Harmonic Neural Networks

Atiyo Ghosh [1]   **Antonio A. Gentile** [1]   **Mario Dagrada** [1]   **Chul Lee** [2]   **Seong-Hyok Kim** [2]   **Hyukgeun Cha** [2]
Yunjun Choi [2]   **Dongho Kim** [2 3]   **Jeong-Il Kye** [2 4]   **Vincent Elfving** [1]

Abstract

Harmonic functions are abundant in nature, appearing in limiting cases of Maxwell's, Navier-Stokes equations, the heat and the wave equation. Consequently, there are many applications of harmonic functions from industrial process optimisation to robotic path planning and the calculation of first exit times of random walks. Despite their ubiquity and relevance, there have been few attempts to incorporate inductive biases towards harmonic functions in machine learning contexts. In this work, we demonstrate effective means of representing harmonic functions in neural networks and extend such results also to quantum neural networks to demonstrate the generality of our approach. We benchmark our approaches against (quantum) physics-informed neural networks, where we show favourable performance.

1. Introduction

Harmonic functions can be defined as the solutions to Laplace's equation $\nabla^2 \phi = 0$, though there are alternative mathematically-equivalent definitions. Such functions are ubiquitous in nature, appearing in limiting cases of Maxwell's equations (Griffiths, 2005), the heat (or diffusion) equation, the wave equation, irrotational flow in fluid dynamics (Anderson, 2011) and first-hitting times of random walks (Redner, 2001). Consequently, harmonic functions are relevant in many practical settings, including navigation in robots (Prestes e Silva et al., 2002), electrostatic imaging (Akduman & Kress, 2002) and heat transport (Sharma et al., 2018), to name but a few examples.

[1]PASQAL SAS, 2 av. Augustin Fresnel, Palaiseau, 91220, France [2]LG Electronics, AI Lab, CTO Div, 19, Yangjae-daero 11-gil, Seocho-gu, Seoul, 06772, Republic of Korea [3]POSCO Holdings, AI R&D Laboratories, 440, Tehera-ro, Gangam-gu, Seoul, 06194, Republic of Korea [4]Pinostory Inc., 1-905 IT Castle, Seoul, 08506, Republic of Korea. Correspondence to: Atiyo Ghosh <atiyo.ghosh@pasqal.com>.

Finite element methods (FEM) often represent the state-of-art approach in solving the corresponding equations, with attempts specifically targeting harmonic functions (Galybin & Irša, 2010). In contrast to FEM, machine learning based approaches offer the chance to assist with data-driven modelling (analogous to (Raissi et al., 2019)), handle noisy-data/boundary conditions, solve inverse problems for design optimization (Lu et al., 2021), perform transfer learning and model discovery (Both et al., 2021). Effective learnable and differentiable representations of harmonic functions could therefore be key in multiple fields.

Physical Inductive Biases in Conventional Machine Learning   Of particular relevance to our work on harmonic functions are physics-informed neural networks (PINNs) (Raissi et al., 2019), which use derivatives of neural networks (NNs) with respect to their inputs as a regularisation term to provide an inductive bias towards a given differential equation, including potentially towards harmonic functions. Even though this concept was introduced decades ago (Lagaris et al., 1998), PINNs have only recently garnered much attention. For a wider overview of PINNs, we direct the interested reader to a recent review (Karniadakis et al., 2021).

Related to PINNs, there has been a growing recent interest in applying machine learning towards differential equation problems, such as the solution of families of differential equations (Li et al., 2020; Lu et al., 2019), or facilitating the solution via dimensionality reduction (Gunzburger et al., 2007). Finding NNs which directly obey a given (linear) differential operator as a hard constraint has attracted some recent attention (Hendriks et al., 2020). Divergence-free vector fields were also used recently to represent continuity equations (Richter-Powell et al., 2022). We also leverage divergence-free fields in parts of our exposition. In addition to physical inductive biases encodable via differentiable equations, there have also been recently developed methods to effectively impose energy conservation on learnt representations (Greydanus et al., 2019; Cranmer et al., 2020).

Physical Inductive Biases in Quantum Machine Learning   A promising avenue for achieving practically-relevant quantum advantage within currently available and near-term noisy intermediate-scale quantum computers consists of

variational quantum algorithms (Cerezo et al., 2021). We introduce this perspective in further detail in Appendix A.2 and refer interested readers to a recent introduction to quantum machine learning (Schuld & Petruccione, 2021) for further background. There are several parallels between variational quantum models and NNs. For example, both can be seen as parameterised functions which are optimised with a conventional optimiser, achieving universal approximation properties (Kurt et al., 1989; Goto et al., 2021). Furthermore, there are means of achieving automatic differentiation directly on quantum circuits (Guerreschi & Smelyanskiy, 2017; Kyriienko & Elfving, 2021), making them effectively a possible generalization of classical NN architectures. It has been speculated that including inductive biases within quantum circuit designs might prove beneficial (Hadfield et al., 2019; Bharti & Haug, 2021; Kübler et al., 2021), possibly also to address difficulties that hinder the achievement of practical quantum advantage (Huang et al., 2021).

Differentiable Quantum Circuits (DQC), outlined in (Kyriienko et al., 2021), adopt an advantageous latent space representation for the mapping of input data, and derivatives of quantum circuits with respect to their inputs to solve given (also non-linear) differential equations. Equipped with appropriate loss functions taking into account soft constraints, as well as automatic differentiation rules providing an effective surrogate of backpropagation rules in conventional NNs, such architectures can then be regarded as a quantum PINN, or for simplicity qPINNs (we emphasise though that this terminology is *not* widely used). Training qPINNs poses in principle no fewer issues than PINNs of equivalent expressivity, and this limitation might be exacerbated by the sheer number of parameters that can potentially describe vast quantum circuits, which can complicate training (McClean et al., 2018; Anschuetz & Kiani, 2022). This makes it an ideal test-bed to investigate the application of relevant inductive biases in quantum circuits. However, the use of inductive biases in designing quantum circuits and quantum machine learning models is still a nascent field, that we review briefly in general in Appendix A.2.

Focus of the Paper   Our main focus in this work is to incorporate inductive biases for harmonic functions in machine learning models. We demonstrate exactly-harmonic functions for simply-connected two-dimensional domains using complex-valued NNs. For multiply-connected domains, we provide a domain decomposition methodology which allows us to continue using complex-valued, conventional NNs as harmonic function representations. We investigate the general applicability of the methods we develop, by demonstrating how they can be generalised to the quantum architectures introduced above. Finally, we use example problem settings from heat distribution, electrostatics and robot navigation to demonstrate the effectiveness

of our proposed inductive biases in both conventional and (simulated) quantum NNs. We provide implementations of our classical architectures in the supplementary material (SI) to facilitate the adoption of these inductive biases.

Note on Terminology   To avoid confusion, henceforth we will use the term 'neural network' to refer to statements applying to both quantum circuits and conventional neural networks. When seeking to refer specifically to quantum circuits or conventional NNs, we will explicitly state so.

2. Theory

We start by considering exact harmonicity in two dimensions in section 2.1.1 and 2.1.2 using techniques leveraging the behaviour of complex functions. However, such constructions do not extend to arbitrary dimensions. Consequently, we explore higher dimensions in section 2.2.

2.1. Exact Harmonicity in Two Dimensions

Consider (quantum and classical) neural networks of $\phi_H : \mathbb{R}^2 \to \mathbb{R}$ of the form

$$\phi_H(x,y) = \Re \circ \mathrm{NN}_h \circ c. \tag{1}$$

where $c(x,y) = x + iy$ and $\Re(x + iy) = x$.

Theorem 2.1. *Architectures of the form in equation* (1) *are harmonic for holomorphic (i.e. complex differentiable) choices of NN_h.*

Proof. Since NN_h is holomorphic, it satisfies the Cauchy-Riemann equations. Writing $\mathrm{NN}_h(x + iy) = u(x,y) + v(x,y)$, the Cauchy-Riemann equations then read

$$\frac{\partial u}{\partial x} = \frac{\partial v}{\partial y} \quad \text{and} \quad \frac{\partial u}{\partial y} = -\frac{\partial v}{\partial x} \tag{2}$$

Taking partial derivatives of the first equation of (2) with respect to x and the second equation with respect to y and summing both equations yields $\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = 0$. $\qquad\square$

Theorem 2.2. *There exist polynomial choices of NN_h allow equation* (1) *to converge uniformly to any harmonic function on the interior of any simply-connected compact proper subdomain of $\mathbb{C}$.*

Proof. Any simply-connected proper (open) subdomain $\mathbb{C}$ is homeomorphic to the (open) unit disc by the Riemann mapping theorem (Bak et al., 2010). This demonstrates the complement of any simply-connected domain is connected in $\mathbb{C}$. Thus, Mergelyan's theorem applies, which guarantees that any continuous function which is holomorphic on the interior of the domain can be approximated uniformly by polynomials (Gaier, 1987). $\qquad\square$

Remark 2.3. Even though (learnable) polynomials can provide a basis for universality arguments, representing arbitrary degree polynomials might be memory intensive for practical applications. Other holomorphic constructions for NN_h might thus provide better practical performance.

However, architectures of the form in (1) are not universal for holomorphic NN_h in the space of arbitrary compact domains in $\mathbb{C}$.

Proposition 2.4. *Architectures of the form in equation* (1) *cannot represent arbitrary harmonic functions on multiply-connected domains for holomorphic NN_h.*

Proof. As a counterexample, the function $u(x, y) = \log(x^2 + y^2)$ is harmonic on $\mathbb{C}\backslash 0$, but is not the real part of any holomorphic function there (Bak et al., 2010). As NN_h equation (1) is restricted to be holomorphic, it cannot represent this function. $\square$

Consequently, in this section, we provide separate techniques for modelling harmonic functions in simply-connected (section 2.1.1) domains and multiply-connected domains (section 2.1.2).

2.1.1. EXACT HARMONICITY: HOLOMORPHIC NETWORKS ON SIMPLY-CONNECTED DOMAINS

Conventional Neural Networks Construct a complex-valued neural network $NN_h : \mathbb{C} \to \mathbb{C}$ by taking a multi-layer perceptron with complex-valued weights in linear layers, and holomorphic activation functions, for example $\sigma(z) = \sin(z)$ or $\sigma(z) = \exp(z)$.

Using such a network in equation (1) is then guaranteed to be harmonic by Theorem 2.1.

There has been much research into complex-valued conventional NNs, but we consider a meaningful review of them to be outside the scope of our work, referring to (Bassey et al., 2021) for an introduction to their theory and application.

Quantum Neural Networks For the interested reader, we derive implementations of equation (1) on quantum circuits in the Appendix.A.1.

2.1.2. EXACT HARMONICITY: MULTIHOLOMORPHIC NETWORKS ON MULTIPLY-CONNECTED DOMAINS

For many problems of interest, harmonic functions on multiply-connected domains need to be constructed. As demonstrated in proposition 2.4, we require architectures beyond those demonstrated in equation (1) in these scenarios. We develop such architectures in this section.

We consider a multiply-connected domain in $\mathbb{R}^2$ which we denote by Ω and disjointly decompose it such that $\Omega = (\dot{\bigcup}_i \omega_i) \dot{\cup} (\dot{\bigcup}_{i,j} \partial\omega_{i,j})$ with $i, j \in \mathbb{N}$, each ω_i being simply-connected and each $\partial\omega_{ij}$ a region of (Lesbegue) measure zero between the subdomains ω_i and ω_j.

