16th World Congress in Computational Mechanics (WCCM)
21-26 July 2024, Vancouver, Canada
A. Korobenko, M. Laforest, S. Prudhomme, R. Vaziri (Eds)

ACCURATE SOLUTION OF LINEAR OPERATOR APPROXIMATIONS USING GREEN'S FUNCTIONS BY A MULTI-LEVEL NEURAL NETWORK APPROACH

ZIAD ALDIRANY[1], CHARLÉLIE BILODEAU[2], RÉGIS COTTEREAU[3], MARC LAFOREST[4] AND SERGE PRUDHOMME[5]

[1] Département de mathématiques et de génie industriel, Polytechnique Montréal
Montréal, Québec, Canada, H3T 1J4
ziad.aldirany@polymtl.ca

[2] Département de mathématiques et de génie industriel, Polytechnique Montréal
Montréal, Québec, Canada, H3T 1J4
charlelie.bilodeau@polymtl.ca

[3] Aix-Marseille Université, CNRS, Centrale Marseille, LMA UMR 7031
Marseille, France
cottereau@lma.cnrs-mrs.fr

[4] Département de mathématiques et de génie industriel, Polytechnique Montréal
Montréal, Québec, Canada, H3T 1J4
marc.laforest@polymtl.ca

[5] Département de mathématiques et de génie industriel, Polytechnique Montréal
Montréal, Québec, Canada, H3T 1J4
serge.prudhomme@polymtl.ca

Keywords: Scientific Machine Learning, Neural Networks, Green's Functions, Numerical Accuracy.

Abstract. Lately, the approximation of operators for partial differential equations using deep learning has been extensively investigated. However, these deep learning approaches have limitations in terms of accuracy. In this work, we present a multi-level approach to accurately approximate linear operators using physics-informed Green operator networks. This method allows for the iterative reduction of the approximation errors through a sequence of operators, each targeting errors of increasing complexity at progressively smaller scales. Numerical examples for the one-dimensional Poisson problem will be presented to demonstrate the effectiveness of the proposed multi-level approach.

1 INTRODUCTION

The solution of boundary-value problems using deep learning approaches has been extensively investigated in recent years. These approaches are used to approximate either a single

boundary-value problem [3, 4, 5], or the operator associated with a family of boundary-value problems [2, 6, 7]. However, achieving high accuracy in the approximations obtained from these methods often remains a significant challenge. Recent works [1, 8, 9, 10] were successful in estimating and reducing the errors of the approximated solutions using a sequence of networks. The multi-level neural networks (MLNNs) approach proposed in [1] allows one to iteratively reduce the error by employing a sequence of networks when using physics-informed neural networks [4]. This is done by training a correction network of increasing complexity to reduce the remaining residual of the boundary value problem after each level.

The authors in [2], introduced the Green operator networks (GreenONets) in which the operator of the wave equation is approximated by learning the corresponding Green's function. In this work, we extend the multi-level approach to estimate and reduce the errors using Green Operator Networks. The main idea is to iteratively reduce the approximation error using a sequence of operators. This approach can be applied to other linear operators, but for simplicity, we will present it only for the one-dimensional Poisson problem. Given a source term $f(x)$ and a boundary condition $b(x)$, the objective is to find the solution $u(x)$, for all $x \in \Omega \subset \mathbb{R}$ satisfying

$$R\big(x, u\big) = f(x) + \partial_{xx} u(x) = 0, \qquad \forall x \in \Omega, \tag{1a}$$

$$B\big(x, u\big) = b(x) - u(x) = 0, \qquad \forall x \in \partial\Omega. \tag{1b}$$

To do so, we will approximate the operator of the Poisson problem for a family of source terms f and a family of boundary conditions b.

