batch size $= 3,000$ – which is about 15% of the training data set) as in the first case yields a very noisy learning curve and poor numerical results (about 50% relative error).

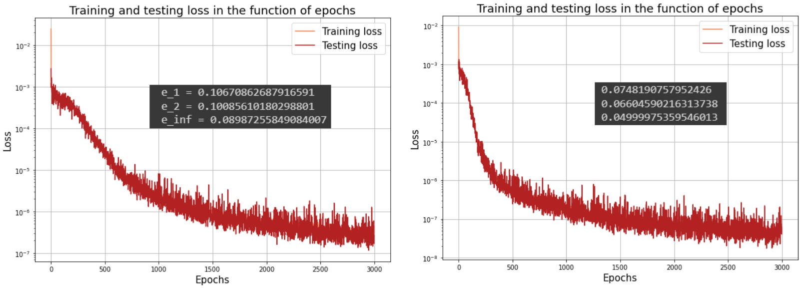
Experiments II-2/a

Increased number of points, including one more hidden layer with biases:



Experiments II-2/b

Further decreasing the batch sizes ≈ increasing epochs improved accuracy.



2. ábra. picking batch sizes of 1,024 and 512 respectively

Research questions

Research questions for the Thesis:

1. What NN structure and optimization method is best suitable for solving these kinds of problems? How can we obtain methods that have a more sensible running time – both for further experiments and application purposes.
2. What is the nature of the convergence hinted at by the above experiments? (Further experiments and perhaps even theoretical work to be done here.)
3. Can we show – either empirically, or theoretically – a better than $\mathcal{O}(h)$ rate of convergence in the case of Laplace's equation?
4. How might we further generalize the class of problems for which this method applies? One such setup is described in the last subsection.
5. How can we use the results obtained here in sensible applications? Can we perhaps use a trained NN to perform time-steps in a time-dependent setup?

Rescources

[1.] Csáji Balázs Csanád. "Approximation with Artificial Neural Networks". In: MSc Thesis, Eötvös Loránd Tudományegyetem, Természettudományi Kar (2001)

[2.] Haffner Domonkos and Izs ak Ferenc. "Solving the Laplace equation by using neural networks". In: url: https://m2.mtmt.hu/api/publication/32625402

[3.] T. Hieu Hoang, Ferenc Izs ak, and G abor Maros. "Approximation properties of fundamental solutions: a three-dimensional study with Sobolev norms". In: 2022.

[4.] Ge Ji and O. Isgor. "On the numerical solution of Laplace's equation with nonlinear boundary conditions for corrosion of steel in concrete". In: (Jan. 2006).

[5.] R. Schaback. A Practical Guide to Radial Basis Functions. url: http : / / num . math . uni -goettingen.de/ schaback/teaching/sc.pdf. (accessed: 15.12.2022).

[6.] Kurt Hornik, Maxwell Stinchcombe, Halbert White. "Multilayer Feedforward Networks are Universal Approximators". In: Neural Networks Vol. 2 (1989), pp. 359–366.

Thank you for your attention!