For each ω_i, we associate a (quantum) harmonic neural network, ϕ_H, of the form outlined in Eq. 1. We can then construct a representation for a harmonic neural network $\phi_{\mathrm{MH}} : \Omega \to \mathbb{R}$ as follows:

$$\phi_{\mathrm{MH}}(\boldsymbol{x}) = \begin{cases} \phi_H^{(i)}(\boldsymbol{x}) & \boldsymbol{x} \in \omega_i \\ \frac{\phi_H^{(i)}(\boldsymbol{x}) + \phi_H^{(j)}(\boldsymbol{x})}{2} & \boldsymbol{x} \in \partial\omega_{ij}. \end{cases} \tag{3}$$

Proposition 2.5. *Architectures of the form in* (3) *are exactly harmonic almost everywhere in Ω. Furthermore, they uniformly converge to any harmonic function almost anywhere if each $\phi_H^{(i)}$ converges uniformly to any harmonic function on each ω_i.*

Proof. The only regions where $\phi_{MH}(x)$ might not be harmonic are in $\partial\omega_{ij}$, which have Lebesgue measure zero, hence allowing for harmonicity almost everywhere. Similarly, the universality of each $\phi_H^{(i)}$ on each ω_i means the universality of ϕ_{MH} can only be violated on each $\partial\omega_{ij}$, allowing for uniform convergence almost everywhere. $\square$

While ϕ_{MH} is harmonic by construction on each ω_i, it remains to make it harmonic across each $\partial\omega_{ij}$. We achieve this by including an extra loss term $L_{\partial\omega}$ alongside any other loss that ϕ_{MH} is being optimised upon:

$$L_{\partial\omega} = \sum_{i,j} \mathbb{E}_{\boldsymbol{x} \sim \partial\omega_{ij}} \left[\left(\phi_H^{(i)}(\boldsymbol{x}) - \phi_H^{(j)}(\mathbf{x}) \right)^2 + \left\| \nabla\phi_H^{(i)}(\boldsymbol{x}) - \nabla\phi_H^{(j)}(\boldsymbol{x}) \right\|_2^2 \right], \tag{4}$$

where the first term in the sum incentivises continuity in the harmonic function, and the second term represents a squared L_2 norm which incentivises continuity in the conservative field of the harmonic function. Such stitching together of domains, with loss functions ensuring continuity of relevant dynamics, is closely related to some domain decomposition techniques in PINNs (Jagtap & Karniadakis, 2020).

The fundamental result of such a decomposition is that ϕ_{MH} can be harmonic almost surely, but it need not correspond to any holomorphic function on each $\partial\omega_{ij}$. Since this construction is only non-harmonic on boundaries, we consider it to be harmonic almost everywhere. We consider the practical performance of this in Sect.3.5.

2.2. Approximate Harmonicity: Curl-Driven Harmonic Networks

While the previous sections demonstrate exact harmonicity in two dimensions, they are not applicable in higher dimensions. In this section, we present techniques applicable to arbitrary dimensions. Note that Laplace's equation can be written as $\nabla \cdot (\nabla \phi) = 0$. This demonstrates that the gradient of any harmonic function must be divergence-free. Furthermore, in three dimensions, a general divergence-free field can be obtained by the identity $\nabla \cdot (\nabla \times A) = 0$. This motivates the construction of an inductive bias comprising of two networks: $\phi_C(\boldsymbol{x}; \boldsymbol{\theta}_\phi)$ representing a harmonic potential field and $A(\boldsymbol{x}, \boldsymbol{\theta}_A)$ representing a neural network whose curl will be taken to provide a representation of the underlying conservative field of ϕ_C, where $\boldsymbol{\theta}_\phi$ and $\boldsymbol{\theta}_A$ represent trainable parameters. Including the following loss term during network optimisation thus provides an inductive bias towards harmonic functions:

$$L_c(\boldsymbol{\theta}_\phi, \boldsymbol{\theta}_A) = \mathbb{E}\left[\|\nabla \phi_C - \nabla \times A\|_2^2\right] \qquad (5)$$

where the expectation is taken with respect to a probability distribution whose support covers the domain over which the inductive bias is desired. During optimisation, both ϕ_C and A can be trained to minimise (5), there is no minimax game as in generative adversarial networks that might destabilise training in such a procedure.

Even though the curl operator is only well-defined in three dimensions, we note that we can use the exterior calculus of differential forms to derive similarly divergence-free operators in arbitrary dimensions.

Proposition 2.6. *Given an $(N-2)$ form in $\mathbb{R}^N$ whose components in a Cartesian basis are given by a neural network $A : \mathbb{R}^d \to \mathbb{R}^N$, $d = \binom{N}{N-2}$, the exterior derivative dA represents a divergence-free neural network.*

Proof. The exterior derivative of an $(N-2)$ form A, yields an exact $(N-1)$ form, dA. Since there is a correspondence between the divergence of a field and the exterior derivative of an $(N-1)$ form, and since $d(d\phi) = 0$ for any differential form ϕ, the result follows. $\qquad \square$

While a full discussion of differential forms is outside the scope of this work, we provide example calculations in Appendix B to facilitate incorporating such architectures in wider work. We also note that similar arguments have been recently presented in (Richter-Powell et al., 2022).

In two dimensions, this leads (with some abuse of notation in reusing the curl symbol to denote exterior derivatives) to:

$$\nabla \times A = \left(\frac{\partial y_1}{\partial x_2}, -\frac{\partial y_1}{\partial x_1}\right), \qquad (6)$$

with $A : \mathbb{R}^2 \to \mathbb{R}$ defined by $A(x_1, x_2) = y_1$. A sample four-dimensional calculation is demonstrated in Appendix B.2.

We note that such operators are amenable to implementation on quantum circuits. Two independent quantum circuits representing ϕ_C and A appearing in Eq. 5 (as well as determining additional loss contributions detailed in Sect. 3.1) can be estimated via simple cost functions (e.g. the *total magnetization*). Composing such contributions into the relevant loss term(s) can be attained with additional classical automatic differentiation routines, e.g. (Bergholm et al., 2018). The latter ones can also handle circuit derivatives (here necessary up to the 2_{nd} order), which can be alternatively attained with analytical methods on quantum hardware, such as the parameter-shift rule (Guerreschi & Smelyanskiy, 2017; Mari et al., 2021).

3. Applications

We exemplify our approaches on sample applications geared towards electrostatics (Sect.3.3), heat distribution (Sects.3.4&3.5), and robot navigation (Sect.3.6) and a further 3D fluid flow example in Sect.3.7. In order to provide robust benchmarks, we benchmark several forward solutions against finite element methods (FEM). Since the variational form of the FEM solution is linear, we assume that it converges reliably to provide a strong baseline. In addition to the methods introduced in this work, we also include comparisons to PINNs (Raissi et al., 2019), as well as PINNs with Dirichlet-condition constrained architectures (Lu et al., 2021). We demonstrate a proof of concept of a quantum holomorphic network in Sect.3.4. A multiply-connected domain is explicitly constructed in Sect.3.5 to demonstrate the Multiholomorphic networks of Sect.2.1.2. Multiholomorphic networks are not tested in other settings since their spatial domains do not require a simply-connected decomposition. No holomorphic networks are tested in Sect.3.7 since holomorphic networks do not apply to the 3D domain of that benchmark.

While we present results for both classical and quantum circuits, we emphasise that we do not wish to compare classical and quantum neural networks. The comparison is fraught with difficulties. For example, there is no clear way to compare a number of qubits with multilayer perceptron size.

Full finite elements and classical NN code can be found in the Supplementary Material (SI). We continue by outlining some loss functions we use with different methods, as well as summarising the implementation of each method.

3.1. Methods

3.1.1. LOSS FUNCTIONS

Dirichlet Losses   Imposing Dirichlet boundary conditions can be done variationally, using mean squared error losses. Given a Dirichlet boundary Γ, where the target function is known to take constant value $c \in \mathbb{R}$, we train an arbitrary (quantum or conventional neural network) $\phi(\boldsymbol{x}, \boldsymbol{\theta})$ to obey such a condition by including a term:

$$L_d = \mathbb{E}_{\boldsymbol{x} \sim \Gamma} \left[(\phi(\boldsymbol{x}, \boldsymbol{\theta}) - c)^2 \right] \tag{7}$$

Physics-Informed Harmonic Losses   Given a spatial-domain Ω and a (quantum) neural network $\phi(\boldsymbol{x}, \boldsymbol{\theta})$, we follow established methodologies (Raissi et al., 2019) to define a physics-informed loss towards harmonic functions by zeroing the Laplacian

$$L_{\mathrm{PI}_H} = E_{\boldsymbol{x} \sim \Omega} \left[\left(\nabla^2 \phi(\boldsymbol{x}, \boldsymbol{\theta}) \right)^2 \right] \tag{8}$$

Dielectric Interface Loss   To model the behaviour of a potential field across a dielectric interface, we must impose Maxwell's interface conditions, namely that potential fields (i.e. harmonic functions) are continuous across the interface, and that the normal components of underlying electric fields (i.e. the gradient of the harmonic function) are scaled according to material permittivities (Jackson, 1999). Consequently, given two domains Ω_1, Ω_2 with permittivities of ϵ_1 and ϵ_2 and (quantum) neural networks $\phi_1(\boldsymbol{x}; \boldsymbol{\theta}_1)$ and $\phi_2(\boldsymbol{x}; \boldsymbol{\theta}_2)$ defined on each domain respectively, we construct a normal unit vector $\boldsymbol{n}(\boldsymbol{x})$ to the boundary $\partial\Omega$ between Ω_1 and Ω_2 and define the following loss term:

$$
\begin{aligned}
L_{\mathrm{PI}_{\mathrm{DE}}}(\boldsymbol{\theta}_1, \boldsymbol{\theta}_2) = \quad & E_{\boldsymbol{x} \sim \partial\Omega} \Big[(\phi_1(\boldsymbol{x}; \boldsymbol{\theta}_1) - \phi_2(\boldsymbol{x}; \boldsymbol{\theta}_2))^2 \\
& + \left(\epsilon_1 \frac{\partial E_1}{\partial \boldsymbol{n}} - \epsilon_2 \frac{\partial E_2}{\partial \boldsymbol{n}} \right)^2 \Big]
\end{aligned}
\tag{9}
$$

where E_i represents the electric field in domain Ω_i. In general contexts, we have $E_i = \nabla\phi_i(\boldsymbol{x}, \boldsymbol{\theta})$. However, in the case of our curl-based approach, we represent E_i by the underlying divergent free field $\nabla \times A$ as outlined in Sect. 2.2.

3.1.2. NEURAL NETWORKS

(q)PINN (baseline)   We define a single (quantum or conventional) neural network, $\phi_{\mathrm{PINN}}(\boldsymbol{x}; \boldsymbol{\theta})$ and minimise the loss

$$L_{\mathrm{PINN}} = L_{\mathrm{PI}_H} + L_d \tag{10}$$

with L_{PI_H} and L_d defined in Eqs. 7 and 8 respectively.

h(q)PINN (baseline)   Given a multilayer perceptron $\mathrm{MLP}(\boldsymbol{x}; \boldsymbol{\theta})$ and a Dirichlet boundary Γ taking constant value $c \in \mathbb{R}$ on it, we can construct a neural network that exactly satisfies the boundary condition while leaving flexibility to optimise towards harmonic functions (Lu et al., 2021). First, we define a distance function $d(\boldsymbol{x})$ representing the shortest distance between a generic point $x \in \Omega$ and Γ, and then define a neural network of the form:

$$\phi_{\mathrm{hPINN}} = ce^{-kd(\boldsymbol{x})} + (1 - e^{-kd(\boldsymbol{x})})\mathrm{MLP}(\boldsymbol{x}; \boldsymbol{\theta}), \quad \tag{11}$$

where $k \in \mathbb{R}$ is a hyperparameter to be chosen. Since this network satisfies Dirichlet condition automatically, it can be trained by optimising on Eq. 8 directly.