2 GREEN OPERATOR NETWORKS

We start by briefly reviewing the Green operator networks approach proposed in [2] and present it for the Poisson problem. For given Banach spaces $\mathcal{U}$ and $\mathcal{S}$, the objective is to learn the operator $Q : \mathcal{S} \to \mathcal{U}$ such that, for any input parameter $s \in \mathcal{S}$, $Q(s) \equiv u \in \mathcal{U}$ is the solution to the Poisson problem (1). In this work, the input parameter s will represent the source term f or the Dirichlet boundary condition b. We start by defining the input vector $[s(x_i)]_{i=1,\dots,m_s}$ with the input function evaluated at a collection of m_s points $\{x_i\}_{i=1}^{m_s}$, known as sensors. The GreenONet, inspired by the Green's function solution, aims to approximate the operator $Q(s)$ with the following architecture

$$\hat{Q}(s)(x) = \frac{1}{m_s} \sum_{i=1}^{m_s} G(x, x_i) s(x_i), \tag{2}$$

where G is the output of a simple feedforward neural network.

To train this operator, we consider a physics-informed loss, where the network is trained by penalizing the residuals associated with the governing partial differential equation and with the boundary conditions for a family of N input functions $\{s^{(i)}\}_{i=1}^{N}$. The input functions will be randomly sampled from a zero-mean Gaussian random field (GRF) with a length scale l, as

presented in [7]. The loss function in this case reads:

$$\mathcal{L}(\theta) = \frac{1}{N}\sum_{i=1}^{N}\left[w_r \int_\Omega R\big(x, \hat{Q}(s^{(i)})(x)\big)^2 dx + w_{bc}\int_{\partial\Omega} B\big(x, \hat{Q}(s^{(i)})(x)\big)^2 dx\right]. \tag{3}$$

3 MULTI-LEVEL NEURAL NETWORKS WITH GREEN OPERATOR NETWORKS

In this section, we extend the multi-level approach introduced in [1] to Green operator networks. A straightforward approach is to approximate the Green's function with an initial network as presented with GreenONets, then add further corrections to the Green's function to continuously reduce the residual of the training functions. Although this method seems promising, it suffers from overfitting as one needs a very large family of training functions to generalize well on small scales. To overcome this issue, we propose to approximate multiple operators, each trained with a family of input functions of increasing complexity. First, we present this approach for the Poisson problem with strongly imposed boundary conditions and then extend it to weakly imposed boundary conditions.

3.1 STRONGLY IMPOSED BOUNDARY CONDITIONS

We start by considering the Poisson problem with zero Dirichlet boundary condition, thus $b(x) = 0$, $x \in \partial\Omega$, in (1b). Our objective is to approximate the operator Q_f that solves the Poisson problem for a family of source terms $f(x)$, i.e. $s \equiv f$. The boundary conditions in Section 2 are weakly imposed and are not exactly verified. For simplicity, we consider in this paragraph strongly imposed boundary conditions by multiplying the output of the network by a function $g(x)$ that vanishes on the boundary. Hence, the operator Q_f is approximated by $\hat{Q}_f$ defined by (2) with G given as

$$G(x, x_i) = g(x)J(x, x_i), \tag{4}$$

where $J(x, x_i)$ is the output of a feedforward neural network. Therefore, the loss can be given as

$$\mathcal{L}(\theta) = \frac{1}{N}\sum_{i=1}^{N}\int_\Omega R\big(x, \hat{Q}_f(f^{(i)})(x)\big)^2 dx, \tag{5}$$

where the $f^{(i)}$, $i = 1, 2, \ldots, N$, are the different training source terms. Similar to what has been observed in [1], the errors would remain large if one uses only one level. In order to control and reduce the errors when approximating a solution with GreenONets, we shall consider the following approach:

1. Approximate the operator Q_f with the GreenONets, $\hat{Q}_{f,0}, \ldots, \hat{Q}_{f,L}$, by training each independently with source terms, created with GRF of decreasing length scales, i.e. increasing frequencies. In other words, the GreenONet $\hat{Q}_{f,k}$, $k = 0, 1, \ldots, L$, is trained using a GRF of length scale l_k, where $l_k < l_j$ for $k > j$.

2. Calculate the initial approximation to u with $\tilde{u}_0 = \hat{Q}_{f,0}(f_0)$, where $f_0 = f$.

3. Calculate the first correction $\tilde{u}_1 = \hat{Q}_{f,1}(f_1)$, where $f_1 = f_0 + \partial_{xx}\tilde{u}_0$, which corresponds to the PDE residual of the initial approximation.

