Let quantum neural networks choose their own frequencies

Ben Jaderberg,[1] Antonio A. Gentile,[1] Youssef Achari Berrada,[2] Elvira Shishenina,[2] and Vincent E. Elfving[1]
[1]*PASQAL, 7 rue Léonard de Vinci, 91300 Massy, France*
[2]*BMW Group, 80788 Munich, Germany*

(Received 27 September 2023; revised 25 February 2024; accepted 29 February 2024; published 22 April 2024)

Parameterized quantum circuits as machine learning models are typically well described by their representation as a partial Fourier series of the input features, with frequencies uniquely determined by the feature map's generator Hamiltonians. Ordinarily, these data-encoding generators are chosen in advance, fixing the space of functions that can be represented. In this work we consider a generalization of quantum models to include a set of trainable parameters in the generator, leading to a trainable-frequency (TF) quantum model. We numerically demonstrate how TF models can learn generators with desirable properties for solving the task at hand, including nonregularly spaced frequencies in their spectra and flexible spectral richness. Finally, we showcase the real-world effectiveness of our approach, demonstrating an improved accuracy in solving the Navier-Stokes equations using a TF model with only a single parameter added to each encoding operation. Since TF models encompass conventional fixed-frequency models, they may offer a sensible default choice for variational quantum machine learning.

DOI: 10.1103/PhysRevA.109.042421

I. INTRODUCTION

The field of quantum machine learning (QML) remains a promising application for quantum computers. In the fault-tolerant era, the prospect of quantum advantage is spearheaded by the exponential speedups in solving linear systems of equations [1], learning distributions [2,3], and topological data analysis [4]. Yet the arrival of large fault-tolerant quantum computers is not anticipated in the next decade, reducing the practical impact of such algorithms today.

One approach to solving relevant problems in machine learning with today's quantum computers is through the use of parameterized quantum circuits (PQCs) [5,6], which have been applied to a variety of use cases [7–13]. A PQC consists of quantum feature maps (FMs) $\hat{U}_F(\vec{x})$, which encode an input $\vec{x}$ into the Hilbert space, and variational ansätze $\hat{U}_A(\vec{\theta}_A)$, which contain trainable parameters. Previously, it was shown that the measured output of many variational QML models can be mathematically represented as a partial Fourier series in the network inputs [14], leading to a range of new insights [15–17]. Most strikingly, it follows that the set of frequencies Ω appearing in the Fourier series are uniquely determined by the eigenvalues of the generator Hamiltonian of the quantum FM, while the series coefficients are tuned by the variational parameters $\vec{\theta}_A$.

Conventionally, a specific generator is chosen beforehand, such that the model frequencies are fixed throughout training. In theory this is not a problem since, by choosing a generator that produces regularly spaced frequencies, the basis functions of the Fourier series form an orthogonal basis set. This ensures that asymptotically large fixed-frequency (FF) quantum models are universal function approximators [14]. Yet in reality, finite-sized quantum computers will permit models with only a finite number of frequencies. Thus, in practice, great importance should be placed on the *choice* of basis functions, for which the orthogonal convention may not be the best.

This raises an additional complexity: what is the optimal choice of basis functions? Indeed, for many problems, it is not obvious what this would be without prior knowledge of the solution. Here, we address this issue by exploring a natural extension of quantum models in which an additional set of trainable parameters $\vec{\theta}_F$ is included in the FM generator. This simple idea has a significant impact on the effectiveness of quantum models, allowing the generator eigenspectrum to change over the course of training in a direction that minimizes the objective loss. This in turn creates a quantum model with trainable frequencies as visualized in Fig. 1. In a quantum circuit learning [6] setting, we numerically demonstrate cases in which trainable-frequency (TF) models can learn a set of basis functions that better solves the task at hand, such as when the solution has a spectral decomposition with nonregularly spaced frequencies.

Furthermore, we show that an improvement is realizable for more advanced learning tasks. We train quantum models with the differentiable-quantum-circuit (DQC) algorithm [18] to learn the solution to the two-dimensional (2D) time-dependent Navier-Stokes differential equations, a family of equations that has proven challenging to solve with quantum models previously [19]. For the problem of predicting the wake flow of fluid passing a circular cylinder, a TF quantum model achieves lower loss and better predictive accuracy than the equivalent FF model. Overall, our results raise the prospect that TF quantum models could improve performance for other near-term QML problems.

II. PREVIOUS WORKS

The idea of including trainable parameters in the feature-map generator is present in some previous works. In a study of FM input redundancy, Gil Vidal and Theis [20] hypothesized that a variational input-encoding strategy may improve the

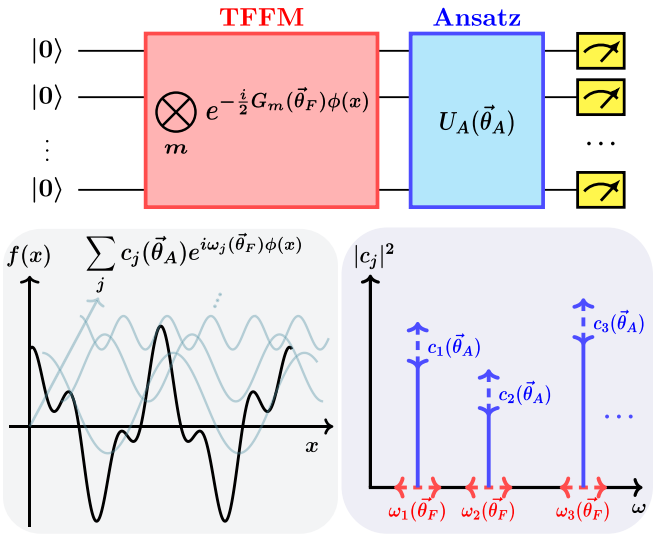

FIG. 1. An overview of the concepts discussed in this paper. Top: we introduce a parameterized quantum circuit in which the generator of the data-encoding block is a function of trainable parameters $\vec{\theta}_F$ alongside the standard trainable variational ansatz. Bottom left: the output of such a model is a Fourier-like sum over different individual modes. Bottom right: in conventional quantum models, tuning the ansatz parameters $\vec{\theta}_A$ allows the coefficients of each mode to be changed. By using a trainable-frequency feature map (TFFM), tuning $\vec{\theta}_F$ leads to a quantum model in which the frequencies of each mode can also be trained.

expressiveness of quantum models, followed by limited experiments [21]. Other works also suggest that encoding unitaries with trainable weights can reduce circuit depths of quantum models, as discussed for single-qubit classifiers [22] and quantum convolutional neural networks [23]. Nevertheless, our work is different due to contributions demonstrating (1) an analysis of the effect of trainable generators on the Fourier modes of quantum models, (2) evidence of specific spectral features in data for which TF models offer an advantage over FF models, and (3) direct comparison between FF and TF models for a practically relevant learning problem.