(q)Holomorphic (ours)   We construct a harmonic neural network as defined in Eq. 1 and a harmonic quantum neural network as defined in Eq. 17. Since these networks are harmonic by construction, our task of training them is reduced to a supervised learning problem on the boundary terms/data. Consequently, we train them by minimising Eq. 7 alone.

Curl(q)Net (ours)   We define two neural networks $\phi_c(\boldsymbol{x}, \boldsymbol{\theta}_1)$ and $A(\boldsymbol{y}, \boldsymbol{\theta}_2)$ as outlined in Sect. 2.2. Note that $\nabla\phi_c$ and $\nabla \times A$ must operate in the same space, however in general the two (q)NNs can be defined on different spaces.

The Curl(q)Net solution is achieved by minimising the following objective function:

$$L_{\phi_c} = L_d + L_c \tag{12}$$

with L_d and L_c defined in Eqs. 7 and 5 respectively. Indeed, satisfying L_c amounts to satisfy L_{PI_H}. We substitute Eq. 6 into 5 in our scenario since our applications are in two dimensions. We further note that the term L_d can be dropped from the optimisation procedure if ϕ_c is defined in a manner analogously to Eq. 11.

Multiholomorphic (ours)   In the case of multiply-connected domains, we decompose the domain into simply-connected components so that we can still leverage holomorphic function to derive harmonic neural networks (see also Fig. 2a). Thus, we take a multiholomorphic network ϕ_{MH} as defined in Eq. (3) and minimise the following:

$$L_{MH} = L_{\partial\omega} + L_d \tag{13}$$

with L_d and $L_{\partial\omega}$ defined in Eqs. 7 and 4 respectively.

XPINN (baseline)   Since our multiholomorphic function involves partitioning a domain, we include a domain decomposition strategy applied to PINNs as a further benchmark to try and maintain a fair perspective. The fundamental idea

behind XPINNs is to decompose a domain into disjoint sub-domains (Jagtap & Karniadakis, 2020), whilst variationally inducing the continuity of the solutions across each subdomain. We can define a neural network $\phi_{\text{XPINN}}(\boldsymbol{x}; \boldsymbol{\theta})$ analogously to the multiholomorphic network in Eq. 3, except we use real-valued multilayer perceptrons on each subdomain as opposed to harmonic neural networks. Consequently we can train $\phi_{\text{XPINN}}(\boldsymbol{x}; \boldsymbol{\theta})$ by minimising

$$L_{\text{XPINN}} = L_{\partial\omega} + L_d + L_{\text{PI}_H}, \quad (14)$$

with $L_{\partial\omega}$, L_d and L_{PI_H} defined in equations 4, 7 and 8 respectively, the latter contribution being necessary due to dropping of conditions imposing harmonicity.

3.2. Experimental Setup

We conduct all (both quantum and conventional) experiments in Python3 and make use of NumPy (Harris et al., 2020), PyTorch (Paszke et al., 2019) and Matplotlib (Hunter, 2007) packages. FEM ground truths were constructed using FEniCS (Alnæs et al., 2015). All the QNNs used in this paper are implemented with proprietary code, leveraging upon the packages PyTorch and Yao.jl (Luo et al., 2020).

We implement all classical neural networks in Py-Torch (Paszke et al., 2019). We consistently use multilayer perceptrons with 3 hidden layers, width 32, initialised with Kaiming Uniform initialisers (He et al., 2015), optimised for 16,000 epochs over full-batches with an Adam optimizer (Kingma & Ba, 2014) with a learning rate (LR) of 10^{-3}. We use $\tanh$ activations for real-valued NNs and $\sin$ activations for holomorphic NNs. All the QNNs used in this paper are implemented with proprietary code, leveraging upon the packages PyTorch and Yao.jl (Luo et al., 2020). Further details, including details on simulated quantum circuits and hardware specifics, are in Appendix C. Here, however, we observe that a QNN defined over an n-qubit quantum circuit has an expressivity comparable to a spectral expansion employing 2^n terms. This is empirically observed e.g. in Sect. 3.4. Such an exponential increase in the expressivity with the quantum circuit size holds promise for potential future advantage in expressing targeted solutions.

For each application, we construct boundary conditions comprising of lines where we sample 100 uniformly-spaced points. To minimise physics-informed losses, we sampled 1024 collocation points randomly (uniformly) on the interior of each domain. These points are sampled once and then kept constant for each experiment to allow each run the same amount of information.

3.3. Results A: Dielectric Material in a Charged Box

Consider a dielectric material placed in a 2D box grounded on 3 sides, and an electric potential applied to the bottom edge. We consider the task of inferring the underlying
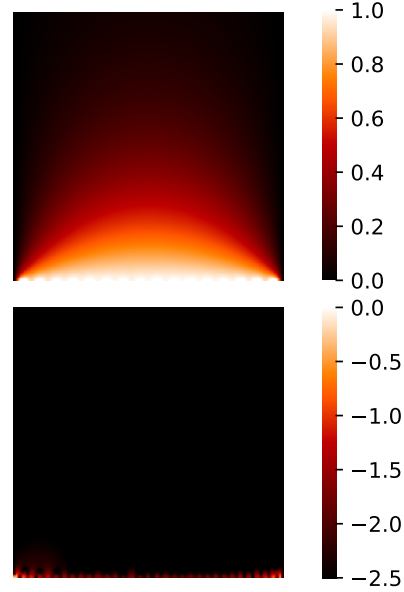

Figure 1. (top) heat profile in arbitrary units for a refrigerated box with the bottom side heated, computed with a numerical simulation of a 4 qubits hQNN. (bottom) $\log_{10}$-scale error of hQNN vs analytical solution.

electric potential field within the box, in the presence of different materials.

We thus introduce a first dielectric domain $\Omega_1 = [0, 1] \times [0, 0.5]$, with permittivity $\epsilon_1 = 1.0$ and free-space above the dielectric as $\Omega_2 = [0, 1] \times [0.5, 1]$ with permittivity $\epsilon_2 = 0.01$. The bottom lid ($y = 0$) has an applied voltage of $V = 1.0$ (in arbitrary units), whereas $V = 0.0$ on all other boundaries. We define neural networks and losses on each Ω_1 and Ω_2 as outlined in Sect. 3.1. In addition, we couple the losses of both networks using Eq. 9 and optimize both networks jointly. We do not consider our multiholomorphic or the XPINN architecture in this scenario, since they were devised for use on multiply-connected domains, as opposed to the simply-connected setting in this application.

We report in Table 1 the results as (i) root mean squared errors (RMSE) against FEM solutions, i.e. $\sum_{\boldsymbol{x}_i \in \tilde{\Omega}} \sqrt{(\phi(\boldsymbol{x}_i) - \phi_{\text{FEM}}(\boldsymbol{x}_i))^2}/|\tilde{\Omega}|$, where $\tilde{\Omega} \equiv \{\boldsymbol{x}_i\}$ a set of collocation points uniformly sampled in the interior domain Ω, along with (ii) the expected absolute Laplacian over Ω.

We note that PINNs achieve a low mean Laplacian over the domain, even though they do not represent the best-performing solutions. One possible explanation would be that the Laplacian is directly optimised for PINNs (equations 8 and 10), but minimising such a loss could still lead to systematic errors across a spatial domain.

6

Table 1. Performance of our methods on the electrostatic, heat-distribution and navigation benchmarks illustrated respectively in Sect. 3.3, 3.5 & 3.6. Note that all numbers are multiplied by 100 for clarity. We present the mean and standard deviation of each method over 10 repeated runs. In addition, we report the expected absolute Laplacian over the spatial domain, which should be identically zero for perfectly harmonic functions. We separate quantum (see also Fig. 2) and conventional techniques with a single horizontal line. Lower numbers indicate better performance. Each best-performing method is highlighted by bold text.

	Electrostatics		Heat Distribution		Robot Navigation		Fluid Flow	
	RMSE	$E\left[\left\|\nabla^2\phi\right\|\right]$	RMSE	$E\left[\left\|\nabla^2\phi\right\|\right]$	RMSE	$E\left[\left\|\nabla^2\phi\right\|\right]$	RMSE	$E\left[\left\|\nabla^2\phi\right\|\right]$
Holomorphic (ours)	**0.3** ±**0.09**	0.0 ±0.0	19.1 ±0.05	0.0 ±0.0	17.1 ±0.5	0.0 ±0.0	-	-
CurlNet (ours)	1.4 ±0.06	165 ±27.3	2.4 ±0.004	0.04 ±0.05	**1.6** ±**1.1**	2013 ±1502	**11.2** ±**0.7**	162 ±8.3
PINN	6.7 ±1.8	5.3 ±0.6	2.4 ±0.0006	0.002 ±0.0003	24.2 ±28.1	11.2 ±1.8	18.1 ±0.6	10.9 ±5.9
hPINN	10.0 ±0.03	11.4 ±1.3	2.4 ±0.006	0.03 ±0.05	-	-	14.6 ±0.08	91.4 ±33.7
Multiholomorphic (ours)	-	-	**0.2** ±**0.09**	0.0 ±0.0	-	-	-	-
XPINN	-	-	2.4 ±0.02	0.1 ±0.3	-	-	-	-
CurlqNet (ours)	**3.4** ±**0.03**	269 ±15.4	**2.1** ±**0.02**	2.8 ±0.4	**6.6** ±**0.09**	307 ±9.4	-	-
qPINN	13.8 ±0.3	8.4 ±0.7	17.2 ±0.3	28.2 ±2.4	33.0 ±0.4	14.3 ±0.6	-	-
hqPINN	16.2 ±1.8	10.1 ±19.7	6.5 ±0.04	2.4 ±0.5	-	-	-	-
qHolomorphic (ours)	-	-	-	-	16.5 ±0.8	0.0 ±0.0	-	-

3.4. Results B: Heat Distribution in a Box with Single-Sided Heating

We consider a refrigerated square box with uniform heating on a single side, where we wish to find the steady-state heat distribution described by ϕ. We consider a domain $\Omega = [0,1] \times [0,1]$. Desired Dirichlet boundary conditions are $\phi = 1$ on the edge $x = 0$, and $\phi = 0$ (in arbitrary units) when $y = 0$, $y = 1$, or $x = 1$. Note that this problem is a single (simply-connected) domain. We know the analytical solution as an infinite series, and plot the converged result when adopting a *qHolomorphic* architecture, along with the absolute error with the analytical result, in Fig. 1. We use the availability of an analytical result for this case to highlight the expressivity of QNN architectures in comparison with classical spectral solutions, as discussed in detail in Appendix A.4).

3.5. Results C: Heat Distribution Around a Heater

We consider a heated triangular body in a refrigerated box, where we wish to find the steady-state heat distribution described by ϕ.

We consider a domain $\Omega = [0,10] \times [0,10]$, with an equilateral triangle of length 4 centred in the box as illustrated in Fig. 2a. Desired Dirichlet boundary conditions are $\phi = 1$ on the triangle edges and $\phi = 0$ (in arbitrary units) when $x = 0$, $x = 10$, $y = 0$ or $y = 10$. With these conditions, ϕ defines a harmonic function representing the steady-state temperature distribution inside the box.