4. Repeat this process and calculate the corrections $\tilde{u}_k = \hat{Q}_{f,k}(f_k)$, where $f_k = f_{k-1} + \partial_{xx}\tilde{u}_{k-1}$, which corresponds to the PDE residual of the previous approximation.

After obtaining the initial solution $\tilde{u}_0(x)$ and the corrections $\tilde{u}_k$, $k = 1, \ldots, L$, the final approximation is obtained as

$$\tilde{u}(x) = \sum_{k=0}^{L} \tilde{u}_k(x).$$

Similar to the classical MLNN approach, the errors at higher levels exhibit higher frequencies. This justifies the training of GreenONets, $\hat{Q}_{f,0}, \ldots, \hat{Q}_{f,L}$, using input functions of increasing frequencies. We note that the source terms are not normalized here, as done in the classical multi-level approach. Actually, the correction with a GreenONet at the $k\text{th}$ level is given as

$$\tilde{u}_k(x) = \hat{Q}_{f,k}(f_k)(x) = \frac{1}{m_s} \sum_{i=1}^{m_s} G_k(x, x_i) f_k(x_i).$$

Since the architecture of GreenONets is scale-independent, the solution is naturally scaled with the source term f_k. In other words, if the source term has a small magnitude, the solution inherits the same property without the need to rescale the source term. Furthermore, this approach minimizes the error in the approximation by reducing the residual of the PDE, which constitutes the only source of error here. Nonetheless, in cases where boundary conditions are weakly imposed, an operator for these conditions is necessary to further reduce the error associated with the boundary term at each level.

3.2 WEAKLY IMPOSED BOUNDARY CONDITIONS

In the following, our goal is to approximate the solution of the Poisson problem for a family of source terms $f(x)$ and boundary conditions $b(x)$. The boundary conditions will be weakly imposed, and the GreenONets are trained by minimizing the loss (3). Due to the superposition principle of linear operators, we will approximate the operators Q_f and Q_b separately, which respectively take as input the source term and the boundary condition. In other words, we need to approximate the operator Q_f, that solves:

$$R_f\big(x, Q_f(f)\big) = f(x) + \partial_{xx} Q_f(f)(x) = 0, \qquad \forall x \in \Omega, \tag{6a}$$

$$B_f\big(x, Q_f(f)\big) = Q_f(x) = 0, \qquad \forall x \in \partial\Omega, \tag{6b}$$

and the operator Q_b that solves:

$$R_b\big(x, Q_b(b)\big) = \partial_{xx} Q_b(b)(x) = 0, \qquad \forall x \in \Omega, \tag{7a}$$

$$B_b\big(x, Q_b(b)\big) = b(x) - Q_b(b)(x) = 0, \qquad \forall x \in \partial\Omega. \tag{7b}$$

Thus the solution verifying Problem (1) can be obtained as $u(x) = Q_b(b)(x) + Q_f(f)(x)$.

In order to apply the multi-level approach, one can simultaneously reduce the PDE and the boundary condition residuals by superposing the two operators. Hence, the multi-level approach for weak boundary conditions is given as:

1. Approximate the operator Q_f with the GreenONets, $\hat{Q}_{f,0}, \ldots, \hat{Q}_{f,L}$, by training each independently with source terms, created with GRF of increasing complexity, i.e. decreasing length scale.

2. Approximate the operator Q_b with the GreenONet $\hat{Q}_b$, by training it for a family of boundary conditions. *We note that since we are considering a 1D problem the boundary consists of two points, hence the concept of complexity of the boundary function does not exist and one operator is enough.*

3. Calculate the initial approximation to u with $\tilde{u}_0 = \hat{Q}_{f,0}(f_0) + \hat{Q}_b(b_0)$, where $f_0 = f$ and $b_0 = b$.

4. Calculate the first correction by $\tilde{u}_1 = \hat{Q}_{f,1}(f_1) + \hat{Q}_b(b_1)$, where $f_1 = f_0 + \partial_{xx}\tilde{u}_0$ and $b_1 = b_0 - \tilde{u}_0$.