In the language of quantum kernels [8,24,25], several recent works used the term "trainable feature map" to describe the application of unitaries with trainable parameters on data already encoded into the Hilbert space [26,27]. Such a distinction is necessary because quantum kernel models often contain no trainable parameters at all. However, the trainable parameters of these feature maps do not apply directly to the generator Hamiltonian. As discussed in Sec. III, this is intrinsically different from our scheme as it does not lead to a model with trainable frequencies. To not confuse the two schemes, here, we adopt the wording "trainable-*frequency* feature map" and "trainable-*frequency* models."

III. METHOD

Practically, quantum computing entails the application of sequential operations (e.g., laser pulses) to a physical system of qubits. Yet to understand how these systems can be theoretically manipulated, it is often useful to work at the higher-level framework of linear algebra, from which insights can be translated back to real hardware. This allows studying strategies encompassing digital, analog, and digital-analog paradigms [28]. In a more abstract formulation, a broad class of FMs can be described mathematically as the tensor product of an arbitrary number of subfeature maps, each represented by the time evolution of a generator Hamiltonian applied to an arbitrary subset of the qubits,

$$\hat{U}_F(\vec{x}) = \bigotimes_m e^{-\frac{i}{2}\hat{G}_m(\gamma_m)\phi(\vec{x})}, \tag{1}$$

where for the subfeature map m, $\hat{G}_m(\gamma_m)$ is the generator Hamiltonian that depends on nontrainable parameters γ_m. Furthermore, $\phi(\vec{x}) : \mathbb{R}^n \to \mathbb{R}^n$ is an encoding function that depends on the input features $\vec{x}$. Practically speaking, γ_m is typically related to the index of the tensor-product space the subfeature map is applied to and can be used to set the number of unique frequencies the model has access to. Furthermore, in some cases $\hat{U}_F(\vec{x})$ can be applied several times across the quantum circuit, interleaved with variational ansatz $\hat{U}_A(\vec{\theta}_A)$ layers, for example, in data reuploading [22,29] and serial feature maps [14].

The measured output of a quantum model with a feature map defined in Eq. (1) can be expressed as a Fourier-type sum in the input dimensions

$$f(\vec{x}, \vec{\theta}_A) = \sum_{\vec{\omega}_j \in \Omega} \vec{c}_j(\vec{\theta}_A) e^{i\vec{\omega}_j \cdot \phi(\vec{x})}, \tag{2}$$

where $\vec{c}_j$ are the coefficients of the multidimensional Fourier mode with frequencies $\vec{\omega}_j$. Crucially, the frequency spectrum Ω of a model is uniquely determined by the eigenvalues of $\hat{G}_m$ [14]. More specifically, let us define the final state produced by the PQC as $|\psi_f\rangle$. If the model is a quantum kernel, where the output derives from the distance to a reference quantum state $|\psi\rangle$ [e.g., $f(\vec{x}, \vec{\theta}_A) = |\langle\psi|\psi_f\rangle|^2$], then Ω is explicitly the set of eigenvalues of the composite generator $\hat{G}$ such that $\hat{U}_F(\vec{x}) = e^{-\frac{i}{2}\hat{G}\phi(\vec{x})}$. When the generators $\hat{G}_m$ commute, the composite generator is simply $\hat{G} = \sum_m G_m$. If the model is a quantum neural network (QNN), in which the output is derived from the expectation value of a cost operator $\hat{C}$ [e.g., $f(\vec{x}, \vec{\theta}_A) = \langle\psi_f|\hat{C}|\psi_f\rangle$], then Ω contains the gaps in the eigenspectrum of $\hat{G}$.

The key insight here is that in such quantum models, a specific feature-map generator is chosen in advance. This fixes the frequency spectrum over the course of training, setting predetermined basis functions $e^{i\vec{\omega}_j \cdot \phi(\vec{x})}$ from which the model can construct a solution. For these FF models, only the coefficients $\vec{c}_j$ can be tuned by the variational ansatz during training.

In this work, we replace the FM generator with one that includes trainable parameters $\hat{G}_m(\gamma_m, \vec{\theta}_F)$. In doing so, the generator eigenspectrum, and thus the model frequencies, can also be tuned over the course of training. For this reason we refer to such feature maps as trainable-frequency feature maps (TFFMs), which in turn create TF quantum models. The output of a TF quantum model will be a Fourier-type sum in which the frequencies of each mode depend explicitly on the parametrization of the feature map. For example, for a QNN