Note that this problem is not on a simply-connected domain. For example, a closed path around the triangular body cannot be reduced to a point, without crossing the body. Consequently, we have reason to believe that the resulting solution will not be representable by our holomorphic formulation, though it could be representable by a multiholomorphic for-
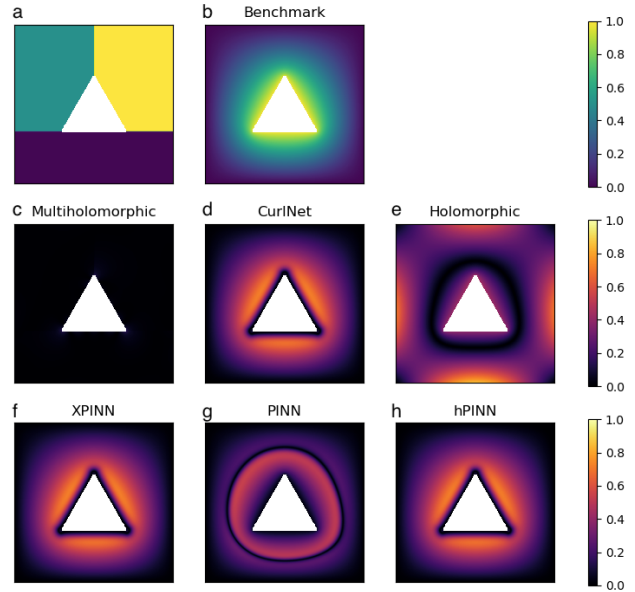

Figure 2. (a) A partitioning of the multiply-connected domain in Sect.3.5 into three simply-connected domains on which we define our multiholomorphic and XPINN networks. Note that the white triangle is not considered part of the domain and is excluded from the analysis. (b) The solution $\phi(x,y)_{\text{FEM}}$ of the heater problem described in Sect. 3.5, as provided by a FEM solver for benchmark. (c-h) A comparison of the performance for different neural network solutions, by plotting the absolute error of each network compared against $\phi(x,y)_{\text{FEM}}$ given respectively by (c) a Multiholomorphic neural network, (d) a CurlNet architecture, (e) a Holomorphic network, (f) an XPINN domain decomposition based approach, (g) a regular PINN and (h) an hPINN architecture with Dirichlet boundary conditions included in the network architecture. Additional details and definitions are provided in the main text. All plots report a sample of the reference metric on a uniform 512×512 grid. *Results for the same test case adopting various quantum architectures are reported in App. Fig. 6.*

mulation. We partition our domain as outlined in Fig. 2a for both XPINN and multiholomorphic approaches, and report our results in Table 1. Note that as in section 3.3, we note that a low expected absolute Laplacian does not necessarily correspond to a good solution in terms of RMSE. All the figures of merit for this case are equivalent to those outlined in Sect. 3.3.

3.6. Results D: Robot Navigation in a Previously Explored Environment

We consider a robot navigating a known, static environment, e.g. a pre-explored corridor. The use of general potential to guide robot movement is well-established, and it can present evident advantages over traditional path-finding algorithms (Rimon, 1990). For example, it offers a symbolic, parameterized representation of the environment that can be easily updated with new information, as well as exploiting special properties of the mechanical system to ensure more stringent adherence to problem constraints, such as e.g. a maximum available torque to the robot. In particular, it has been noted that harmonic functions have favourable properties for robot navigation (Connolly & Grupen, 1993): harmonic functions do not attain local minima anywhere (except for constant harmonic functions), which prevents navigation paths from becoming stuck in such minima. This property led to a few dedicated studies (Kazemi & Mehrandezh, 2004; Loizou, 2014).

Define a domain $\Omega = ([0, 0.5] \times [0, 0.6]) \cup ([0.5, 1.0] \times [0.4, 1.0])$. We construct a harmonic function, ϕ with Dirichlet boundary conditions $\phi = -1$ at $y = 0$ and $\phi = 1$ at $y = 1$. A summary of resulting paths and errors in the potential relative to finite element benchmarks can be found in Fig. 3.

We note that even though the holomorphic network did not characterise the underlying potential field as accurately as the CurlNet, it provided the most consistent robot navigation paths. This might be explained by the high errors in Laplacian measured in the CurlNet approach.

That CurlNet offers a more accurate potential despite having a higher Laplacian is noteworthy. In this particular spatial domain, we note that there are rapidly changing conservative fields (see Fig.3b. The CurlNet approach is able to match this behaviour well (Fig.3e), but at the expense of introducing order of magnitude increases in its Laplacian (Fig. 3f). However, holomorphic and PINN approaches directly constrain a for a zero Laplacian ($\frac{\partial^2 \phi}{\partial x^2} + \frac{\partial^2 \phi}{\partial y^2} = 0$). Evidently, optimising for a zero Laplacian but a large gradient is a difficult optimisation task. Consequently, in order to maintain a low Laplacian, we note that the conservative field in holomorphic and PINN approaches show high errors (Fig.3h and k). In order to maintain low Laplacians in these regions
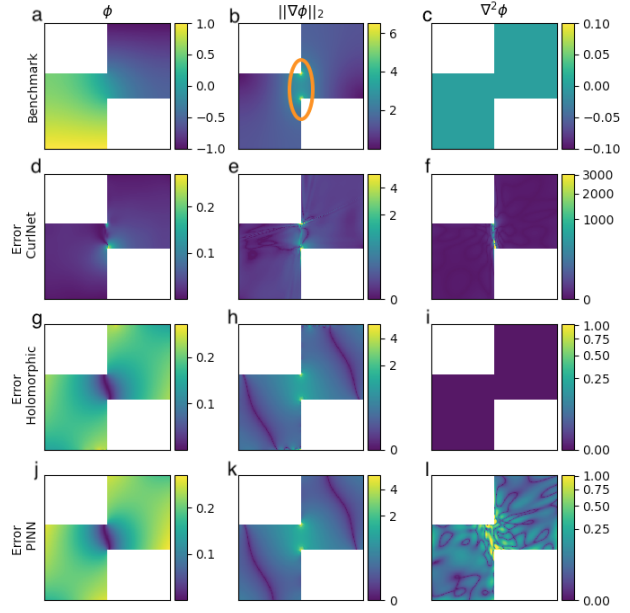

Figure 3. Benchmark and sample errors from the CurlNet, Holomorphic and PINN approaches in the robot navigation task. The orange oval in subplot b highlights an area of rapidly changing conservative field, which can be difficult for models to fit effectively. (a) Baseline potential (i.e. ϕ) solution generated by finite elements. (b) Baseline magnitude of conservative field (i.e. $\|\nabla\phi\|_2$) calculation generated by finite elements (c) The exact reference Laplacian (i.e. identically zero). (d) The absolute error in the potential from a sample CurlNet run. (e) The absolute error in the magnitude of the conservative field in the CurlNet solution. (f) The absolute error in the Laplacian of the CurlNet solution. (g) The absolute error in the potential of the holomorphic solution. (h) The absolute error in the magnitude of the conservative field in the holomorphic solution. (i) The absolute error in Laplacian in the holomorphic solution. (j) The absolute error in the potential of the PINN solution. (k) The absolute error in the magnitude of the conservative field in the PINN solution. (l) The absolute error in Laplacian in the PINN solution. Please note the log colour scale of subplot f, showing very high Laplacians in a small central region. *Results for the same test case adopting various quantum architectures are reported in App. Fig. 7.*

See Table 1 for a full comparison of mean squared errors in the potentials and Laplacian's over valid points (i.e. within the environment accessible to the robot) extracted from a uniform 64×64 grid, for both classical methods and their previously introduced quantum counterparts.

3.7. Results E: Potential Flow Through a 3D Pipe

We consider fluid flowing through a square pipe and study potential flow, denoted by ϕ. We take the domain to be the unit cube: $\Omega = [0, 1]^3$ with a boundary that we denote by $\partial\Omega$. As boundary conditions, we impose $\phi(0.0, y, z) = 1.0$,

$\phi(1.0, y, z) = -1.0$ and $\phi(x, y, z) = 0.0$ everywhere else on $(x, y, z) \in \partial\Omega$.

We benchmark the classical methodologies suitable for problems in dimensions higher than two on this system, where we find favourable performance relative to PINNs, as outlined in Table 1.

4. Discussion

In this work we demonstrated the inclusion of inductive biases in classical and quantum neural networks alike towards harmonic functions, observing how despite their crucial importance, there had been only few attempts at including specifically harmonic inductive biases. In summary, we constructed exactly harmonic functions in two dimensions in simply-connected domains, harmonic functions almost everywhere in two-dimensional multiply-connected domains, and approximately harmonic functions in arbitrary domains. We demonstrated our approach with comparisons to (quantum) physics-informed neural networks (Raissi et al., 2019; Kyriienko et al., 2021) in a range of tasks, namely heat distribution, electrostatics and robot navigation. Previous tools in this domain were mostly limited to finite-element modelling (Galybin & Irša, 2010), which is less suited for problems such as inverse design and transfer learning. Training upon conventional neural networks as well as simulated quantum circuits confirmed the benefits of our newly proposed architectures. Interestingly, we note that achieving a low Laplacian in a given solution is not always consistent with achieving low mean squared errors against a trustworthy benchmark. It is perhaps unsurprising that PINNs typically have low differential equation residuals when trained since they optimise against PDE residuals directly. However, it is interesting that this might not necessarily be the best strategy for achieving a low error in the resulting solution, as evidenced by the fact that our CurlNet architectures consistently achieve lower RMSEs than their PINN counterparts, whilst exhibiting higher errors in their Laplacians, which we explain further in Sect.3.6. Besides the merits in all those applications where harmonic functions are relevant, our work highlights how developments in conventional machine learning architectures can be sometimes readily ported to the corresponding realm of quantum machine learning, here exemplified by trainable, parameterised quantum circuit architectures. Potential future work might include extending holomorphic functions to higher dimensions. We note that there are connections between so-called regular quaternionic functions and harmonic functions that might facilitate such advancements (Sudbery, 1979). We hope that our findings encourage further cross-pollination between the quantum and classical machine learning fields.

5. Acknowledgements

References

Supplementary code. doi: 10.6084/m9.figshare.23260154. v2. URL https://figshare.com/articles/ software/Code_accompanying_the_paper_ Harmonic_Neural_Networks_ICML_2023_ /23260154.

Abbas, A., Sutter, D., Zoufal, C., Lucchi, A., Figalli, A., and Woerner, S. The power of quantum neural networks. *Nature Computational Science*, 1(6):403–409, 2021.

Akduman, I. and Kress, R. Electrostatic imaging via conformal mapping. *Inverse Problems*, 18(6):1659, 2002.

Alnæs, M., Blechta, J., Hake, J., Johansson, A., Kehlet, B., Logg, A., Richardson, C., Ring, J., Rognes, M. E., and Wells, G. N. The fenics project version 1.5. *Archive of Numerical Software*, 3(100), 2015.

Anderson, J. *EBOOK: Fundamentals of Aerodynamics (SI units)*. McGraw hill, 2011.

Anschuetz, E. R. and Kiani, B. T. Beyond barren plateaus: Quantum variational algorithms are swamped with traps. *arXiv preprint arXiv:2205.05786*, 2022.

Arute, F., Arya, K., Babbush, R., Bacon, D., Bardin, J. C., Barends, R., Biswas, R., Boixo, S., Brandao, F. G., Buell, D. A., et al. Quantum supremacy using a programmable superconducting processor. *Nature*, 574(7779):505–510, 2019.

Bak, J., Newman, D. J., and Newman, D. J. *Complex analysis*, volume 8. Springer, 2010.

Bassey, J., Qian, L., and Li, X. A survey of complex-valued neural networks. *arXiv preprint arXiv:2101.12249*, 2021.

