Large-Scale Neural Solvers for Partial Differential Equations

Patrick Stiller[1,2(✉)], Friedrich Bethke[1,2], Maximilian Böhme[3], Richard Pausch[1], Sunna Torge[2], Alexander Debus[1], Jan Vorberger[1], Michael Bussmann[1,3], and Nico Hoffmann[1]

[1] Helmholtz-Zentrum Dresden-Rossendorf, Dresden, Germany
p.stiller@hzdr.de
[2] Technische Universität Dresden, Dresden, Germany
[3] Center for Advanced Systems Understanding (CASUS), Görlitz, Germany

Abstract. Solving partial differential equations (PDE) is an indispensable part of many branches of science as many processes can be modelled in terms of PDEs. However, recent numerical solvers require manual discretization of the underlying equation as well as sophisticated, tailored code for distributed computing. Scanning the parameters of the underlying model significantly increases the runtime as the simulations have to be cold-started for each parameter configuration. Machine Learning based surrogate models denote promising ways for learning complex relationship among input, parameter and solution. However, recent generative neural networks require lots of training data, i.e. full simulation runs making them costly. In contrast, we examine the applicability of continuous, mesh-free neural solvers for partial differential equations, physics-informed neural networks (PINNs) solely requiring initial/boundary values and validation points for training but no simulation data. The induced curse of dimensionality is approached by learning a domain decomposition that steers the number of neurons per unit volume and significantly improves runtime. Distributed training on large-scale cluster systems also promises great utilization of large quantities of GPUs which we assess by a comprehensive evaluation study. Finally, we discuss the accuracy of GatedPINN with respect to analytical solutions- as well as state-of-the-art numerical solvers, such as spectral solvers.

1 Introduction

Scientific neural networks accelerate scientific computing by data-driven methods such as physics-informed neural networks. One such prominent application is surrogate modelling which is e.g. used in particle physics at CERN [1]. Enhancing neural networks by prior knowledge about the system makes the prediction more robust by regularizing either the predictions or the training of neural networks. One such prominent approach is a physics-informed neural network (PINN) which makes use of either learning [2] or encoding the governing equations of a physical system into the loss function [3] of the training procedure.

J. Nichols et al. (Eds.): SMC 2020, CCIS 1315, pp. 20–34, 2020.
https://doi.org/10.1007/978-3-030-63393-6_2

Surrogate models based on PINN can be seen as a *neural solvers* as the trained PINN predicts the time-dependent solution of that system at any point in space and time. Encoding the governing equations into the training relies on automatic differentiation (AD) as it is an easy computing scheme for accessing all partial derivatives of the system. However, AD also constrains the neural network architecture to use C^{k+1} differentiable activation functions provided the highest order of derivatives in the governing system is k. Furthermore, the computational cost increases with the size of the neural network as the whole computational graph has to be evaluated for computing a certain partial derivative. The main contribution of this paper is three-fold. First, we introduce a novel 2D benchmark dataset for surrogate models allowing precise performance assessment due to analytical solutions and derivatives. Second, we improve the training time by incorporating and learning domain decompositions into PINN. Finally, we conduct a comprehensive analysis of accuracy, power draw and scalability on the well known example of the 2D quantum harmonic oscillator.

2 Related Works

Accelerated simulations by surrogate modelling techniques are carried out in two main directions. Supervised learning methods require full simulation data in order to train some neural network architecture, e.g. generative adversarial networks [1] or autoencoders [4], to reproduce numerical simulations and might benefit from interpolation between similar configurations. The latter basically introduces a speedup with respect to numerical simulations, however generalization errors might challenge this approach in general. In contrast, self-supervised methods either embed neural networks within numerical procedures for solving PDE [5], or incorporate knowledge about the governing equations into the loss of neural networks, so called physics-informed neural networks (PINN) [3]. The latter is can be seen as variational method for solving PDE. Finally, [2] demonstrated joint discovery of a system (supervised learning) and adapting to unknown regimes (semi-supervised learning). Recently, [6] proved convergence of PINN-based solvers for parabolic and hyperbolic PDEs. Parareal physics-informed neural networks approach domain decomposition by splitting the computational domain into temporal slices and training a PINN for each slice [7]. We are going to generalize that idea by introducing conditional computing [8] into the physics-informed neural networks framework, hereby enabling an arbitrary decomposition of the computational domain which is adaptively tuned during training of the PINN.