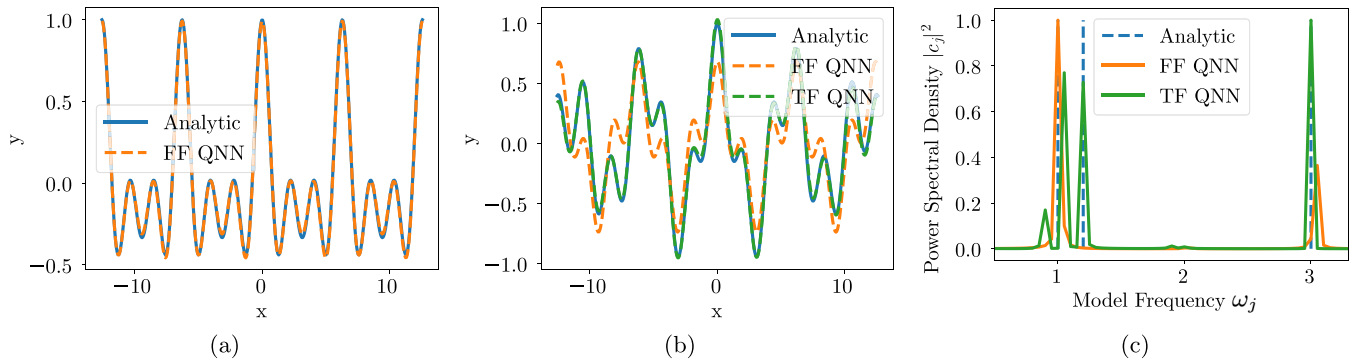

FIG. 2. Fitting cosine series of different frequencies using fixed-frequency (FF) and trainable-frequency (TF) QNNs. (a) Prediction after training on data with $\Omega_d = \{1, 2, 3\}$. (b) Prediction after training on data with $\Omega_d = \{1, 1.2, 3\}$. (c) Spectra of trained models in (b) as obtained using a discrete Fourier transform. The blue dashed lines indicate frequencies of the data.

the output of the model can be written as

$$f(\vec{x}, \vec{\theta}_A, \vec{\theta}_F) = \sum_{\vec{\omega}_j \in \Omega} \vec{c}_j(\vec{\theta}_A, \hat{C}) e^{i\vec{\omega}_j(\vec{\theta}_F)\cdot\phi(\vec{x})}. \tag{3}$$

Moreover, as $\vec{\theta}_F$ is optimized with respect to minimizing a loss function $\mathcal{L}$, the introduction of trainable frequencies $\vec{\omega}_j(\vec{\theta}_F)$ ideally allows the selection of spectral modes that better fit the specific learning task. For gradient-based optimizers, the derivative of parameters in the generator $\frac{\partial \mathcal{L}}{\partial \vec{\theta}_F}$ can be calculated as laid out in Appendix A, which for the generators used in our experiments simplifies to the parameter-shift rule (PSR) [17,30].

We note that the idea here is fundamentally different from simply viewing a combination of feature maps and variational ansätze (e.g., serial feature maps) as a higher level abstraction containing one unitary $\hat{U}_{\tilde{F}}(\vec{x}, \vec{\theta}) = \hat{U}_F(\vec{x})\hat{U}_A(\vec{\theta})$. The key concept in TF models is that a parametrization is introduced that acts directly on the generator Hamiltonian in the exponent of Eq. (1). This is what allows a trainable eigenvalue distribution, leading to quantum models with trainable frequencies.

Furthermore, in this work the FF models we consider are those in which the model frequencies are regularly spaced (i.e., integers or integer-valued multiples of a base frequency), such that the basis functions form an orthogonal set. This has become the conventional choice in the literature [12,31–36], owing to the theoretical grounding that such models are universal function approximators in the asymptotic limit [14]. However, it should be made clear that a FF model could mimic any TF model if the nontrainable unitaries in the FM were constructed with values corresponding to the final trained values of $\vec{\theta}_F$. The crux, however, is that having knowledge of such values without going through the training process is highly unlikely and might occur only where considerable *a priori* knowledge of the solution is available.

IV. PROOF OF PRINCIPLE RESULTS

The potential advantage of TF quantum models stems their ability to be trained such that their frequency spectra contain nonuniform gaps, producing nonorthogonal basis functions. We demonstrate this effect by first considering a fixed-frequency feature map (FFFM) in which, for simplicity, we restrict $\hat{G}_m$ to single-qubit operators. Overall, we choose a generator Hamiltonian $\hat{G} = \sum_{m=1}^{N} \gamma_m \hat{Y}^m/2$, where N is the number of qubits, $\hat{Y}^m$ is the Pauli matrix applied to the tensor-product space of qubit m, $\gamma_m = 1$, and $\phi(\vec{x}) = x$ is a one-dimensional Fourier encoding function. This is the commonly used angle-encoding FM [37], which we use to train a QNN to fit data produced by a cosine series,

$$y(x) = \frac{1}{|\Omega_d|} \sum_{\omega_d \in \Omega_D} \cos(\omega_d x). \tag{4}$$

Here, the data function contains a set of frequencies Ω_d, from which n_d data points are generated equally spaced in the domain $\mathcal{D} = [-4\pi, 4\pi]$. The value of n_d is determined by the Nyquist sampling rate such that $n_d = \lceil 2|\mathcal{D}| \max(\Omega_d) \rceil$.

Figure 2 demonstrates where FF models succeed and fail. In these experiments, a small QNN with $N = 3$ qubits and $L = 4$ variational ansatz layers is used. More details of the quantum models and training hyperparameters for all experiments can be found in Appendix B.

The FF QNN defined above is first trained on data with frequencies $\Omega_d = \{1, 2, 3\}$. After training, the prediction of the model is recorded as shown in Fig. 2(a). Here, we see that the underlying function can be perfectly learned. To understand why, we note that the set of degenerate eigenvalues of $\hat{G}$ are $\lambda = \{-\frac{3}{2}, -\frac{1}{2}, \frac{1}{2}, \frac{3}{2}\}$ and thus the unique gaps are $\Delta = \{1, 2, 3\}$. In this case, the natural frequencies of the model Δ are equal to the frequencies of the data Ω_d, making learning trivial. Furthermore, we find that similar excellent fits are possible for data containing frequencies that differ from the natural model frequencies by a constant factor (e.g., $\Omega_d = \{1.5, 3, 4.5\}$), provided the quantum model is given the trivial classical resource of parameters that can globally scale the input x.

By contrast, Fig. 2(b) illustrates how a FF QNN fails to fit data with frequencies $\Omega_d = \{1, 1.2, 3\}$. This occurs because no global scaling of the data can enable the fixed generator eigenspectrum to contain gaps with unequal spacing. In such a setting, no additional training would lead to accurate fitting of the data. Furthermore, no practical number of extra ansatz layers would enable the FF model to fit the data (see Sec. VI for further discussion), which we verified up to $L = 128$.