Bergholm, V., Izaac, J., Schuld, M., Gogolin, C., Alam, M. S., Ahmed, S., Arrazola, J. M., Blank, C., Delgado, A., Jahangiri, S., et al. Pennylane: Automatic differentiation of hybrid quantum-classical computations. *arXiv preprint arXiv:1811.04968*, 2018.

Bharti, K. and Haug, T. Iterative quantum-assisted eigensolver. *Physical Review A*, 104(5):L050401, 2021.

Both, G.-J., Choudhury, S., Sens, P., and Kusters, R. Deepmod: Deep learning for model discovery in noisy data. *Journal of Computational Physics*, 428:109985, 2021.

Cerezo, M., Arrasmith, A., Babbush, R., Benjamin, S. C., Endo, S., Fujii, K., McClean, J. R., Mitarai, K., Yuan, X., Cincio, L., et al. Variational quantum algorithms. *Nature Reviews Physics*, 3(9):625–644, 2021.

Connolly, C. I. and Grupen, R. A. The applications of harmonic functions to robotics. *Journal of robotic Systems*, 10(7):931–946, 1993.

Cranmer, M., Greydanus, S., Hoyer, S., Battaglia, P., Spergel, D., and Ho, S. Lagrangian neural networks. *arXiv preprint arXiv:2003.04630*, 2020.

Elsken, T., Metzen, J. H., and Hutter, F. Neural architecture search: A survey. *The Journal of Machine Learning Research*, 20(1):1997–2017, 2019.

Gaier, D. *Lectures on complex approximation*, volume 188. Springer, 1987.

Galybin, A. and Irša, J. On reconstruction of three-dimensional harmonic functions from discrete data. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 466(2119):1935–1955, 2010.

Gentile, A. A., Flynn, B., Knauer, S., Wiebe, N., Paesani, S., Granade, C. E., Rarity, J. G., Santagati, R., and Laing, A. Learning models of quantum systems from experiments. *Nature Physics*, 17(7):837–843, 2021.

Giles, C. L. and Maxwell, T. Learning, invariance, and generalization in high-order neural networks. *Applied optics*, 26(23):4972–4978, 1987.

Goto, T., Tran, Q. H., and Nakajima, K. Universal approximation property of quantum machine learning models in quantum-enhanced feature spaces. *Phys. Rev. Lett.*, 127: 090506, Aug 2021.

Granade, C. E., Ferrie, C., Wiebe, N., and Cory, D. G. Robust online Hamiltonian learning. *New Journal of Physics*, 14(10):0–31, oct 2012. ISSN 13672630. doi: 10.1088/1367-2630/14/10/103013.

Greiter, M., Schnells, V., and Thomale, R. Method to identify parent hamiltonians for trial states. *Physical Review B*, 98(8):081113, 2018.

Greydanus, S., Dzamba, M., and Yosinski, J. Hamiltonian neural networks. *Advances in Neural Information Processing Systems*, 32, 2019.

Griffiths, D. J. Introduction to electrodynamics, 2005.

Grimsley, H. R., Economou, S. E., Barnes, E., and Mayhall, N. J. An adaptive variational algorithm for exact molecular simulations on a quantum computer. *Nature communications*, 10(1):1–9, 2019.

Guerreschi, G. G. and Smelyanskiy, M. Practical optimization for hybrid quantum-classical algorithms. *arXiv preprint arXiv:1701.01450*, 2017.

Gunzburger, M. D., Peterson, J. S., and Shadid, J. N. Reduced-order modeling of time-dependent pdes with multiple parameters in the boundary data. *Computer methods in applied mechanics and engineering*, 196(4-6): 1030–1047, 2007.

Hadfield, S., Wang, Z., Rieffel, E. G., O'Gorman, B., Venturelli, D., and Biswas, R. Quantum approximate optimization with hard and soft constraints. In *Proceedings of the Second International Workshop on Post Moores Era Supercomputing*, pp. 15–21, 2017.

Hadfield, S., Wang, Z., O'gorman, B., Rieffel, E. G., Venturelli, D., and Biswas, R. From the quantum approximate optimization algorithm to a quantum alternating operator ansatz. *Algorithms*, 12(2):34, 2019.

Harris, C. R., Millman, K. J., Van Der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., et al. Array programming with numpy. *Nature*, 585(7825):357–362, 2020.

Harrow, A. W., Hassidim, A., and Lloyd, S. Quantum algorithm for linear systems of equations. *Phys. Rev. Lett.*, 103:150502, Oct 2009.

He, K., Zhang, X., Ren, S., and Sun, J. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pp. 1026–1034, 2015.

Heim, N., Ghosh, A., Kyriienko, O., and Elfving, V. E. Quantum model-discovery. *arXiv preprint arXiv:2111.06376*, 2021.

Hendriks, J., Jidling, C., Wills, A., and Schön, T. Linearly constrained neural networks. *arXiv preprint arXiv:2002.01600*, 2020.

Huang, H.-Y., Broughton, M., Mohseni, M., Babbush, R., Boixo, S., Neven, H., and McClean, J. R. Power of data in quantum machine learning. *Nature Communications*, 12(1):2631, May 2021. ISSN 2041-1723.

Hunter, J. D. Matplotlib: A 2d graphics environment. *Computing in science & engineering*, 9(03):90–95, 2007.

Jackson, J. D. Classical electrodynamics, 1999.

Jagtap, A. D. and Karniadakis, G. E. Extended physics-informed neural networks (xpinns): A generalized space-time domain decomposition based deep learning framework for nonlinear partial differential equations. *Communications in Computational Physics*, 28(5):2002–2041, 2020.

Karniadakis, G. E., Kevrekidis, I. G., Lu, L., Perdikaris, P., Wang, S., and Yang, L. Physics-informed machine learning. *Nature Reviews Physics*, 3(6):422–440, 2021.

Kazemi, M. and Mehrandezh, M. Robotic navigation using harmonic function-based probabilistic roadmaps. In *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA'04. 2004*, volume 5, pp. 4765–4770. IEEE, 2004.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Kübler, J., Buchholz, S., and Schölkopf, B. The inductive bias of quantum kernels. *Advances in Neural Information Processing Systems*, 34, 2021.

Kurt, H., Maxwell, S., and Halbert, W. Multilayer feed-forward networks are universal approximators. *Neural Networks*, 2(5):359–366, 1989. ISSN 0893-6080.

Kyriienko, O. and Elfving, V. E. Generalized quantum circuit differentiation rules. *Physical Review A*, 104(5):052417, 2021.

Kyriienko, O., Paine, A. E., and Elfving, V. E. Solving nonlinear differential equations with differentiable quantum circuits. *Physical Review A*, 103(5):052416, 2021.

Lagaris, I. E., Likas, A., and Fotiadis, D. I. Artificial neural networks for solving ordinary and partial differential equations. *IEEE transactions on neural networks*, 9(5):987–1000, 1998.

Larocca, M., Sauvage, F., Sbahi, F. M., Verdon, G., Coles, P. J., and Cerezo, M. Group-invariant quantum machine learning. *arXiv preprint arXiv:2205.02261*, 2022.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Lee, J., Huggins, W. J., Head-Gordon, M., and Whaley, K. B. Generalized unitary coupled cluster wave functions for quantum computation. *Journal of chemical theory and computation*, 15(1):311–324, 2018.

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.

Liu, D. C. and Nocedal, J. On the limited memory bfgs method for large scale optimization. *Mathematical programming*, 45(1):503–528, 1989.

Loizou, S. G. The multi-agent navigation transformation: tuning-free multi-robot navigation. In *Robotics: Science and Systems*, volume 6, pp. 1516–1523, 2014.

Lu, L., Jin, P., and Karniadakis, G. E. Deeponet: Learning nonlinear operators for identifying differential equations based on the universal approximation theorem of operators. *arXiv preprint arXiv:1910.03193*, 2019.

Lu, L., Pestourie, R., Yao, W., Wang, Z., Verdugo, F., and Johnson, S. G. Physics-informed neural networks with hard constraints for inverse design. *SIAM Journal on Scientific Computing*, 43(6):B1105–B1132, 2021.

Luo, X.-Z., Liu, J.-G., Zhang, P., and Wang, L. Yao. jl: Extensible, efficient framework for quantum algorithm design. *Quantum*, 4:341, 2020.

Mari, A., Bromley, T. R., and Killoran, N. Estimating the gradient and higher-order derivatives on quantum hardware. *Physical Review A*, 103(1):012405, 2021.

McArdle, S., Jones, T., Endo, S., Li, Y., Benjamin, S. C., and Yuan, X. Variational ansatz-based quantum simulation of imaginary time evolution. *npj Quantum Information*, 5 (1):75, Sep 2019. ISSN 2056-6387.

McClean, J. R., Boixo, S., Smelyanskiy, V. N., Babbush, R., and Neven, H. Barren plateaus in quantum neural network training landscapes. *Nature communications*, 9 (1):1–6, 2018.

Mernyei, P., Meichanetzidis, K., and Ceylan, I. I. Equivariant quantum graph circuits. *arXiv preprint arXiv:2112.05261*, 2021.

Mitarai, K., Negoro, M., Kitagawa, M., and Fujii, K. Quantum circuit learning. *Phys. Rev. A*, 98:032309, Sep 2018.

Mohseni, M., Rezakhani, A. T., and Lidar, D. A. Quantum-process tomography: Resource analysis of different strategies. *Physical Review A - Atomic, Molecular, and Optical Physics*, 77(3):32322, 2008. ISSN 10502947. doi: 10.1103/PhysRevA.77.032322.

Motta, M., Sun, C., Tan, A. T. K., O'Rourke, M. J., Ye, E., Minnich, A. J., Brandão, F. G. S. L., and Chan, G. K.-L. Determining eigenstates and thermal states on a quantum computer using quantum imaginary time evolution. *Nature Physics*, 16(2):205–210, Feb 2020. ISSN 1745-2481.

Nielsen, M. A. and Chuang, I. Quantum computation and quantum information, 2002.

Paine, A. E., Elfving, V. E., and Kyriienko, O. Quantum quantile mechanics: Solving stochastic differential equations for generating time-series. *arXiv preprint arXiv:2108.03190*, 2021.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga,

L., et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

Perot, J. B. and Zusi, C. J. Differential forms for scientists and engineers. *Journal of Computational Physics*, 257: 1373–1393, 2014.

Prestes e Silva, E., Engel, P. M., Trevisan, M., and Idiart, M. A. Exploration method using harmonic functions. *Robotics and Autonomous Systems*, 40(1):25–42, 2002. ISSN 0921-8890.

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*, 378:686–707, 2019.

Redner, S. *A guide to first-passage processes*. Cambridge university press, 2001.

Richter-Powell, J., Lipman, Y., and Chen, R. T. Neural conservation laws: A divergence-free perspective. *arXiv preprint arXiv:2210.01741*, 2022.

Rimon, E. *Exact robot navigation using artificial potential functions*. Yale University, 1990.

Schuld, M. Supervised quantum machine learning models are kernel methods, 2021.

Schuld, M. and Killoran, N. Quantum machine learning in feature hilbert spaces. *Physical review letters*, 122(4): 040504, 2019.

Schuld, M. and Petruccione, F. *Machine learning with quantum computers*. Springer, 2021.

Sharma, R., Farimani, A. B., Gomes, J., Eastman, P., and Pande, V. Weakly-supervised deep learning of heat transport via physics informed loss. *arXiv preprint arXiv:1807.11374*, 2018.

Shor, P. Algorithms for quantum computation: discrete logarithms and factoring. In *Proceedings 35th Annual Symposium on Foundations of Computer Science*, pp. 124–134, 1994.

Sudbery, A. Quaternionic analysis. In *Mathematical Proceedings of the Cambridge Philosophical Society*, volume 85, pp. 199–225. Cambridge University Press, 1979.