3 Methods

The governing equations of a dynamic system can be modeled in terms of non-linear partial differential equations

$$u_t + \mathcal{N}(u; \lambda) = 0 \ ,$$

with $u_t = \frac{\partial u}{\partial t}$ being the temporal derivative of the solution u of our system while $\mathcal{N}$ denotes a non-linear operator that incorporates the (non-)linear effects of our system. One example of such a system is the quantum harmonic oscillator,

$$i\frac{\partial \psi(\mathbf{r}, t)}{\partial t} - \hat{H}\psi(\mathbf{r}, t) = 0 \ ,$$

where $\psi(\mathbf{r}, t)$ denotes the so-called state of the system in the spatial base and $\hat{H}$ is the Hamilton-operator of the system. The systems state absolute square $|\psi(\mathbf{r}, t)|^2$ is interpreted as the probability density of measuring a particle at a certain point $\mathbf{r}$ in a volume $\mathcal{V}$. Thus, $|\psi(\mathbf{r}, t)|^2$ has to fulfill the normalization constraint of a probability density

$$\int_{\mathcal{V}} d^3r \, |\psi(\mathbf{r}, t)|^2 = 1.$$

The Hamilton operator of a particle in an external potential is of the form

$$\hat{H} = -\frac{1}{2}\Delta + V(\mathbf{r}, t),$$

where Δ is the Laplace operator and $V(\mathbf{r}, t)$ is a scalar potential. The first term is the kinetic energy operator of the system and $V(\mathbf{r}, t)$ its potential energy. In this work, we use the atomic unit system meaning that $\hbar = m_e = 1$. $\hat{H}$ is a Hermitian operator acting on a Hilbertspace $\mathcal{H}$. In this work we are focusing on the 2D quantum harmonic oscillator (QHO), which is described by the Hamiltonian

$$\hat{H} = -\frac{1}{2}\left(\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}\right) + \frac{\omega_0^2}{2}(x^2 + y^2) = \hat{H}_x + \hat{H}_y.$$

where $x \in \mathbb{R}$ and $y \in \mathbb{R}$ denote spatial coordinates. The solution of the QHO can be determined analytically and is the basis for complicated systems like the density function theory (DFT). Therefore the QHO is very well suited as a test system which allows a precise evaluation of the predicted results. In addition, the QHO can also be used as a test system for evaluating the results. Furthermore, the QHO is classified as linear parabolic PDE, which guarantees the functionality of the chosen PINN approach according to Shin et al. [6]. Figure 1 shows the analytic solution of the quantum harmonic oscillator over time.

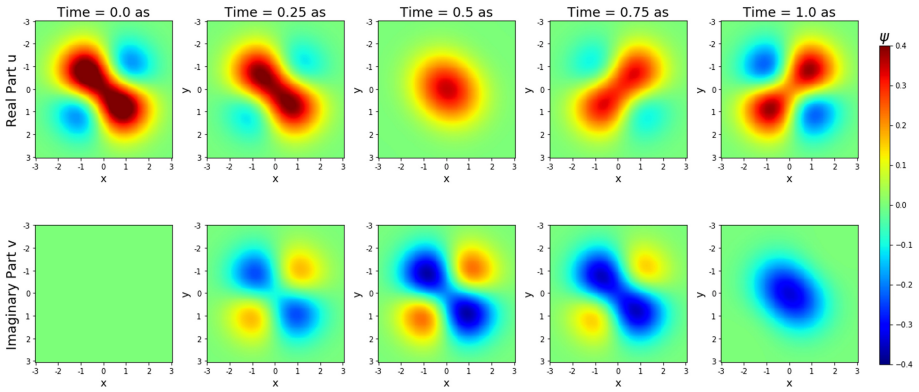


Fig. 1. Analytic solution of the quantum harmonic oscillator

3.1 Physics-Informed Quantum Harmonic Oscillator

The solution $\psi(x, y, t)$ of our quantum harmonic oscillator at some position x, y and time t is approximated by a neural network $f : \mathbb{R}^3 \to \mathbb{C}$, i.e.

$$\widehat{\psi}(x, y, t) = f(x, y, t) \ .$$

In this work, we model f by a simple multilayer perceptron (MLP) of $1 \leq l \leq m$ layers, a predetermined number of neurons per layer k_l and respective weight matrices $W^l \in \mathbb{R}^{k_l \times k_l}$

$$y^l = g(W^l y^{l-1}) \ ,$$

with $y^0 = (x, y, t)$ and $y^m = \widehat{\psi}(x, y, t)$. The training of Physics-informed neural networks relies on automatic differentiation which imposes some constraints on the architecture. In our case, the network has to be 3 times differentiable due to the second-order partial derivatives in our QHO (Eq. 3). This is achieved by choosing at least one activation function g which fulfills that property (e.g. tanh). The training of the neural network is realized by minimizing the combined loss $\mathcal{L}$ defined in Eq. (2). The three terms of $\mathcal{L}$ relate to the error of representing the initial condition L_0, the fulfillment of the partial differential equation L_f as well as boundary condition L_b.