5. Repeat this process and calculate the corrections $\tilde{u}_k = \hat{Q}_{f,k}(f_k) + \hat{Q}_b(b_k)$, where $f_k = f_{k-1} + \partial_{xx}\tilde{u}_{k-1}$ and $b_k = b_{k-1} - \tilde{u}_{k-1}$.

Finally, the approximation is obtained as

$$\tilde{u}(x) = \sum_{k=0}^{L} \tilde{u}_k(x).$$

4 NUMERICAL EXAMPLE

In the following section, we want to accurately solve the one-dimensional Poisson problem using the multi-level GreenONets. In the first example, we consider strongly imposed boundary conditions following the approach presented in Section 3.1. And for the second example, the boundary conditions are weakly imposed and the solution is approximated as described in Section 3.2.

4.1 STRONGLY IMPOSED BOUNDARY CONDITIONS

In this example, we will consider the Poisson equation with homogeneous Dirichlet boundary conditions. The boundary conditions are strongly imposed by multiplying the output of the network by the function $g(x) = x(1-x)$. We first train seven Green operator networks $\hat{Q}_{f,k}$, $k = 0, \ldots, 6$, with the following length scales $l_k \in \{1, 0.5, 0.2, 0.1, 0.05, 0.02, 0.01\}$. For all networks, we choose $N = 5{,}000$ training functions. The sensor points are uniformly distributed on $(0, 1)$ with $m_s = 201$. Each GreenONet is trained using Adam followed by L-BFGS with the learning rates 10^{-3} and unity respectively. For higher levels, the complexity of the network and

the number of iterations are increased, as shown in Table 1, to approximate more complicated solutions. Note that each L-BFGS iteration consists of 20 sub-iterations as set up by default in the library PyTorch.

Table 1: Hyperparameters used in the example of Section 4.

Hyperparameters	$\hat{Q}_{f,0}$	$\hat{Q}_{f,1}$	$\hat{Q}_{f,2}$	$\hat{Q}_{f,3}$	$\hat{Q}_{f,4}$	$\hat{Q}_{f,5}$	$\hat{Q}_{f,6}$
# Hidden layers n	2	2	2	3	3	3	4
Widths of the hidden layers	20	20	30	30	40	50	50
# Adam iterations	5,000	5,000	5,000	5,000	5,000	5,000	5,000
# L-BFGS iterations	300	300	300	300	300	400	400

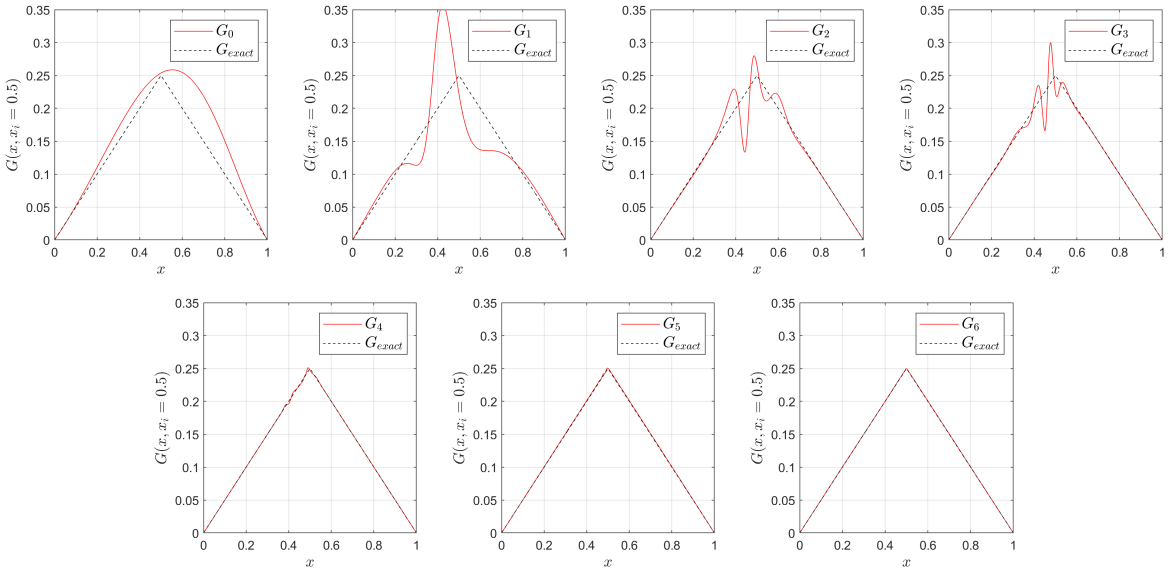

Figure 1: Example of Section 4.1: Green's function $G_k(x, x_i = 0.5)$, $k = 0, \ldots, 6$, compared to the exact Green's function.