Varsamopoulos, S., Philip, E., van Vlijmen, H. W. T., Menon, S., Vos, A., Dyubankova, N., Torfs, B., Rowe, A., and Elfving, V. E. Quantum extremal learning. *arXiv preprint arXiv:2205.02807*, 2022.

Weintraub, S. H. *Differential forms: theory and practice*. Elsevier, 2014.

Yuan, X., Endo, S., Zhao, Q., Li, Y., and Benjamin, S. C. Theory of variational quantum simulation. *Quantum*, 3: 191, October 2019. ISSN 2521-327X.

Zhong, H.-S., Wang, H., Deng, Y.-H., Chen, M.-C., Peng, L.-C., Luo, Y.-H., Qin, J., Wu, D., Ding, X., Hu, Y., et al. Quantum computational advantage using photons. *Science*, 370(6523):1460–1463, 2020.

Appendix

A. Details on Quantum Approaches

A.1. Exactly Harmonic Quantum Circuits

We define a quantum variational state $|\Psi_{\theta,x,y}\rangle$, parametrised by θ-dependent unitaries $\hat{\mathcal{U}}_{\theta_k}$ as well as a quantum feature map (QFM) - e.g. of Chebyshev tower type (Kyriienko et al., 2021) - used to port the input variables to the Hilbert space acted upon by $\hat{\mathcal{U}}_{\theta_k}$:

$$|\Psi_{\theta,x,y}\rangle = \hat{\mathcal{U}}_{\theta_2}\mathrm{QFM}(x,y)\hat{\mathcal{U}}_{\theta_1}|\emptyset\rangle; \tag{15}$$

where $|\emptyset\rangle$ represents a reference state. A QNN is typically modelled as a measurement of a cost Hamiltonian $\hat{\mathcal{H}}$:

$$\phi_Q(x,y) = \langle\Psi_{\theta,x,y}|\hat{\mathcal{H}}|\Psi_{\theta,x,y}\rangle. \tag{16}$$

Here, instead, we modify a 'quantum kernel' approach (Schuld & Killoran, 2019; Schuld, 2021) to introduce a harmonic architecture.

Theorem A.1. *Adopting an imaginary feature map of the form IQFM$(x,y) = e^{-(x+iy)\hat{\mathcal{H}}}$, the output obtained from the circuit as:*

$$\phi_{QH}(x,y) = \Re\{\langle\emptyset|\Psi_{\theta,x,y}\rangle\} \tag{17}$$

is harmonic over the inputs x,y.

Remark A.2. For $x=0$, IQFM(x,y) represents a unitary evolution generated by $\hat{\mathcal{H}}$, similar to the Hamiltonian evolution feature maps in (Kyriienko et al., 2021; Kyriienko & Elfving, 2021). For non-zero x, the operation is generally non-unitary, requiring Quantum Imaginary Time Evolution (QITE) to be implemented with quantum gates in hardware (McArdle et al., 2019; Yuan et al., 2019; Motta et al., 2020).

Proof. The bulk of the proof is reported for brevity in Appendix A.3, where we show how extending QITE formalism, we can symbolically represent:

$$\phi_{QH}(x,y) = \Re\{\sum_m c_m(\theta_1)\langle\emptyset|\hat{\mathcal{U}}_{\theta_2}|\psi_m\rangle e^{-(x+iy)E_m}\}. \tag{18}$$

where E_m is the m-th (scalar) eigenvalue of Hamiltonian $\hat{\mathcal{H}}$ and the state $|\Psi\rangle$ was decomposed into the basis set $|\psi_m\rangle$ with coefficients $c_m(\theta_1)$, summing over all 2^N eigenvalues m for an N-qubit Hamiltonian. Proven Eq. 18, we can then observe how ϕ_{QH} represents the real part of a sum over an exponential number of holomorphic functions (with coefficients tuned by circuit parameters θ_1 and θ_2), and is therefore in total harmonic. $\qquad\square$

We can thus generalise to harmonic quantum neural networks the construct in equation (1), where $\mathrm{NN}_h(x+iy) = \langle\emptyset|\Psi_{\theta,x,y}\rangle$.

A.2. Further Details about (Inductive Biases for) Quantum Algorithms

Quantum computing is a computational paradigm that holds theoretical promise for exponential speedups over conventional computers in, for example, factoring prime numbers (Shor, 1994) and solving linear systems of equations (Harrow et al., 2009). However, the practical application of such provably advantageous algorithms is hindered by the lack of fault-tolerant quantum computers, which are still thought to be years away in terms of development. In spite of such theoretically-guaranteed computational advantages being limited to future devices, there have already been indications of experimental quantum advantage having been achieved for certain restricted classes of problems (Arute et al., 2019; Zhong et al., 2020). While such experimental advantages have been demonstrated, it remains to find advantages in applications which have wider practical application, leading to the development of the variational quantum algorithms (Cerezo et al., 2021) and QNNs discussed in the main paper. In particular, QNNs have been used for classification and regression tasks (Mitarai et al., 2018), solving differential equations (Kyriienko et al., 2021; Paine et al., 2021), model discovery (Heim et al., 2021) and extremal learning (Varsamopoulos et al., 2022).

These quantum NNs (QNNs) hold the promise of being advantageous in terms of the *expressivity* (Schuld & Killoran, 2019; Abbas et al., 2021), offered by parameterised quantum circuits, whereas their *trainability* is susceptible to issues (McClean et al., 2018; Anschuetz & Kiani, 2022), e.g. large regions of parameter space exhibit exponentially-vanishing parameter gradients with respect to loss functions, so-called 'barren plateaus'.

One successful means to mitigate barren plateaus has been to include inductive biases within quantum circuit designs (Hadfield et al., 2019; Bharti & Haug, 2021). It has even been argued that, in some settings, including inductive biases in quantum circuits is a prerequisite for quantum advantage (Kübler et al., 2021). Including inductive biases in conventional NNs has been a staple of progress in machine learning, e.g. ranging from convolutional neural networks (LeCun et al., 1998) and geometrically-invariant networks (Giles & Maxwell, 1987) through to recent examples such as Hamiltonian neural networks (Greydanus et al., 2019) and physics-informed neural networks (Raissi et al., 2019). However, the use of inductive biases in designing quantum circuits and quantum machine learning models is still a nascent field, though there is a growing research interest on this front (Mernyei et al., 2021; Larocca et al., 2022). the though there is a growing research interest on this front (Mernyei et al., 2021; Larocca et al., 2022). In the context of variational quantum algorithms, inductive biases are often referred to as problem-inspired ansätze. A well-established example is the unitary coupled cluster ansatz for modelling the ground-state energy of molecular Hamiltonians (Lee et al., 2018). Hard constraints in ansätze for combinatorial optimization have also been previously reported (Hadfield et al., 2017). Analogously to how neural architecture search (Elsken et al., 2019) might learn architectures with given inductive biases, there have been developments on adaptively constructing ansätze for molecular simulations (Grimsley et al., 2019). Another well-known paradigm concerning the adoption of inductive biases in the context of variational quantum algorithms refers to the characterisation of quantum systems. In this case, learning a full dynamic map describing a quantum process (i.e. performing ab initio a quantum tomography) is unfeasible already for systems of very few qubits (Mohseni et al., 2008). However, the system dynamics can be modelled constraining such maps by knowledge of (parametrised) generators of the dynamics (e.g. the system's Hamiltonian (Granade et al., 2012)) and of their eigenstates (Greiter et al., 2018). Similarly to aforementioned ansätze in molecular simulations, such parametrised generators can also be made adaptive (Gentile et al., 2021).

A.3. Holomorphic Quantum Circuits and Harmonic Quantum Neural Networks

Our aim is to construct quantum circuits which exhibit holomorphic properties.
In a general form, a QNN model can be written as

$$\phi_Q(\boldsymbol{x}) = \langle \emptyset | \hat{\mathcal{U}}(\boldsymbol{x}, \boldsymbol{\theta})^\dagger \hat{\mathcal{H}}(\boldsymbol{x}, \boldsymbol{\theta}) \hat{\mathcal{U}}(\boldsymbol{x}, \boldsymbol{\theta}) | \emptyset \rangle, \tag{19}$$

where the unitary circuit ansatz $\hat{\mathcal{U}}(\boldsymbol{x}, \boldsymbol{\theta})$, and the Hermitian cost Hamiltonian $\hat{\mathcal{H}}(\boldsymbol{x}, \boldsymbol{\theta})$, are functions of function-input variables and model parameters $\boldsymbol{\theta}$, and $|\emptyset\rangle$ represents the 0-state, i.e. a reference state for the quantum computation. Alternatively, a different type of quantum model can be written as

$$\phi_{QH}(\boldsymbol{x}) = \Re \langle \emptyset | \hat{\mathcal{U}}(\boldsymbol{x}, \boldsymbol{\theta}) | \emptyset \rangle, \tag{20}$$

which represents the real part of a complex number obtained from measuring the overlap of the parametrized state (i.e. the reference 0-state evolved by the unitary $\hat{\mathcal{U}}$, for example prepared on a register of qubits), with the 0-state itself. Such overlap can be computed using e.g. the Hadamard or SWAP tests (Nielsen & Chuang, 2002).
To compute the value of $\phi_Q(\boldsymbol{x})$ or $\phi_{QH}(\boldsymbol{x})$, or even approximate it, is believed to be very computationally intensive on a classical computer for systems involving large unitaries $\hat{\mathcal{U}}$ (which in turn corresponds to the size of the target system). On the contrary, this is typically feasible on a quantum computer, provided some assumption on the circuit structure, and eventually the cost Hamiltonian for $\phi_Q(\boldsymbol{x})$ (Arute et al., 2019).
For simplicity and ease of analysis, we here restrict ourselves to a specific structure of the cost Hamiltonian as being independent of $\boldsymbol{x}$ and $\boldsymbol{\theta}$, i.e. $\hat{\mathcal{H}}(\boldsymbol{x}, \boldsymbol{\theta}) = \hat{\mathcal{H}}$, and the circuit structure as

$$\hat{\mathcal{U}}(\boldsymbol{x}, \boldsymbol{\theta}) = \hat{\mathcal{U}}_{\theta_2} \text{QFM}(\boldsymbol{x}) \hat{\mathcal{U}}_{\theta_1}, \tag{21}$$

is then a single quantum feature map $\text{QFM}(\boldsymbol{x})$, squeezed between two general variational circuits $\hat{\mathcal{U}}_{\theta_j}$ (Kyriienko et al., 2021) which only depend on model parameters $\boldsymbol{\theta}_j$.