$$\mathcal{L} = \alpha L_0(\mathcal{T}_0) + L_f(\mathcal{T}_f) + L_b(\mathcal{T}_b) \tag{1}$$

$\mathcal{L}_0$ is the summed error of predicted real- $u = real(\psi)$ and imaginary- $v = imag(\psi)$ of the initial state with respect to groundtruth real- u^i and imaginary part v^i at points $\mathcal{T}_r$. We introduce a weighting term α into $\mathcal{L}$ allowing us to emphasize the contribution of the initial state.

$$L_0(\mathcal{T}_0) = \frac{1}{|\mathcal{T}_0|} \sum_{i=1}^{|\mathcal{T}_0|} \left| u\left(t_0^i, x_0^i, y_0^i\right) - u^i \right|^2 + \frac{1}{|\mathcal{T}_0|} \sum_{i=1}^{|\mathcal{T}_0|} \left| v\left(t_0^i, x_0^i, y_0^i\right) - v^i \right|^2$$

The boundary conditions (Eq. 3) are modelled in terms of L_b at predetermined spatial positions T_b at time t.

$$L_b\left(T_b, t\right) = 1 - \left(\iint_{T_b} \left(u(t,x,y)^2 + v(t,x,y)^2 \right) dxdy \right)^2$$

$\mathcal{L}_f$ is divided into real- and imaginary part, such that f_u represents the correctness of the real- and f_v the correctness of imaginary part of the predicted solution. This loss term is computed on a set $\mathcal{T}_f$ of randomly distributed *residual points* that enforce the validity of the PDE at residual points $\mathcal{T}_f$.

$$L_f(\mathcal{T}_f) = \frac{1}{|\mathcal{T}_f|} \sum_{i=1}^{|\mathcal{T}_f|} \left| f_u\left(t_f^i, x_f^i, y_f^i\right) \right|^2 + \frac{1}{|\mathcal{T}_f|} \sum_{i=1}^{|\mathcal{T}_f|} \left| f_v\left(t_f^i, x_f^i, y_f^i\right) \right|^2$$

$$f_u = -u_t - \frac{1}{2} v_{xx} - \frac{1}{2} v_{yy} + \frac{1}{2} x^2 v + \frac{1}{2} y^2 v$$

$$f_v = -v_t + \frac{1}{2} u_{xx} + \frac{1}{2} u_{yy} - \frac{1}{2} x^2 u - \frac{1}{2} y^2 u$$

3.2 GatedPINN

Numerical simulations typically require some sort of domain decomposition in order to share the load among the workers. physics-informed neural networks basically consist of a single multilayer perceptron network f which approximates the solution of a PDE for any input (x, y, t). However, this also implies that the capacity of the network per unit volume of our compute domain increases with the size of the compute domain. This also implies that the computational graph of the neural network increases respectively meaning that the time and storage requirements for computing partial derivatives via automatic differentiation increases, too. This limits the capacity of recent physics-informed neural network.

We will be tackling these challenges by introducing conditional computing into the framework of physics-informed neural networks. Conditional Computing denotes an approach that activate only some units of a neural network depending on the network input [9]. A more intelligent way to use the degree of freedom of neural networks allows to increase the network capacity (degree of freedom) without an immense blow up of the computational time [8]. [7] introduced a manual decomposition of the compute domain and found that the capacity of the neural network per unit volume and thus the training costs are reduced. However, this approach requires another coarse-grained PDE solver to correct predictions. A decomposition of the compute domain can be learned by utilizing the mixture of expert approach [8] based on a predetermined number of so-called experts (neural networks). A subset k of all N experts are active for any point in space and time while the activation is determined by gating network which introduces an adaptive domain decomposition. The combination of mixture of experts and physics-informed neural networks leads to a new architecture called *GatedPINN*.

Architecture. The architecture comprises of a gating network $G(x, y, t)$ that decides which expert $E_i(x, y, t)$ to use for any input (x, y, t) in space and time (see Fig. 2). Experts E_i with $1 \leq i \leq N$ are modelled by a simple MLP consisting of linear layers and tanh activation functions. The predicted solution $\widehat{\psi}$ of our quantum harmonic oscillator (QHO) becomes a weighted sum of expert predictions E_i

$$\widehat{\psi}(x, y, t) = \sum_{i=1}^{N} G(x, y, t)_i \cdot E_i(x, y, t) \ .$$

GatedPINN promise several advantages compared to the baseline PINN: First, the computation of partial derivatives by auto differentiation requires propagating information through a fraction k/N of the total capacity of all experts. That allows to either increase the computational domain and/or increase the overall capacity of the neural network without a blow up in computational complexity.