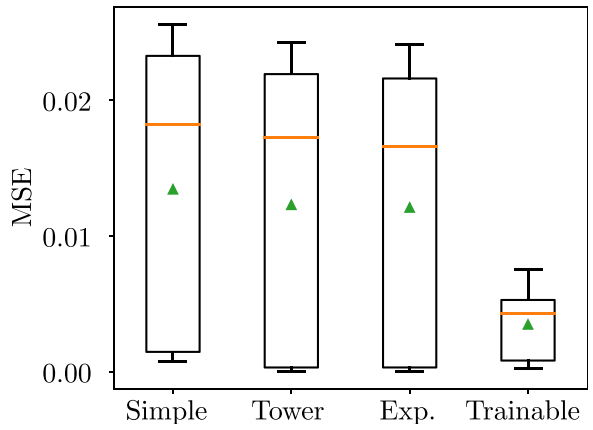

FIG. 3. Prediction MSE of simple, tower, and exponential FFFMs compared to a TFFM when training on multiple datasets with equally spaced frequencies $\vec{\omega}_d \in [1, 3]$. For each box, the triangle and orange line denote the mean and median, respectively.

Conversely, a QNN containing a TFFM with the simple parametrization $\hat{G}_\theta = \sum_{m=1}^{N} \theta_m \hat{Y}^m/2$ does significantly better in fitting the data. This is precisely possible because training of the generator parameters converges on the values $\vec{\theta}_F = \{0.89, 1.05, 1.04\}$, leading to an eigenvalue spectrum which contains gaps $\Delta_\theta = \{\ldots, 0.95, 1.050, 1.200, 3.000, \ldots\}$, as shown in Fig. 2(c). In this experiment and all others using TF models, the TFFM parameters are initialized as the unit vector $\vec{\theta}_F = \mathbb{1}$.

A further advantage of TF models is their flexible spectral richness. For FF models using the previously defined $\hat{G}$, one can pick values $\gamma_m = 1$, $\gamma_m = m$, and $\gamma_m = 2^{(m-1)}$ to produce generators in which the number of unique spectral gaps $|\Omega|$ scales as $O(N)$, $O(N^2)$, and $O(2^N)$, respectively. Typically, a practitioner may need to try all of these so-called simple, tower [18], and exponential [13] FMs, yet a TFFM can be trained to effectively represent any of these. To test this, we again sample from Eq. (4) to construct seven datasets that contain between one and seven frequencies equally spaced in the range $\omega_d \in [1, 3]$.

Figure 3 shows the mean-square error (MSE) achieved by each QNN across these datasets. In the best case, the TF and FF models perform equally well since each fixed generator produces a specific number of frequencies for which it is well suited. However, we find that a TF model outperforms FF models in the average and worst-case scenarios. Despite the data having orthogonal basis functions, the FF models have either too few frequencies (e.g., simple FM for data with $|\Omega_d| > 3$) or too many (e.g., exponential FM for data with $|\Omega_d| < 7$) to perform well across all datasets. Thus, we find that even when the optimal basis functions are orthogonal, TF models can be useful when there is no knowledge of the ideal number of spectral modes of the solution.

V. APPLICATION TO FLUID DYNAMICS

In this section we demonstrate the impact of TF models on solving problems of practical interest. Specifically, we focus on the DQC algorithm [18], in which a quantum model is trained to find a solution to a partial differential equation (PDE). The PDE to be solved is the incompressible 2D time-dependent Navier-Stokes equations (NSEs), defined as

$$\zeta_x(x, y, t) = \frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} - \frac{1}{\mathrm{Re}}\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right) + \frac{\partial p}{\partial x} = 0, \tag{5}$$

$$\zeta_y(x, y, t) = \frac{\partial v}{\partial t} + u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y} - \frac{1}{\mathrm{Re}}\left(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2}\right) + \frac{\partial p}{\partial y} = 0, \tag{6}$$

where u and v are the velocity components in the x and y directions, respectively, p is the pressure, and Re is the Reynolds number, which represents the ratio of inertial forces to viscous forces in the fluid.

The goal of this experiment is to train a quantum model to solve the downstream wake flow of fluid moving past a circular cylinder. This is one of the canonical systems of study in physics-informed neural networks, the classical analog of DQCs, due to the vortex shedding patterns and other complex dynamics exhibited even in the laminar regime [38–40]. A high-resolution dataset for $\mathrm{Re} = 100$ is obtained from [38] for the region $x = 1$ to $x = 8$ downstream from a cylinder at $x = 0$, solved using the NEKTAR high-order-polynomial finite-element method (FEM) [41,42].

The quantum model is trained by minimizing a loss $\mathcal{L} = \mathcal{L}_{\mathrm{PDE}} + \mathcal{L}_{\mathrm{data}}$. The equation loss $\mathcal{L}_{\mathrm{PDE}}$ is given by

$$\mathcal{L}_{\mathrm{PDE}} = \frac{1}{M}\sum_{i=1}^{M} \zeta_x(x_i, y_i, t_i)^2 + \zeta_y(x_i, y_i, t_i)^2, \tag{7}$$

where x_i, y_i, and t_i are the coordinates of a collocation point i in a total batch of M collocation points. Importantly, here, the terms ζ_x and ζ_y are evaluated with observables $\tilde{u}$, $\tilde{v}$, and $\tilde{p}$ predicted by the quantum model. Meanwhile, $\mathcal{L}_{\mathrm{data}}$ is a supervised loss term,

$$\mathcal{L}_{\mathrm{data}} = \frac{1}{M}\sum_{i=1}^{M} [u(x_i, y_i, t_i) - \tilde{u}(x_i, y_i, t_i)]^2 + [v(x_i, y_i, t_i) - \tilde{v}(x_i, y_i, t_i)]^2 + [p(x_i, y_i, t_i) - \tilde{p}(x_i, y_i, t_i)]^2, \tag{8}$$

where the reference values u, v, and p are given by the dataset. Notably, the dataset used in training contains only 1% of the total points in the reference solution. This means that the remainder of the flow must be predicted by learning a solution that directly solves the NSEs.