In Figure 1, we show the approximation of the Green's function $G_k(x, x_i = 0.5)$, $k = 0, \ldots, 6$, when trained with different length scales. We observe that when the GreenONet is trained with higher frequencies, the approximated Green's function converges to the exact Green's function of the Poisson problem. After training the operators we test our approach with the source function

$$f(x) = (m\pi)^2 \sin(m\pi x),$$

corresponding to the exact solution $u(x) = \sin(m\pi x)$. We start by considering the case with $m = 1$. As described in the multi-level GreenONets, we compute the initial approximation $\tilde{u}_0$

and the corrections $\tilde{u}_k$, $k = 1, \ldots, 6$, using the corresponding Green's functions G_k. In Figure 2, we show the pointwise error $e_k(x) = u(x) - \sum_{j=0}^{k} \tilde{u}_j(x)$, for each level. We observe that the maximum pointwise error is around 3×10^{-2} after the initial approximation, then it decreases after each correction to attain a maximum pointwise error around 7×10^{-9}. Moreover, as observed in MLNNs, larger gradients appear in the errors as more corrections are introduced. For this reason, one cannot correct the solution with the same operator, so an operator that handles higher frequencies is needed.

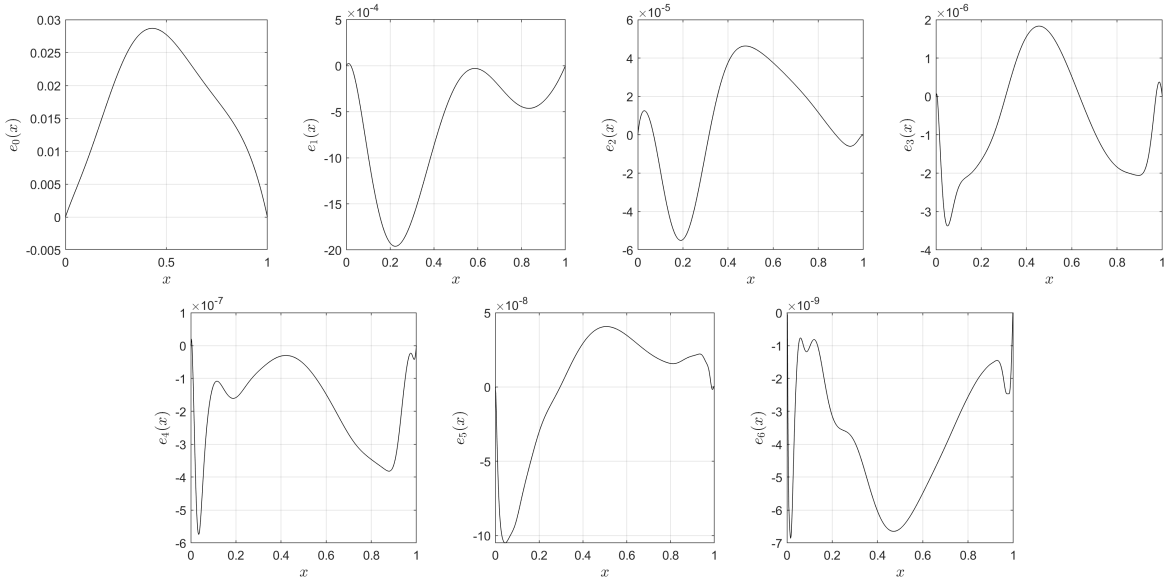

Figure 2: Example of Section 4.1: Errors $e_k(x) = u(x) - \sum_{j=0}^{k} \tilde{u}_j(x)$, at each level.