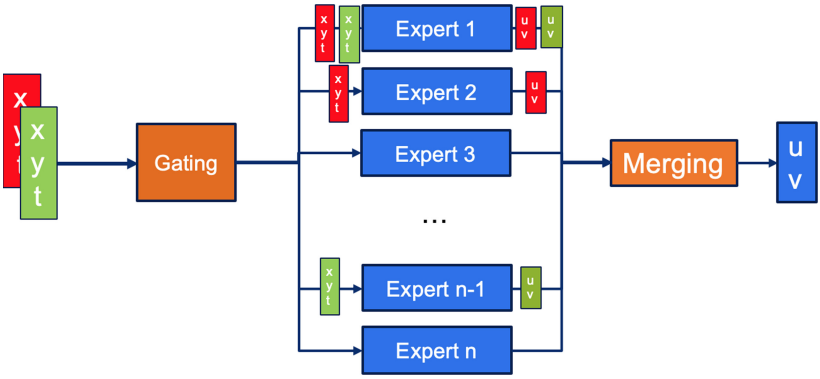


Fig. 2. Visualization of the Gated-PINN architecture

Similarly to [8], an importance loss $L_I = w_{\mathrm{I}} \cdot CV(I(x, y, t))^2$ penalizes uneven distribution of workload among all N experts:

$$L(\mathcal{T}, \theta) = L_0(\mathcal{T}_0, \theta) + L_f(\mathcal{T}_f, \theta) + L_b(\mathcal{T}_b, \theta) + \sum_{(x,y,t) \in T} L_I(X) \ , \tag{2}$$

given $T = T_0 \cup T_b \cup T_f$. The importance loss $L_I(X)$ requires the computation of an importance measure $I(X) = \sum_{x \in X} G(x, y, t)$. The coefficient of variation $CV(z) = \sigma(z)/\mu(z)$ provided $I(X)$ quantifies the sparsity of the gates and thus the utilization of the experts. Finally, coefficient w_I allows us to weight the contribution of our importance loss with respect to the PDE loss. The importance loss is defined as follows:

$$L_I(X) = w_I \cdot CV(\mathrm{I}(X))^2 \ .$$

Adaptive Domain Decomposition. A trainable gating network G allows us to combine the predictions of k simple neural networks for approximating the solution of our QHO at any point in space x, y and time t. Hereby, we restrict the size of the computational graph to k-times the size of each individual neural network E^i with $0 \leq i \leq k$.

$$G(x, y, t) = \text{Softmax}(\text{KeepTopK}(H(x, y, t, \omega)))$$

and basically yields a N dimensional weight vector with k non-zero elements [8]. The actual decomposition is learnt by the function H:

$$H(x, y, t) = ([x, y, t] \cdot W_g) + \text{StandardNormal}() \cdot \text{Softplus}(([x, y, t]^T \cdot W_{noise})) \ .$$

The noise term improves load balancing and is deactivated when using the model. Obviously, this gating results in a decomposition into linear subspaces due to W_g. Non-linear domain decomposition can now be realized by replacing the weight matrix W_g by a simple MLP NN_g, i.e. $([x, y, t] \cdot W_g)$ becomes $NN_g(x, y, t)$. This allows for more general and smooth decomposition of our compute domain.

4 Results

All neural networks were trained on the Taurus HPC system of the Technical University of Dresden. Each node consists of two IBM Power9 CPUs and is equipped with six Nvidia Tesla V-100 GPUs. We parallelized the training of the neural networks using Horovod [10] running on MPI communication backend. Training of the Physics-informed neural network, i.e. solving our QHO, was done on batches consisting of 8.500 points of the initial condition (i.e. $|T_0|$), 2.500 points for the boundary condition (i.e. $|T_b|$) and 2 million residual points (i.e. $|T_f|$).

4.1 Approximation Quality

Training of physics-informed neural networks can be seen as solving partial differential equations in terms of a variational method. State-of-the-art solvers for our benchmarking case, the quantum harmonic oscillator, make us of domain knowledge about the equation by solving in Fourier domain or using Hermite polynomials. We will be comparing both, state-of-the-art spectral method [11] as well as physics-informed neural networks, to the analytic solution of our QHO. This enables a fair comparison of both methods and allows us to quantify the approximation error.

For reasons of comparison, we use neural networks with similar capacity. The baseline model consists of 700 neurons at 8 hidden layer. The GatedPINN with linear and nonlinear gating consists of $N = 10$ experts while the input is processed by one expert ($k = 1$). The experts of the GatedPINN are small MLP with 300 neurons at 5 hidden layers. Furthermore, the gating network for

the nonlinear gating is also a MLP. It consists of a single hidden layer with 20 neurons and the ReLu activation function.