Given the increased problem complexity compared to Sec. IV, here, we employ a more advanced quantum architecture. The overall quantum model consists of two QNNs. The output of the first QNN is the predicted pressure $\tilde{p}$. Meanwhile, the output of the second QNN is the predicted stream function $\tilde{\psi}$, a quantity from which the predicted velocities $\tilde{u}$ and $\tilde{v}$ can be obtained via the relations

$$\tilde{u} = \frac{\partial \tilde{\psi}}{\partial y}, \quad \tilde{v} = -\frac{\partial \tilde{\psi}}{\partial x}.$$

Computing the velocities this way ensures that the mass continuity equation is automatically satisfied, which would

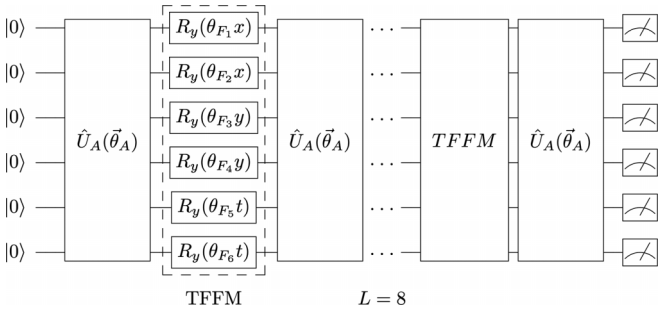

FIG. 4. Circuit diagram of the TF QNN architecture used in experiments solving the Navier-Stokes equations. The TFFM (dashed box) contains the generator parameters θ_F that allow training of the underlying model frequencies. The TFFM is followed by $L = 8$ ansatz layers, data reuploading, and then a final ansatz layer before the qubits are measured. Here, we study both digital and digital-analog versions of such layered abstraction.

otherwise require a third term in the loss function. Furthermore, the derivatives of $\tilde{p}$ and the derivatives of $\tilde{u}$ and $\tilde{v}$ required in Eq. (7) can be computed from derivative quantum circuits of the first and second QNNs, respectively, as defined in the DQC algorithm [18].

The architecture of each QNN, consisting of $N = 6$ qubits, is shown in Fig. 4. First, a single ansatz layer $\hat{U}_A(\vec{\theta}_A)$ is applied, followed by a TFFM which encodes each dimension (x, y, t) in parallel blocks. Each dimension of the TF encoding once again uses the simple parametrization $\hat{G}_\theta = \sum_{m=1}^{N} \theta_m \hat{Y}^m/2$, while the FF model uses the same generator without trainable parameters $\hat{G} = \sum_{m=1}^{N} \hat{Y}^m/2$. After the FM, a sequence of $L = 8$ ansatz layers is then applied. Subsequently a data-reuploading feature map is applied, which is a copy of the TFFM block, including sharing the parameters. Finally, the QNN architecture ends with a single ansatz layer. Overall, each QNN has a circuit depth of 52 and 180 trainable ansatz parameters.

Figure 5 gives a visualization of the results of this experiment, where the pressure field at a specific time is compared for different methods. Here, the quantum models are trained for 5000 iterations; more details can be found in Appendix B. The left panel shows the reference solution for the pressure p at time $t = 3.5$. Here, the 10×5 grid of cells with red borders corresponds to the points that are given to the quantum models at each time step to construct $\mathcal{L}_{\text{data}}$, overall 1% of the total 100×50 grid. The middle panel illustrates the prediction of the TF QNN. While the model does not achieve perfect agreement with the reference solution, it captures important qualitative features, including the formation of a large negative-pressure bubble on the left and two additional separated bubbles on the right. By contrast, the FF QNN solution correctly predicts only the global background, unable to resolve the distinct different regions of pressure that form as fluid passes to the right. This demonstrates just how impactful TFFMs can be on the expressiveness of quantum models, even with limited width and depth. Further still, we show in Appendix C how deeper TF models can match the FEM solution, whereas FF models cannot.

To quantify this benefit, the mean absolute error relative to the median (MAERM) $\frac{1}{\mathcal{N}} \sum_{i=1}^{\mathcal{N}} |\frac{\hat{y}_i - y_i}{\tilde{y}}|$ is calculated for each time step and observable, where the sum spans the $\mathcal{N}$ spatial grid points. The results, presented in Table I, numerically demonstrate the improved performance of the TF model across all observables and time steps. Particularly notable is the large improved accuracy of the vertical velocity v.

Finally, it is worth considering how the inclusion of additional trainable parameters in the feature map of the TF model affects the cost of training compared to FF models which have otherwise the same architecture. When training using a gradient-based optimizer such as Adam, regardless of the feature map chosen, each training iteration requires the calculation of $\partial L/\partial\theta_i$ for all trainable parameters θ_i in the circuit. If one were to calculate these gradients on real quantum hardware, one would need to use the PSR. For the quantum architecture used in this section, all parameterized unitaries decompose into single-qubit Pauli rotations, such that only two circuit evaluations are required to compute the gradient of each parameter in $\vec{\theta}_A$ and $\vec{\theta}_F$. Thus, the factor $C_f = \frac{|\vec{\theta}_F| + |\vec{\theta}_A|}{|\vec{\theta}_A|}$ describes the additional circuit evaluations required to train the TF model due to the additional parameters appearing in