Additionally, we test the same networks with different values of m. We show in Figure 3 (left) the evolution of the L^2 error when approximating the solutions with the multi-level GreenONets. We also report in Figure 3 (right) the L^2 error of the solutions approximated only with the GreenONet at level k (we do not have a sequential correction here). In other words, at level k, the approximated solution is computed solely with G_k, such as

$$\tilde{u}(x) = \tilde{u}_k(x) = \frac{1}{m_s} \sum_{i=1}^{m_s} G_k(x, x_i) f(x_i).$$

We observe that for $m \in \{1, 4, 7, 10\}$ we were able to reduce the L^2 error using the multi-level approach compared to the approximation with a single network G_k, for any k. For example, for $m = 7$ we can attain a minimum L^2 error of the order of 10^{-5} with the multi-level GreenONets. On the other hand, the best approximation for $m = 7$ with only a single network yields an L^2 error of the order of 10^{-2}.

We notice in Figure 3 (left) that for $m = 7$ and $m = 10$ the error increases at the early levels and then it starts to decrease when the frequencies of the solution are captured. Therefore,

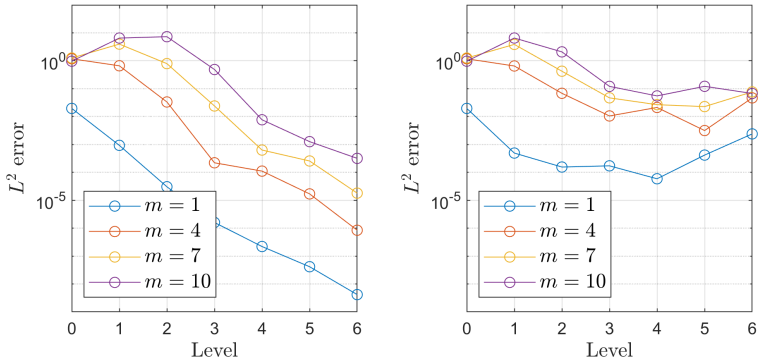

Figure 3: Example of Section 4.1: (left) Evolution of the L^2 error using the multi-level GreenONet approach for different values of m. (right) L^2 error of the solution when approximated solely with the GreenONet at level k (without sequential correction) for different values of m.

one can start the initial approximation at a subsequent level to avoid the error introduction at early levels. In Figure 4, we show the evolution of the residual of the PDE and the error in the L^2 norm with the initial approximation at level 0 or level 3. We observe that, since the latter approach avoids the introduction of errors in the early levels, we were able to reduce both the error and the residual by around an additional order of magnitude. Therefore, one should start with the first level that reduces the residual to avoid introducing errors in early levels.

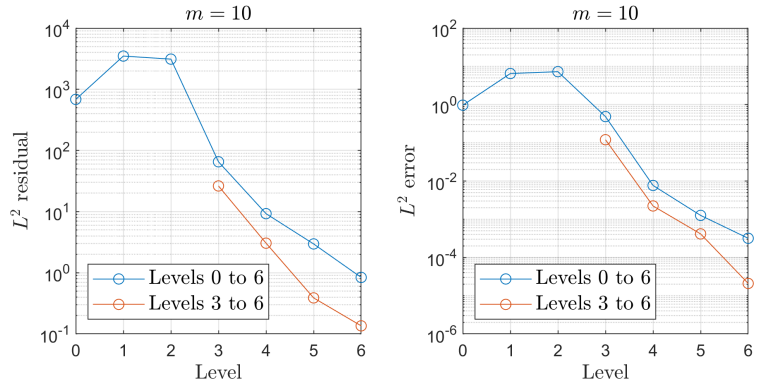

Figure 4: Example of Section 4.1: (left) Evolution of the residual in the L^2 norm using the multi-level GreenONet for $m = 10$ when starting the training from either level 0 or level 3. (right) Evolution of the L^2 error using the multi-level GreenONet for $m = 10$ when starting the training from either level 0 or level 3.