The approximation error is quantified in terms of the infinity norm:

$$err_\infty = ||\widehat{\psi} - \psi||_\infty \ , \tag{3}$$

which allow us to judge the maximum error while not being prone to sparseness in the solution. The relative norm is used for quantifying the satisfaction of the boundary conditions. The relative norm is defined with the approximated surface integral and the sampling points from dataset T_b as follows

$$err_{rel} = ||1 - \iint_{T_b} \psi \ dxdy|| \cdot 100\% \ . \tag{4}$$

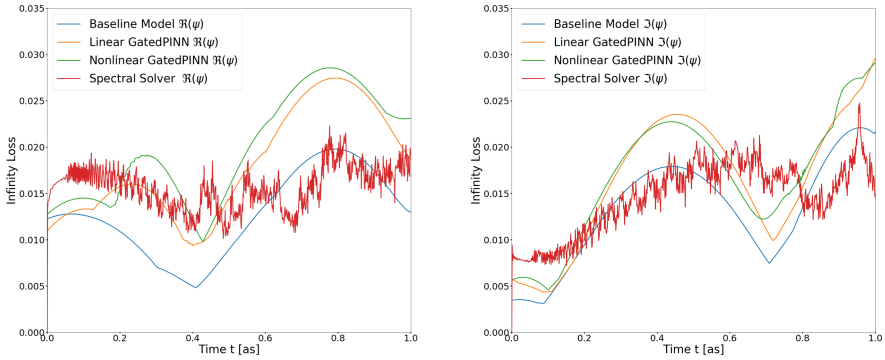
Table 1. Real part statistics of the infinity norm

Approach	err_∞	Min	Max
Spectral solver	$\mathbf{0.01562} \pm 0.0023$	5.3455e−7	0.0223
PINN	0.0159 ± 0.0060	0.0074	0.0265
Linear GatedPINN	0.0180 ± 0.0058	0.0094	0.0275
Nonlinear GatedPINN	0.0197 ± 0.0057	0.0098	0.0286

Table 2. Imaginary part statistics of the infinity norm

Approach	err_∞	Min	Max
Spectral solver	0.01456 ± 0.0038	0.0000	0.0247
PINN	$\mathbf{0.0144} \pm 0.0064$	0.0034	0.0269
Linear GatedPINN	$0.0164 \ \pm 0.0069$	0.0043	0.0296
Nonlinear GatedPINN	0.0167 ± 0.0066	0.0046	0.0291

Physics-informed neural networks as well as GatedPINN are competitive in quality to the spectral solver for the quantum harmonic oscillator in the chosen computational domain as can be seen in Fig. 3. The periodic development in the infinity norm relates to the rotation of the harmonic oscillator which manifests in the real as well as imaginary at different points in time (see Fig. 1).



(a) Real Part (b) Imaginary Part

Fig. 3. Quality of the real part and imaginary part predictions over time in comparison to the spectral solver in reference to the analytically solution

Figure 4 and Fig. 5 show the time evolution of the PINN predictions. The prediction of the baseline model and the GatedPINN models show the same temporal evolution as in Fig. 1.