To describe a particular type of QFM in the form of a Hamiltonian evolution, we first make use of the following decomposition of a Hamiltonian evolution applied to some arbitrary initial state $|\Psi_{\text{ini}}\rangle$:

$$e^{-it\hat{\mathcal{G}}} |\Psi_{\text{ini}}\rangle = \sum_m c_m e^{-itE_m} |\psi_m\rangle. \tag{22}$$

For the examples in our paper $\boldsymbol{x} \equiv (x, y)$ - as we only deal with 2D problems - and for a QFM of the form in (22), we can think that the dependency upon the two variables can be expressed via the time:

$$t = t(x, y). \tag{23}$$

Inserting this expression in (19), where $|\Psi_{\text{ini}}\rangle \equiv |\emptyset\rangle$, using the assumption (21), and reminding that the decomposition in (22) involves orthonormal $|\psi_m\rangle$, which are also eigenstates of $\hat{\mathcal{G}}$, i.e. $\hat{\mathcal{G}}|\psi_m\rangle = E_m|\psi_m\rangle$, we find:

$$\phi_Q(x,y) = \langle\Psi(\theta_1)|e^{it\hat{\mathcal{G}}}\hat{\tilde{\mathcal{H}}}(\theta_2)e^{-it\hat{\mathcal{G}}}|\Psi(\theta_1)\rangle = \sum_{\langle m,n\rangle} c_{mn}(\theta_1,\theta_2)e^{-it(E_m-E_n)} \tag{24}$$

where $\hat{\tilde{\mathcal{H}}} = \hat{\mathcal{U}}_{\theta_2}^\dagger \hat{\mathcal{H}}\hat{\mathcal{U}}_{\theta_2}$ indices $\langle m,n\rangle$ sum over all m and n from 1 to the size of the accessible Hilbert space (i.e. 2^N for an N-qubit register). While c_{mn} are generally complex-valued, the resulting function output is guaranteed to be real-valued. For a QNN of the form $\phi_{QH}(x,y)$ we find

$$\phi_{QH}(x,y) = \Re\{\sum_m c_m(\theta_1)\langle\emptyset|\hat{\mathcal{U}}_{\theta_2}|\psi_m\rangle e^{-itE_m}\} = \Re\{\sum_m \tilde{c}_m(\theta_1,\theta_2)e^{-itE_m}\}. \tag{25}$$

If we now consider a time evolution in (23) of the form $t(x,y) = y - xi$, we obtain the result from the main text in Eq. 4. This yields a sum over exponentials of holomorphic functions, which is thus holomorphic itself and has a harmonic real part. With this choice, we find the quantum models to be represented by linear combinations of $e^{-(x+iy)E}$ with E some number representing an eigenvalue, or gap between two eigenvalues, of the QFM evolution Hamiltonian. The number of unique terms is therefore equal to the number of unique (non-degenerate) eigenvalues or gaps in the spectrum of the QFM evolution Hamiltonian. The more unique terms, the more basis functions and thus the more expressivity the quantum model can obtain. We describe here the specific complex-exponential QFM we chose for the results section

$$\text{QFM}(x,y) = e^{(x-iy)\pi\hat{\mathcal{H}}} \quad \text{where} \quad \hat{\mathcal{H}} = \sum_{j=0}^{N-1} 2^j \hat{Z} + 2^N. \tag{26}$$

this Hamiltonian has 2^N unique eigenvalues $E_m = 2m - 1 = \{1, 3, 5, 7 \ldots (2 \cdot 2^N - 1)\}$.

A.4. Applying Harmonic QNNs to Solve an Instance of the Laplace Equation

In the main paper, Sect. 4.4 (Results B), we considered the solution of the Laplace equation for the heat distribution $f(x,y)$ inside a unit-square box refrigerated on all sides (where then the Dirichlet b.c. applicable impose $f|_{\partial\Omega_{1-3}} = 0$) but one (occurring at $x = 0$), where we assume the presence of a heater, modelled by imposing a Dirichlet b.c. $f|_{\partial\Omega_4} = 1$. The general solution to this problem setting is known in closed form as a series expansion

$$f(x,y) = \sum_{n \text{ odd}}^{\infty} \frac{4}{n\pi}e^{-n\pi x}\sin(n\pi y) = \frac{2}{\pi}\arctan\frac{\sin(\pi y)}{\sinh \pi x} \tag{27}$$

The exact solution can thus also be numerically approximated; as the coefficients become smaller with the term index, by truncating the sum to a given number of terms $\tilde{N}$, the solution becomes closer and closer to exact. We note that expansions equivalent to (27) exist also for cases where $f|_{\partial\Omega_4} = f(y)$, and for the general case of a 2D Laplace equation with Dirichlet b.c., it is expected that alternative expansions can naturally approximate the solution. However, in general, the analytical function representation of the series is not always known, and therefore numerical truncation methods are the best alternative. Having access to a large number of terms improves the approximation.

In Fig. 4 we plot the converged results and the loss convergence profile for a QNN adopting $N = 4$ qubits. We find the solution matches very well overall, with only a slight deviation at the $x = 0$ boundary, for all y, with clear oscillatory behaviour. This can be explained by the so-called Gibbs phenomenon, which is a demonstration that step-function behaviour can in principle be modelled accurately with periodic functions, but the amplitude error remains significant even for a larger cut-off in the series. The step-function is required by the sharp transition $\lim_{y\to 0^+} f(x = 0, y) = 1$ whilst $\lim_{y\to 0^-} f(x = 0, y) = 0$, which occurs at the corners of the unit-box for $x = 0$ due to the discontinuous b.c. imposed.

We observe very similar profiles (results omitted) for the solution represented by a finite sum, with a cut-off at 16 terms. This confirms the intuition about the qHolomorphic circuit expressivity outlined in the last paragraph: the $N = 4$ case employs $2^N = 16$ terms so that we expect a performance similar to expressing the solution by introducing the same number of terms in (27) - where the coefficients of the terms are fixed and not variationally optimised. This intuitive reasoning signifies the capacity of the qHolomorphic circuit to represent basis functions to the Laplace problem.
Finally, we stress how the qHolomorphic architecture converges very rapidly to the approximate solution, in less than 100 epochs.
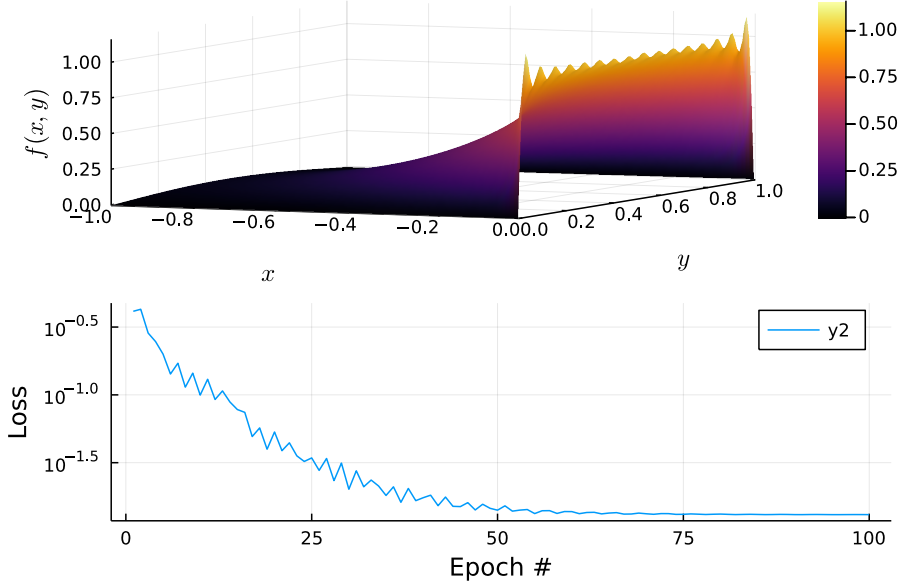
Figure 4. (Top) Converged QNN output for solving Laplace's equation on a unit square domain with boundary conditions setting three sides to $f(x, y) = 0$ and one side to $f(x, y) = 1$. (Bottom) Loss profile as a function of epoch # in the QNN training process.

A.5. Quantum Imaginary- and Complex-Time Evolution

So far we have assumed we can directly substitute a complex number into the t variable representing time evolution over a Hamiltonian. But how would one implement complex-time evolution in a real quantum system? In this section, we will highlight strategies known in the literature as Quantum Imaginary Time Evolution (QITE) and extend these to general Quantum Complex Time Evolution (QCTE).

QITE is a strategy that was first used in classical computing, in particular in the field of computational quantum chemistry, to prepare ground-states/energies for certain types of Hamiltonians, including molecular and lattice Hamiltonians. More recently, it was shown that quantum algorithms could offer a substantial speedup over classical methods. There are many variations of QITE, but two main approaches are the Trotterized-version by Motta et al. (Motta et al., 2020) and the variational approach described by McArdle et al. (McArdle et al., 2019).

In Motta et al. (Motta et al., 2020), the overall imaginary time evolution step is expanded into a series of Trotter steps. Each step evolves over only a very short time. Based on locality properties, one can construct an approximate linear system that is solved on a classical computer to find a proxy unitary operator that can replace the non-unitary step, including the appropriate normalization factor. In our work, that normalization factor would be re-multiplied as a scalar number in classical post-processing to capture a desirable decaying effect.

Because the proposal is already based on a Trotterization-approach, it is possible to show that the QITE can be extended to QCTE by interleaving imaginary-time-evolution Trotter steps with real-time-evolution Trotter steps as

$$e^{-\beta(\hat{H}_r + i\hat{H}_i)} = (e^{-\Delta\tau \hat{h}_r[1]} e^{-i\Delta\tau \hat{h}_i[1]} e^{-\Delta\tau \hat{h}_r[2]} e^{-i\Delta\tau \hat{h}_i[2]} \dots)^n + \mathcal{O}(\Delta\tau) \tag{28}$$

where we decomposed both $\hat{H}_r$ and $\hat{H}_i$ into sums of mutually commuting operators $\hat{h}_r[m]$ and $\hat{h}_i[m]$ respectively, and $n = \frac{\beta}{\Delta\tau}$.

McArdle et al. (McArdle et al., 2019) aims to circumvent the impractical circuit depth requirements of (Motta et al., 2020) by taking an approximate, variational approach, now also referred to as VarQITE (Variational QITE). McLachlan's variational principle is used to map the imaginary-time evolved states onto a suitable, chosen variational ansatz, allowing to effectively perform QITE on a quantum device. Also in this case, QCTE proposes to replace the real Hamiltonian term by a more general complex Hamiltonian. However, future work would still be required to adapt the VarQITE estimate routine of the normalization factor to our case.

B. Divergence-Free (Quantum) Neural Networks

We use exterior derivatives on differential forms to construct divergence-free networks in arbitrary dimensions. To construct divergence-free networks in an N-dimensional space, we can represent a general $(N-2)$-form with a neural network. Taking the exterior derivative of this network yields a divergence-free $(N-1)$-form. This follows since

1. There is a correspondence between the exterior derivatives of $(N-1)$-forms and divergence operators.

2. Nested exterior derivatives evaluate to zero.

A full treatment of differential forms lies outside the scope of this work (see e.g. (Perot & Zusi, 2014; Weintraub, 2014) for more detailed coverage), so we focus instead on practical demonstrations aimed to facilitate the derivation of divergence-free neural networks in further work.

B.1. Two Dimensions

While divergence-free fields in two dimensions might be constructed by inspection, we use the two-dimensional case as a showcase of the methodology involving differential forms.

Start with the definition of a general zero-form:

$$f = y\left(x_1, x_2\right), \tag{29}$$

on which we take an exterior derivative:

$$df = \frac{\partial y}{\partial x_2}dx_2 + \frac{\partial y}{\partial x_1}dx_1. \tag{30}$$

Consider an exterior derivative on a general 1-form $g = a\,dx_2 - b\,dx_1$:

$$dg = \left(\frac{\partial a}{\partial x_1}dx_1 + \frac{\partial a}{\partial x_2}dx_2\right)dx_2 - \left(\frac{\partial b}{\partial x_1}dx_1 + \frac{\partial b}{\partial x_2}dx_2\right)dx_1 \tag{31}$$

$$= \left(\frac{\partial a}{\partial x_1} + \frac{\partial b}{\partial x_2}\right)dx_1 dx_2, \tag{32}$$

where we use the identity on differential forms that $dx_i dx_j = -dx_j dx_i$, which also implies that $dx_i dx_i = 0$.

This demonstrates the correspondence between the exterior derivative of an $(N-1)$-form and the divergence operator.