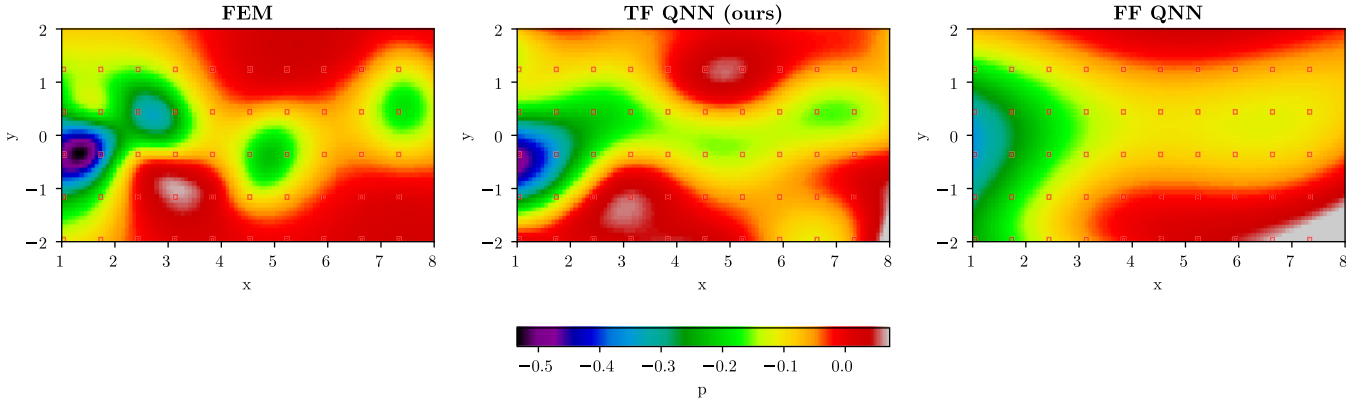

FIG. 5. Pressure field at $t = 3.5$ of the wake flow of fluid passing a circular cylinder at $x = 0$. Left: reference solution obtained with the finite-element method (FEM) in [38]. Cells with red borders indicate the training data accessible to the quantum models (see text). Middle: prediction of TF QNN using generators $\hat{G}_\theta$. Right: prediction of FF QNN using generators $\hat{G}$. The quantum circuits are based on a sliced digital-analog approach [28] suitable for platforms such as neutral atom quantum computers.

TABLE I. MAERM error predicting u, v, and p averaged over 10 runs. The minimum, maximum, and mean are with respect to the 11 time points between $t = 0$ and $t = 5.5$. For each observable, the lowest error per row is highlighted in bold.

	u		v		p	
	FF	**TF**	FF	**TF**	FF	**TF**
Min	16.8 ± 0.3	$\mathbf{12.8 \pm 1.9}$	90.3 ± 2.6	$\mathbf{27.4 \pm 3.3}$	62.9 ± 1.9	$\mathbf{51.0 \pm 6.0}$
Max	18.3 ± 0.4	$\mathbf{14.2 \pm 3.0}$	122.0 ± 2.4	$\mathbf{34.3 \pm 4.6}$	78.3 ± 1.8	$\mathbf{62.8 \pm 10.3}$
Mean	17.5 ± 0.3	$\mathbf{13.2 \pm 2.0}$	103.2 ± 1.4	$\mathbf{30.4 \pm 2.7}$	73.2 ± 1.9	$\mathbf{57.0 \pm 3.7}$

the corresponding feature map. For the models used in this section, the cost factor is $C_f = \frac{12+180}{180} = 1.07$. This represents only a 7% increase in the number of quantum circuits evaluated, a modest cost for the improvement in performance observed.

VI. DISCUSSION

In this work we explore an extension of variational QML models to include trainable parameters in the data-encoding Hamiltonian, compatible with a wide range of models, including those based on digital and digital-analog paradigms. As introduced in Sec. III, when viewed through the lens of a Fourier representation, the effect of such parameters is fundamentally different from those in the variational ansatz, as they enable the frequencies of the quantum model to be trained. Furthermore, in Sec. IV we showed how this leads to quantum models with specific spectral properties inaccessible to the conventional approach of tuning only the coefficients of fixed orthogonal basis functions. Finally, in Sec. V we demonstrated the benefit of TF models for practical learning problems, leading to a learned solution of the Navier-Stokes equations closer to the ground truth than FF models.

We note that, in theory, a FF model could also achieve parity with TF models if it had independent control of the coefficients of each basis function. Given data with a minimum frequency gap $\Delta_{d,\min}$ and spanning a range $r_d = |\Omega_{d,\max} - \Omega_{d,\min}|$, a quantum model with $\frac{r_d}{\Delta_{d,\min}}$ fixed frequencies could span all modes of the data. In this case, such a model could even represent data with nonregularly spaced frequencies (e.g., $\Omega_d = \{1, 1.2, 3\}$ by setting the coefficients $\vec{c}_j = 0$ for $\vec{\omega}_j = \{1.1, 1.3, 1.4, \ldots, 2.9\}$). However, such a model would be exponentially costly to train since independent control of the coefficients of the model frequencies would generally require $O(2^N)$ ansatz parameters. It is for this reason that we present TF models as having a practical advantage within the context of scalable approaches to quantum machine learning.

Looking forward, an interesting open question remains around the performance of other parametrizations of the generator. A notable instance of this would be $\hat{G} = \sum_{m=1}^{N} \mathrm{NN}(\theta_F)\hat{Y}^m/2$, where the parametrization is set by a classical neural network. Interestingly, this has already been implemented in a different context in so-called hybrid quantum-classical networks [43–46], including studies of DQCs [19]. The use of hybrid networks is typically motivated by the desire to relieve the computational burden from today's small-scale quantum models. Our work offers the insight that such architectures are actually using a classical neural network to set the frequencies of the quantum model. Promisingly, there is already early evidence to suggest that this scheme may lead to improved performance for classification [47].

Compellingly, for many different parametrizations of TFFMs, a generator with regularly spaced eigenvalues is accessible within the parameter space. This is particularly true for the parametrizations studied in this work, which can be trivially realized as an orthogonal model when $\vec{\theta}_F = \mathbb{1}$. This implies that, at worst, many classes of TF models can fall back to the behavior of standard FF models. We find this, along with our results, a strong reason to explore in the future whether TF models could be an effective choice as a new default for variational quantum models.

APPENDIX A: COMPUTING DERIVATIVES OF TRAINABLE PARAMETERS IN THE GENERATOR

Training a variational quantum circuit often involves performing gradient-based optimization against the trainable parameters of the ansatz. In this section we make clear how, in the case of TF models, the trainable parameters in the generator can also be optimized in the same way. For gradient-based optimization, one needs to compute $\frac{\partial \mathcal{L}}{\partial \vec{\theta}}$ for a suitably defined loss function $\mathcal{L}$ which captures the adherence of the solution $f(x)$ to the conditions set by the training problem.

Let us first define the state produced by a TF quantum model as

$$\left|f_{\vec{\theta}_F,\vec{\theta}_A}(x)\right\rangle = \hat{U}_A(\vec{\theta}_A)\hat{U}_F(x,\vec{\theta}_F)|0\rangle. \tag{A1}$$

The output of the quantum models used in the main text, given as the expectation value of a cost operator $\hat{C}$,

$$f(x,\vec{\theta}_F,\vec{\theta}_A) = \left\langle f_{\vec{\theta}_F,\vec{\theta}_A}(x)\middle|\hat{C}\middle|f_{\vec{\theta}_F,\vec{\theta}_A}(x)\right\rangle, \tag{A2}$$

can then act as a surrogate for the target function f.