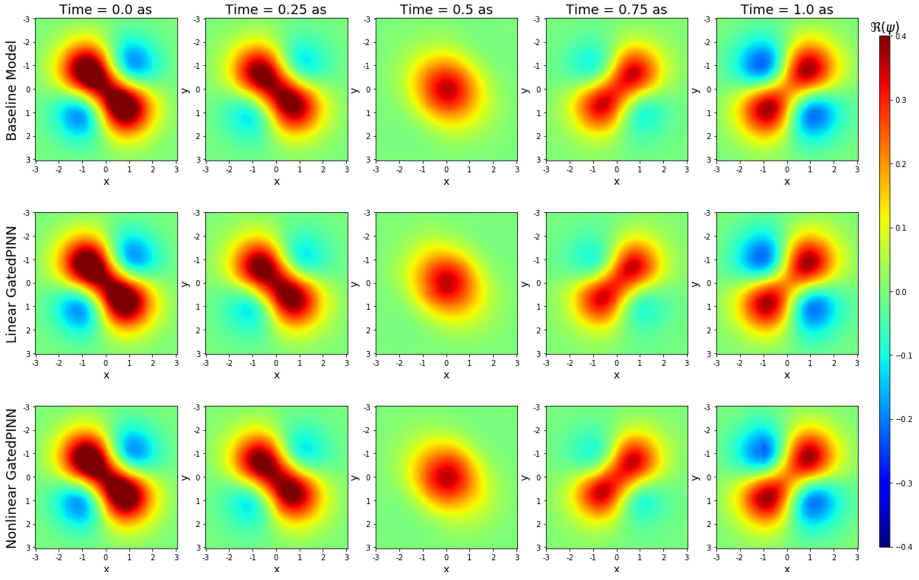


Fig. 4. Real Part predictions of the Baseline and the GatedPINN models

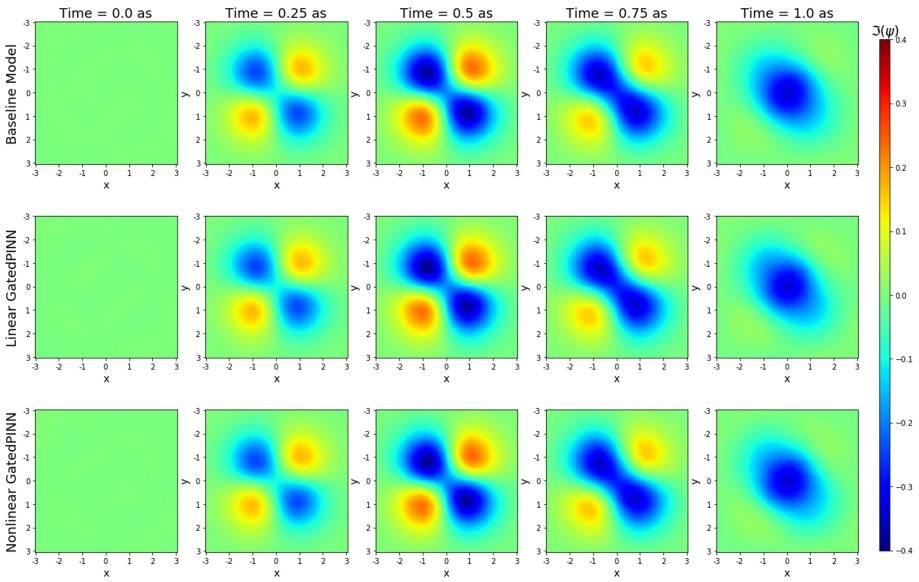


Fig. 5. Imaginary Part predictions of the Baseline and the GatedPINN models

4.2 Domain Decomposition

Table 3. Training time of physics-informed neural networks is significantly reduced by incorporating a domain decomposition into the PINN framework.

Model	Parameters	$\mathcal{L}$	Training time
PINN	**3,438,402**	2.51e−4	29 h 19 min
Linear GatedPINN	3,627,050	**2.115e−4**	**17 h 42 min**
Nonlinear GatedPINN	3,627,290	2.270e−4	18 h 08 min

Table 3 shows the convergence of the PINN-Loss of the baseline, the GatedPINN with linear and nonlinear gating. The Baseline model and the GatedPINN models are trained with 2 million residual points and with the same training setup in terms of batch size, learning rate. Both, the GatedPINN with linear and nonlinear gating have converged to a slightly lower PINN-Loss as the baseline model. However, the training times of the Gated PINN are significantly shorter although the GatedPINN models have more parameters than the baseline model. These results show the efficient usage of the model capacity and automatic differentiation of the GatedPINN architecture. However, both the training time of the PINN and the GatedPINN approach is not competitive to the solution time of the spectral solver (1 min 15 sec). The full potential of PINN can only be

used when they learn the complex relationship between the input, the simulation parameters and the solution of the underlying PDE and thus restarts of the simulation can be avoided.

In Table 1 and 2 we see that the approximation quality of the baseline model is slightly better than the GatedPINN models although the GatedPINN models have converged to a slightly smaller loss $\mathcal{L}$. However, the GatedPINN (linear: 0.329%, nonlinear: 0.268%) satisfies the boundary condition better than the baseline model (1.007%). This result could be tackled by introducing another weighting constant similarly to α to Eq. 2.

The learned domain decomposition of the proposed GatedPINN can be seen in Fig. 6. The nonlinear gating, which is more computationally intensive, shows an more adaptive domain decomposition over time than the model with linear gating. The linear gating converges to a fair distribution over the experts. The nonlinear approach converges to a state where the experts are symmetrically distributed in the initial state. This distribution is not conserved in the time evolution.