Since $d\left(d\left(f\right)\right) = 0$ for any differential form f, equating coefficients between df and g then yields a formula for a divergence-free field in two dimensions. So, given a map $(x_1, x_2) \rightarrow y$, then the field:

$$\left(\frac{\partial y}{\partial x_2}, -\frac{\partial y}{\partial x_1}\right) \tag{33}$$

is divergence-free.

B.2. Four Dimensions

Since the three-dimensional case is well-handled by standard vector calculus, we turn to four dimensions as a final demonstration of practical calculations of divergence-free fields with differential forms. We note that the increasing complexity of these calculations might make them better suited for calculation using a computer algebra system.

We start by defining a general 2-form in four dimensions:

$$f = y_1 dx_1 dx_2 + y_2 dx_1 dx_3 + y_3 dx_1 dx_4 + y_4 dx_2 dx_3 + y_5 dx_2 dx_4 + y_6 dx_3 dx_4. \tag{34}$$

Taking exterior derivatives results in:

$$df = \left(\frac{\partial y_4}{\partial x_4} - \frac{\partial y_5}{\partial x_3} + \frac{\partial y_6}{\partial x_2} \right) dx_2 dx_3 dx_4$$

$$+ \left(-\frac{\partial y_2}{\partial x_4} + \frac{\partial y_3}{\partial x_3} - \frac{\partial y_6}{\partial x_1} \right) dx_3 dx_4 dx_1$$

$$+ \left(\frac{\partial y_1}{\partial x_4} - \frac{\partial y_3}{\partial x_2} + \frac{\partial y_5}{\partial x_1} \right) dx_4 dx_1 dx_2$$

$$+ \left(-\frac{\partial y_1}{\partial x_3} + \frac{\partial y_2}{\partial x_2} - \frac{\partial y_4}{\partial x_1} \right) dx_2 dx_3 dx_4, \tag{35}$$

where again we use $dx_i dx_j = -dx_j dx_i$ and $dx_i dx_i = 0$.

Considering the exterior derivative of general 3-form $g = a dx_2 dx_3 dx_4 + b dx_3 dx_4 dx_1 + c dx_4 dx_1 dx_2 + d dx_2 dx_3 dx_4$ yields:

$$dg = \left(\frac{\partial a}{\partial x_1} + \frac{\partial b}{\partial x_2} + \frac{\partial c}{\partial x_3} + \frac{\partial d}{\partial x_4} \right) dx_1 dx_2 dx_3 dx_4. \tag{36}$$

Now, equating coefficients between df and g yields a divergence-free network in four dimensions, i.e. given six-dimensional map $(x_1, x_2, x_3, x_4, x_5, x_6) \rightarrow (y_1, y_2, y_3, y_4, y_5, y_6)$ the map

$$\left(\frac{\partial y_4}{\partial x_4} - \frac{\partial y_5}{\partial x_3} + \frac{\partial y_6}{\partial x_2}, \right.$$

$$-\frac{\partial y_2}{\partial x_4} + \frac{\partial y_3}{\partial x_3} - \frac{\partial y_6}{\partial x_1},$$

$$\frac{\partial y_1}{\partial x_4} - \frac{\partial y_3}{\partial x_2} + \frac{\partial y_5}{\partial x_1},$$

$$\left. -\frac{\partial y_1}{\partial x_3} + \frac{\partial y_2}{\partial x_2} - \frac{\partial y_4}{\partial x_1} \right). \tag{37}$$

is divergence-free.

C. Experimental Details

The classical neural networks we use are comparatively lightweight, and training for all conventional NNs was done on a single Apple M1 chip running Python 3.9.12 on macOS 12.3.1. Equivalently, experiments involving QNNs were run on an AMD© Ryzen 7 3700x processor, in a Python 3.9.5 Conda environment on Ubuntu 18.04. Each experiment was repeated 10 times for each method to ensure the repeatability of the outcomes, with the attained average values and standard deviation from the mean reported in the main paper tables.

Quantum Machine Learning   All the QNNs used in this paper are implemented with proprietary code, leveraging upon the packages PyTorch and Yao.jl (Luo et al., 2020). Details of the various components introduced in this paragraph can be found in (Kyriienko et al., 2021). We consider two types of QNNs: for the first, of type (16) with $|\Psi_{\boldsymbol{\theta}, x, y}\rangle = \hat{\mathcal{U}}_{\boldsymbol{\theta}_2} \mathrm{QFM}(x, y) \hat{\mathcal{U}}_{\boldsymbol{\theta}_1} |\emptyset\rangle$, we choose to decompose the QFM as two parallel *Chebyshev-tower* QFM's that apply a Y rotation gate, each spanning 4 qubits. These encoding gates are sandwiched between two layers of variational blocks: a pair of parallel 4-qubit variational blocks before, and a single 8-qubit block after the QFM. All variational blocks employ `CNOT` entangling gates interleaved with single-qubit parametrized rotations. The training was performed by a hybrid optimization scheme comprising 1200 epochs of Adam (Kingma & Ba, 2014), followed by 300 epochs of L-BFGS (Liu & Nocedal, 1989), both with a learning rate 0.05. To extract information at the end of the evolution, we employ the total magnetization operator. The overall circuit diagram is illustrated for clarity in the Appendix. The second QNN that we use is the quantum harmonic neural network of type (17) with a complex-exponential QFM, N=4 qubits, and VB's $\hat{\mathcal{U}}_{\theta_k}$ with depths 8 for both $k = \{1, 2\}$.
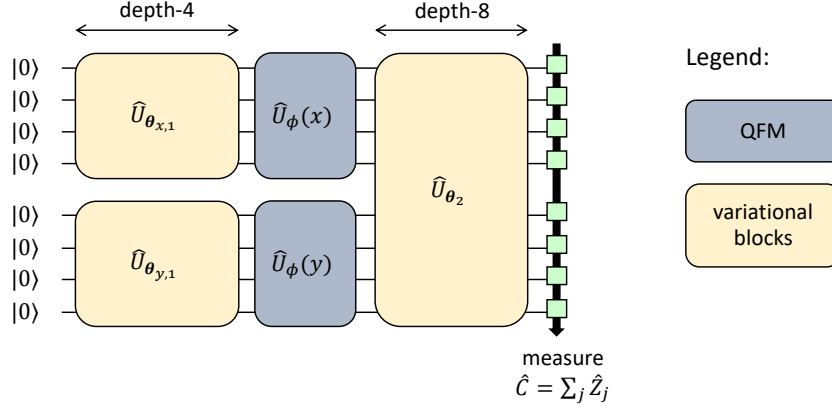
Figure 5. The overall 8-qubit QNN circuit structure, employed as described in Sect. 3.2 of the main paper. Further details on the Quantum Feature Map (QFM) and (hardware-efficient) variational blocks can be found in (Kyriienko et al., 2021).
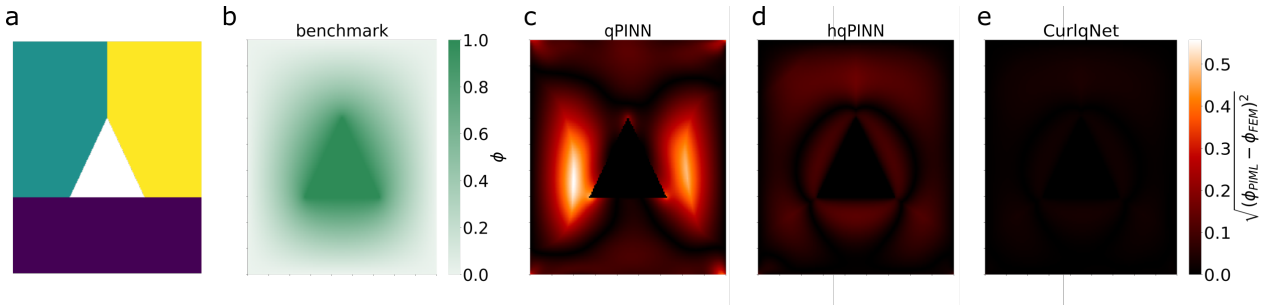

Figure 6. (a) A partitioning of the multiply-connected domain in Sect.3.5 into three simply-connected domains on which we define our multiholomorphic and XPINN networks. Note that the white triangle in the centre is not considered a part of the domain. (b) The solution $\phi(x, y)_{\text{FEM}}$ of the problem of a triangular heater positioned in a square box with Dirichlet boundary conditions, as provided by a FEM solver benchmark. (c-e) A comparison of the performance for different QNN architectures, by plotting the RSE when comparing against $\phi(x, y)_{\text{FEM}}$ the solution $\phi(x, y)_{\text{PIML}}$ provided respectively by (c) a naïve (qPINN), (d) an hqPINN and (e) a CurlqNet architecture. Additional details and definitions are provided in the main text. All plots report a sample of the reference metric on a uniform 150×150 grid. Points within the subdomain belonging to the heater have been excluded from the analysis.
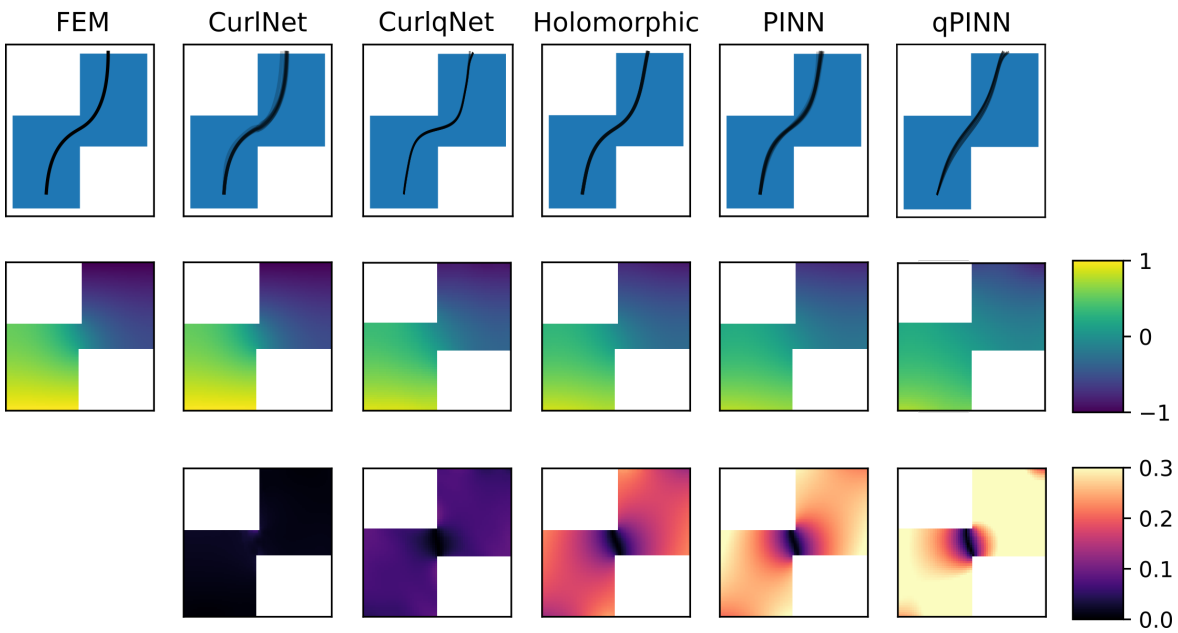
Figure 7. Figures generated from the robot navigation task in Sect.3.6 over 10 different network initialisations. First row: robot navigation paths overlapped, with visualization of the domain. Second row: the potential ϕ generated to guide the robot, as averaged over the 10 exemplary runs. Third row: the absolute error between the potential ϕ generated for each method, and the corresponding finite elements solution. First column: reference finite element solutions, second to sixth columns: results attained with corresponding neural architectures.