Moreover, to show that our sequence of neural networks is not specific to the problem with $f(x) = (m\pi)^2 \sin(m\pi x)$, we test the same operators with a source term obtained from a GRF with a length scale $l = 0.1$, see Figure 5 (left). The approximated solution is calculated using the multi-level GreenONets and is presented in Figure 5 (middle). Finally, we show in Figure 5 (right) the evolution of the L^2 error after each correction. We note here that the error is calculated between the approximated solution and an over-kill solution calculated with the finite

difference method. We observe that the L^2 error starts to decrease at level 2 and is of the order of 10^{-7} after all the corrections.

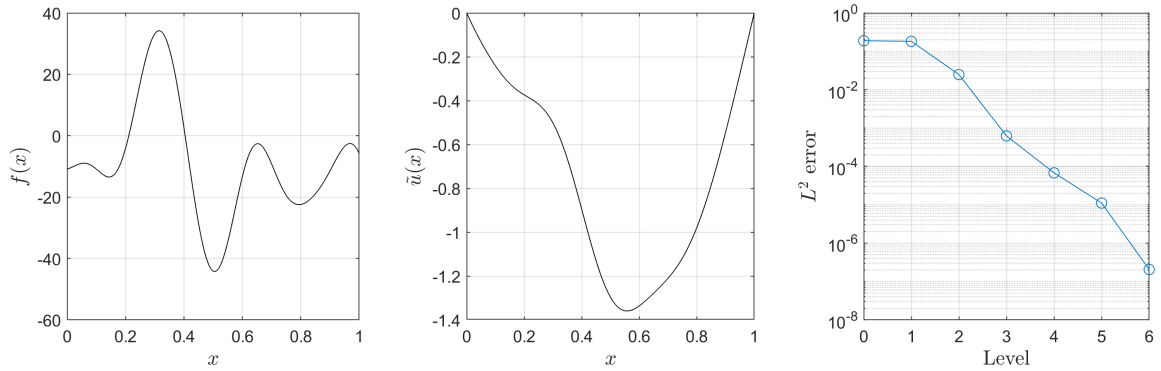

Figure 5: Example of Section 4.1: (left) Source term. (middle) Approximated solution $\tilde{u}(x) = \sum_{k=0}^{6} \tilde{u}_k(x)$. (right) Evolution of the L^2 error using the multi-level GreenONets.

4.2 WEAKLY IMPOSED BOUNDARY CONDITIONS

In this section, we present a numerical example to validate the approach presented in Section 3.2. Similar to the previous example, we train seven GreenONets $\hat{Q}_{f,k}$ using the same hyper-parameters, with the difference being that the boundary conditions are weakly imposed. Hence, the network is not multiplied by $g(x)$ and the loss (3) is used. Additionally, the operator $\hat{Q}_b$ is trained with a network of width 20 and depth 2 for 25,000 Adam iterations only. The training is completed using 1,000 training function $b(x)$ defined with a uniform distribution between 0 and 1. For the sake of clarity, 1,000 couples $(b(0), b(1))$ were generated. Following the training, we test our approach with the same source function as before $f(x) = (m\pi)^2 \sin(m\pi x)$ and $b(x) = 0$.

We show in Figure 6 the evolution of error, PDE residual, and boundary conditions residual in the L^2 norm at each level for $m \in \{1, 4, 7, 10\}$, when the multi-level approach is used. Similar to the previous example, we achieved a significant error reduction by several orders of magnitude by minimizing both the PDE residual and the boundary conditions residual. However, we observe that the error and the boundary residual increase from level 4 to 5 for most cases. Further numerical experiments are necessary to better understand this error behaviour for strongly and weakly imposed boundary conditions.