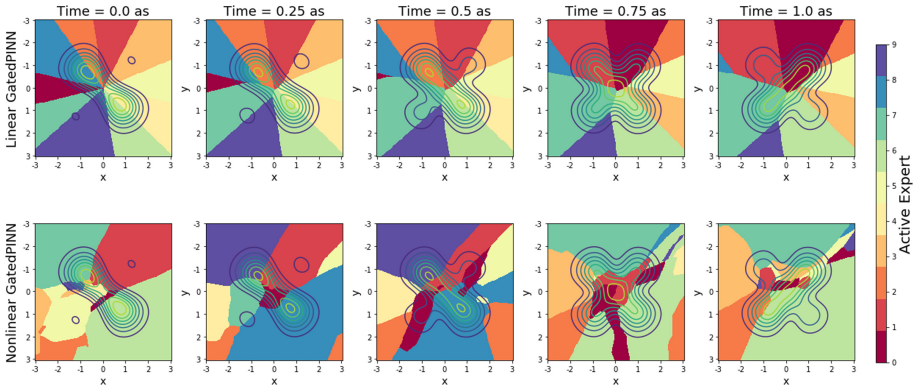


Fig. 6. Learned domain decomposition by the GatedPINN with linear and non-linear gating. The squared norm of the solution ψ is visualized as a contour plot

4.3 Scalability and Power Draw

Training of neural solvers basically relies on unsupervised learning by validating the predicted solution ψ on any residual point (Eq. 2). This means that we only need to compute residual points but do not have to share any solution data. We utilize the distributed deep learning framework Horovod [10]. The scalability analysis was done during the first 100 epochs on using 240 batches consisting of 35000 residual points each and 20 epochs for pretraining. The baseline network is a 8-layer MLP with 200 neurons per layer. Performance measurements were done by forking one benchmark process per compute node.

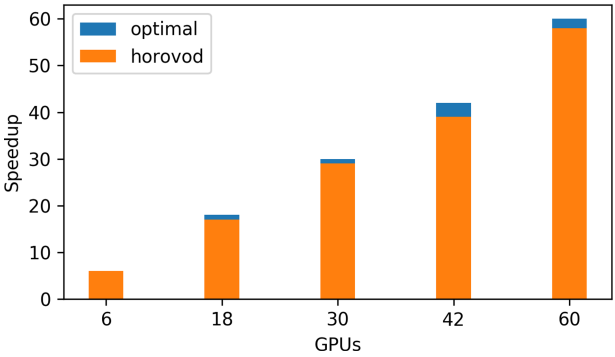


Fig. 7. Speedup comparison

Figure 7 compares the optimal with the actual speedup. The speedup $S(k)$ for k-GPUs was computed by

$$S(k) = t_k/t_1 \ ,$$

provided the runtime for 100 epochs of a single GPU t_1 compared to the runtime of k GPUs: t_k. We found almost linear speedup, though the difference to the optimum is probably due to the latency of the communication between the GPUs and the distribution of residual points and gradient updates. The training achieved an average GPU utilization of 95% ± 0.69% almost fully utilizing each GPU. Memory utilization stays relatively low at an average of 65% ± 0.48% while most of the utilization relates to duplicates of the computational graph due to automatic differentiation.

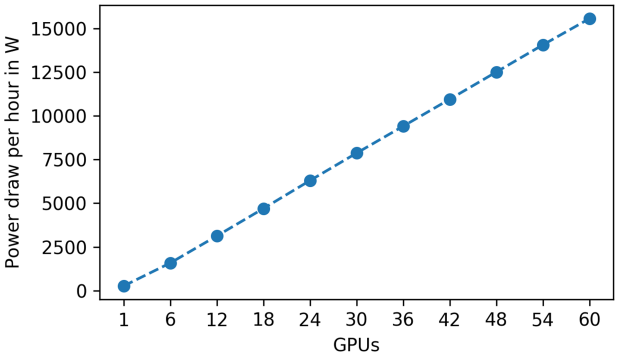


Fig. 8. Power draw comparison

We also quantified the power draw relating to the training in terms of the average hourly draw of all GPUs (See Fig. 8). Note that this rough measure omits

the resting-state power draw of each compute node. We found an almost linear increase in power draw when increasing the number of GPUs. This correlates with the already mentioned very high GPU utilization as well as speedup. These findings imply that total energy for training our network for 100 epochs stays the same - no matter how many GPUs we use. Summarizing, Horovod has proven to be an excellent choice for the distributed training of physics-informed neural networks since training is compute bound. Note that the linear scalability has an upper bound caused by the time needed to perform the ring-allreduce and the splitting of the data.