For a supervised-learning (SVL) loss contribution we can define

$$\mathcal{L}_{\mathrm{SVL}} = \frac{1}{M}\sum_{i}^{M} L(f(x_i,\vec{\theta}_F,\vec{\theta}_A), y_i), \tag{A3}$$

where L is a suitable distance function. In a DQC setting, with each (partial, differential) equation embedded in a functional $F[\partial_X f(x), f(x), x]$ to be estimated on a set of M collocation points $\{x_i\}$, one can define a physics-informed loss function as

$$\mathcal{L}_{\mathrm{DQC}} = \frac{1}{M}\sum_{i}^{M} L(F[\partial_X f(x_i), f(x_i), x_i], 0). \tag{A4}$$

When optimizing the loss against a certain variational parameter θ, if $\theta \in \vec{\theta}_A$ is an ansatz parameter, then $\frac{\partial \mathcal{L}}{\partial \theta}$ can be computed as standard with the PSR and generalized

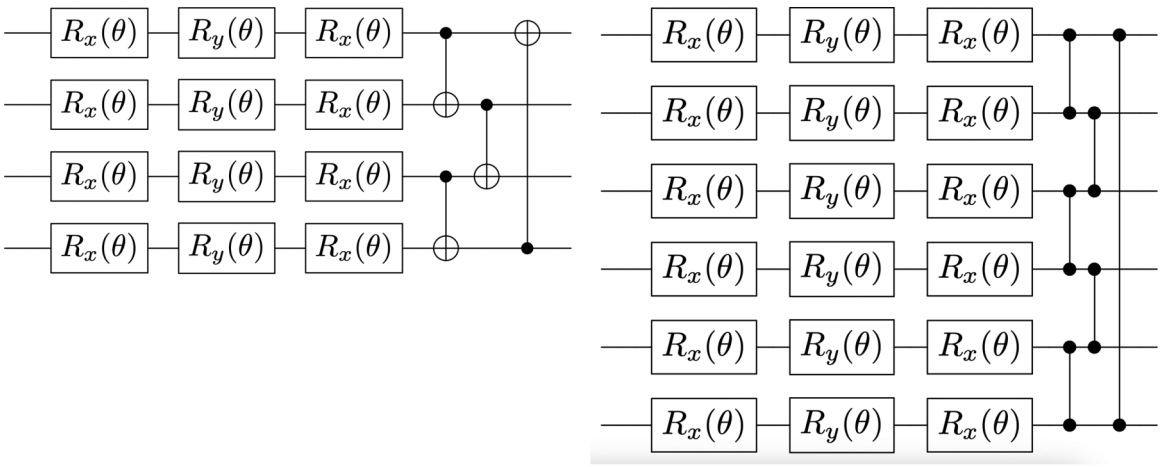

FIG. 6. A single layer of the different types of hardware-efficient ansätze used in this work. All parameters θ are separate and independent.

parameter-shift rules (GPSR) [17,18]. If, instead, $\theta \in \vec{\theta}_F$, we can account for the inclusion of the feature x using the linearity of differentiation

$$\frac{\partial \mathcal{L}_{\text{SVL}}}{\partial \theta} = \frac{1}{M}\sum_i^M \frac{dL}{df}\frac{\partial f(x_i)}{\partial \theta}, \tag{A5}$$

$$\frac{\partial \mathcal{L}_{\text{DQC}}}{\partial \theta} = \frac{1}{M}\sum_i^M \frac{\partial L}{\partial \theta}(F[\partial_X f(x_i), f(x_i), x_i], 0). \tag{A6}$$

Note that in Eq. (A6) we leave the right-hand side (RHS) as its implicit form because it depends not only upon the actual choice of the distance L, as in Eq. (A5), but also upon the terms involved in the functional F. In order to give further guidance on the explicit treatment of this latter case, we can split the discussion according to the various terms involved in the functional F under the likely assumption that the problem variables X are independent of the variational parameter θ:

(1) Terms that depend solely on x_i have null $\frac{\partial \cdot}{\partial \theta}$ and can be neglected.

(2) Terms containing the function f itself can be addressed via the chain rule already elicited in Eq. (A5).

(3) Finally, terms depending on $\partial_X f(x_i)$ can be similarly decomposed as

$$\frac{d^2L}{df\,dX}\frac{\partial^2 f(x_i)}{\partial\theta\,\partial X} = \frac{dL}{d(\partial_X f)}\frac{\partial}{\partial X}\frac{\partial f(x_i)}{\partial\theta}, \tag{A7}$$

where the latter descends from the independence highlighted above and the first term on the RHS can be simply attained from the (known) analytical form of L and F.

Thus, following the chain rule, in all cases we obtain a dependence upon the term $\partial f(x_i)/\partial\theta$.

In terms of computing $\partial f(x_i)/\partial\theta$, we again omit the case where $\theta \in \vec{\theta}_A$ because it is known from the literature [6,17]. For the specific case of $\theta \in \vec{\theta}_F$ instead, let us first combine the unitary ansatz and the cost operator as $\hat{U}_A^\dagger(\vec{\theta}_A)\hat{C}\hat{U}_A(\vec{\theta}_A) \equiv \hat{C}_A(\vec{\theta}_A)$ for brevity. Rewriting Eq. (A2), using the generic FM provided in Eq. (1) with a trainable generator $\hat{G}_m(\vec{\gamma}, \vec{\theta}_F)$ and isolating the only term dependent on the $\theta_{\tilde{m}}$ of interest, we get

$$\hat{U}_F(\vec{x}, \vec{\gamma}, \vec{\theta}_F) = e^{-i\hat{G}_1(\vec{\gamma},\theta_1)\phi_1(\vec{x})} \otimes \cdots \otimes e^{-i\hat{G}_{\tilde{m}}(\vec{\gamma},\theta_{\tilde{m}})\phi_{\tilde{m}}(\vec{x})} \otimes \cdots \otimes e^{-i\hat{G}_M(\vec{\gamma},\theta_M)\phi_M(\vec{x})} \tag{A8}$$

$$\equiv \hat{U}_{FL} \otimes e^{-i\hat{G}_{\tilde{m}}(\vec{\gamma},\theta_{\tilde{m}})\phi_{\tilde{m}}(\vec{x})} \otimes \hat{U}_{FR}. \tag{A9}$$