5 CONCLUSIONS

In this work, we extended the multi-level approach to GreenONets for the Poisson problem. By approximating a sequence of Green's functions for input functions of increasing complexity, we were able to sequentially reduce the errors when approximating the solution for a new source term. We demonstrated the effectiveness of the multi-level GreenONets with source terms of

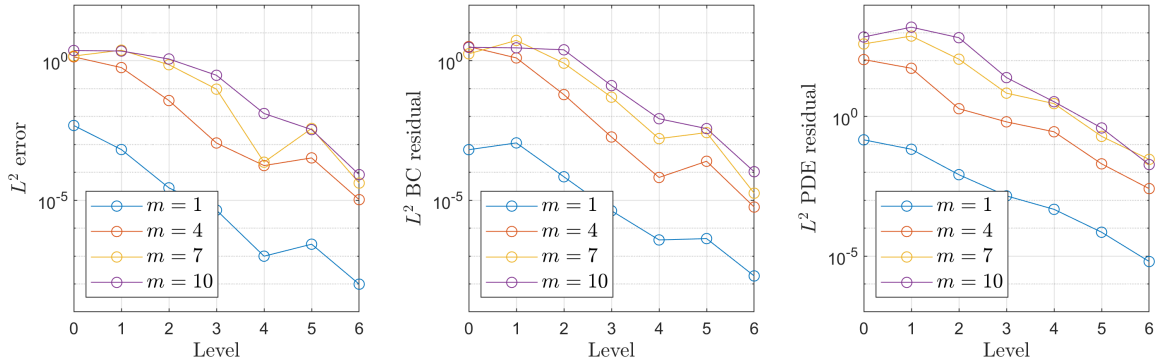

Figure 6: Example of Section 4.2: Evolution of the error (left), the boundary condition residual (middle), and the PDE residual (right) in the L^2 norm, using the multi-level GreenONet approach for different values of m.

varying frequencies, achieving significant error reduction in all cases. These results highlight the substantial benefits of extending the multi-level approach to operator approximation, where error reduction is a primary challenge for further development. These findings should be further extended to approximate different initial and boundary value problems in various dimensions with high precision. Additionally, one could explore the approximation of the Green's functions, $G_1, \ldots, G_L$, using transfer learning [11], where each network G_i is initialized with the parameters of the trained network G_{i-1}. Finally, exploring the extension of the multi-level approach to various neural operator methods presents an intriguing avenue for future work.

REFERENCES

[1] Aldirany, Z., Cottereau, R., Laforest, M., and Prudhomme, S. Multi-level neural networks for accurate solutions of boundary-value problems. *Computer Methods in Applied Mechanics and Engineering*, (2024) **419**:116666.

[2] Aldirany, Z., Cottereau, R., Laforest, M., and Prudhomme, S. Operator approximation of the wave equation based on deep learning of Green's function. *Computers & Mathematics with Applications*, (2024) **159**:21-30.

[3] Lagaris, I.E., Likas, A., and Fotiadis, D.I. Artificial neural networks for solving ordinary and partial differential equations. *IEEE Transactions on Neural Networks*, (1998) **9**(5):987-1000.

[4] Raissi, M., Perdikaris, P., and Karniadakis, G.E. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, (2019) **378**:686-707.

[5] Uriarte, C., Pardo, D., Muga, I., and Muñoz-Matute, J. A deep double Ritz method (D2RM) for solving partial differential equations using neural networks. *Computer Methods in Applied Mechanics and Engineering*, (2023) **405**:115892.

[6] Li, Z., Kovachki, N., Azizzadenesheli, K., Liu, B., Bhattacharya, K., Stuart, A., and Anandkumar, A. Fourier neural operator for parametric partial differential equations. *arXiv preprint arXiv:2010.08895*, (2020).

[7] Lu, L., Jin, P., Pang, G., Zhang, Z., and Karniadakis, G.E. Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators. *Nature Machine Intelligence*, (2021) **3**(3):218-229.

[8] Ainsworth, M., and Dong, J. Galerkin neural networks: A framework for approximating variational equations with error control. *SIAM Journal on Scientific Computing*, (2021) **43**(4):A2474-A2501.

[9] Howard, A.A., Murphy, S.H., Ahmed, S.E., and Stinis, P. Stacked networks improve physics-informed training: applications to neural networks and deep operator networks. *arXiv preprint arXiv:2311.06483*, (2023).

[10] Wang, Y., and Lai, C.-Y. Multi-stage neural networks: Function approximator of machine precision. *Journal of Computational Physics*, (2024):112865.

[11] Weiss, K., Khoshgoftaar, T.M., and Wang, D. A survey of transfer learning. *Journal of Big Data*, (2016) **3**:1-40.