4.4 Discussion

The experimental results of this paper agree with theoretical results on convergence of PINNs for parabolic and elliptic partial differential equations [6] even for large two-dimensional problems such as the quantum harmonic oscillator. This benchmark dataset[1] provides all means for a comprehensive assessment of approximation error as well as scalability due to the availability of an analytic solution while the smoothness of the solution can be altered by frequency ω of the QHO. The approximated solution of Physics-informed neural networks approached the quality of state-of-the-art spectral solvers for the QHO [11]. The training time of PINN or GatedPINN is not competitive to the runtime of spectral solvers for *one* 2D simulation. However, PINN enable warm-starting simulations by transfer learning techniques, integrating parameters (e.g. ω in our case) or Physics-informed solutions to inverse problems [12] making that approach more flexible than traditional solvers. The former two approaches might tackle that challenge by learning complex relationships among parameters [13] or adapting a simulation to a new configuration at faster training time than learning it from scratch while the latter might pave the way for future experimental usage. The GatedPINN architecture finally allows us to approach higher dimensional data when training physics-informed neural networks by training k sub-PINN each representing a certain fraction of the computational domain at $1/k$ of the total PINN capacity. GatedPINN preserve the accuracy of PINN while the training time was reduced by 40% (Table 3). This effect will become even more evident for 3D or higher dimensional problems. Limiting the computational blowup of PINN and retaining linear speedup (see Fig. 7) are crucial steps towards the applications of physics-informed neural networks on e.g. three-dimensional or complex and coupled partial differential equations.

5 Conclusion

Physics-informed neural networks denote a recent general purpose vehicle for machine learning assisted solving of partial differential equations. These neural

[1] The PyTorch implementations of the benchmarking dataset as well as the neural solvers for 1D and 2D Schrodinger equation and pretrained models are available online: https://github.com/ComputationalRadiationPhysics/NeuralSolvers.

solvers are solely trained on initial conditions while the time-dependent solution is recovered by solving an optimization problem. However, a major bottleneck of neural solvers is the high demand in capacity for representing the solution which relates to the size, dimension and complexity of the compute domain. In this work, we approach that issue by learning a domain decomposition and utilizing multiple tiny neural networks. GatedPINNs basically reduce the number of parameters per unit volume of our compute domain which reduces the training time while almost retaining the accuracy of the baseline neural solver. We find these results on a novel benchmark based on the 2D quantum harmonic oscillator. Additionally, GatedPINN estimate high-quality solutions of the physical system while the speedup is almost linear even for a large amount of GPUs.

Acknowledgement. This work was partially funded by the Center of Advanced Systems Understanding (CASUS) which is financed by Germany's Federal Ministry of Education and Research (BMBF) and by the Saxon Ministry for Science, Culture and Tourism (SMWK) with tax funds on the basis of the budget approved by the Saxon State Parliament. The authors gratefully acknowledge the GWK support for funding this project by providing computing time through the Center for Information Services and HPC (ZIH) at TU Dresden on the HPC-DA.

References

1. Vallecorsa, S.: Generative models for fast simulation. J. Phys. Conf. Ser. **1085**(2), 022005 (2018)
2. Raissi, M.: Deep hidden physics models: deep learning of nonlinear partial differential equations. J. Mach. Learn. Res. **19**, 1–24 (2018)
3. Raissi, M., Perdikaris, P., Karniadakis, G.E.: Physics Informed Deep Learning (Part I): Data-driven Solutions of Nonlinear Partial Differential Equations (Part I), pp. 1–22 (2017)
4. Kim, B., Azevedo, V.C., Thuerey, N., Kim, T., Gross, M., Solenthaler, B.: Deep fluids a generative network for parameterized fluid simulations. Comput. Graph. Forum (Proc. Eurograph.) **38**(2), 59–70 (2019)
5. Tompson, J., Schlachter, K., Sprechmann, P., Perlin, K.: Accelerating Eulerian fluid simulation with convolutional networks. In: Proceedings of the 34th International Conference on Machine Learning (2017)
6. Shin, Y., Darbon, J., Karniadakis, G.E.: On the Convergence and generalization of Physics Informed Neural Networks, vol. 02912, pp. 1–29 (2020)
7. Meng, X., Li, Z., Zhang, D., Karniadakis, G.E.: PPINN: Parareal physics-informed neural network for time-dependent PDEs (2019)
8. Shazeer, N., et al.: The sparsely-gated mixture-of-experts layer, Outrageously large neural networks (2017)
9. Bengio, E., Bacon, P.L., Pineau, J., Precup, D.: Conditional computation in neural networks for faster models (2015)
10. Sergeev, A., Del Balso, M.: Horovod: fast and easy distributed deep learning in TensorFlow (2018)
11. Feit, M.D., Fleck Jr., J.A., Steiger, A.: Solution of the schrödinger equation by a spectral method. J. Comput. Phys. **47**(3), 412–433 (1982)

12. Chen, Y., Lu, L., Karniadakis, G.E., Negro, L.D.: Physics-informed neural networks for inverse problems in nano-optics and metamaterials. Opt. Exp. **28**(8), 11618 (2020)
13. Michoski, C., Milosavljevic, M., Oliver, T., Hatch, D.: Solving irregular and data-enriched differential equations using deep neural networks. CoRR **78712**, 1–22 (2019)