Further simplifying the notation using $\hat{U}_{FR}|0\rangle \equiv |f_{FR}\rangle$ and $\hat{U}_{FL}^\dagger \hat{C}_A \hat{U}_{FL} \equiv \hat{C}_{AF}$ produces

$$f(x, \theta_F, \theta_A) = \langle f_{FR}|e^{i\hat{G}_{\tilde{m}}(\vec{\gamma},\theta_{\tilde{m}})\phi_{\tilde{m}}(\vec{x})}\hat{C}_{AF}\,e^{-i\hat{G}_{\tilde{m}}(\vec{\gamma},\theta_{\tilde{m}})\phi_{\tilde{m}}(\vec{x})}|f_{FR}\rangle. \tag{A10}$$

With the model expressed in terms of the dependence on the single FM parameter $\theta_{\tilde{m}}$, we can now address computing $\partial f/\partial\theta_{\tilde{m}}$:

$$\frac{\partial f}{\partial \theta_{\tilde{m}}} = \langle f_{FR}|e^{i\hat{G}_{\tilde{m}}\phi_{\tilde{m}}}\left[\phi_{\tilde{m}}\frac{\partial \hat{G}_{\tilde{m}}}{\partial\theta_{\tilde{m}}}, \hat{C}_{AF}\right]e^{-i\hat{G}_{\tilde{m}}\phi_{\tilde{m}}}|f_{FR}\rangle, \tag{A11}$$

where, for simplicity, we have omitted the parameter dependences of $\phi_{\tilde{m}}(\vec{x})$ and the $\hat{G}_{\tilde{m}}(\vec{\gamma}, \theta_{\tilde{m}})$ generator and we have introduced the commutator notation $[\cdot, \cdot]$.

Observing Eq. (A11), we are thus left with obtaining the partial derivative $\partial\hat{G}_{\tilde{m}}/\partial\theta_{\tilde{m}}$, and to that extent, we distinguish three cases. (1) If $\hat{G}_{\tilde{m}}(\vec{\gamma}, \theta_{\tilde{m}}) = \gamma\theta_{\tilde{m}}\hat{\sigma}^\alpha$, i.e., a single Pauli operator for a chosen α axis, then we can obtain the target derivative using the standard PSR. (2) When a similar dependence on the (non)trainable parameters holds but we generalize beyond involutory and idempotent primitives, i.e., $\hat{G}_{\tilde{m}}(\vec{\gamma}, \theta_{\tilde{m}}) = \gamma\theta_{\tilde{m}}\hat{G}_{\tilde{m}}$, we can instead rely on a single application of the GPSR to obtain $\partial\hat{G}_{\tilde{m}}/\partial\theta_{\tilde{m}}$. (3) Finally, in the most generic case considered in this work, the spectral gaps of $\hat{G}_{\tilde{m}}$ might depend nontrivially upon $\theta_{\tilde{m}}$. In this last case, one should recompute such gaps for each new trained $\theta_{\tilde{m}}$ in order to apply GPSR. Note, however, that one could always decompose $\hat{G}_{\tilde{m}}(\vec{\gamma}, \theta_{\tilde{m}}) = \sum_i^I \hat{P}_i(\vec{\gamma}, \theta_{\tilde{m}})$, i.e., a sum of Pauli strings $\hat{P}$. With this decomposition approach, ignoring any structure in $\hat{G}_{\tilde{m}}$, at most $2I$ circuit evaluations would suffice to retrieve $\partial\hat{G}_{\tilde{m}}/\partial\theta_{\tilde{m}}$ [48].

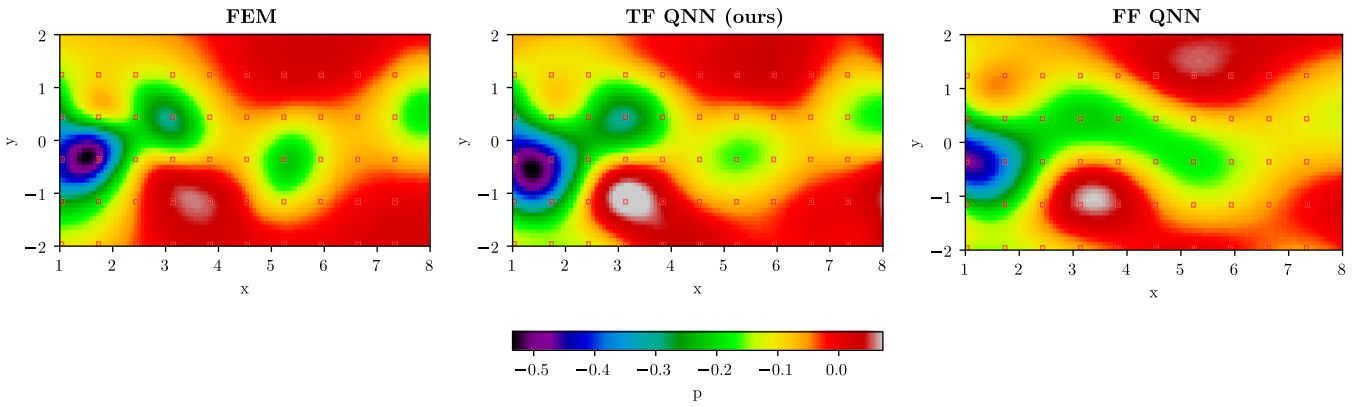

FIG. 7. Pressure field at $t = 4.0$ of the wake flow of fluid passing a circular cylinder at $x = 0$. Left: reference solution obtained by FEM. Middle: prediction of the overparameterized TF QNN model with $L = 64$. Right: prediction of the equivalent FF QNN.

APPENDIX B: QUANTUM MODELS AND TRAINING HYPERPARAMETERS

For all quantum models used in this work, the variational ansatz is a variant of the hardware-efficient ansatz (HEA) [49], with entangling unitaries connecting the qubits in a ring topology as shown in Fig. 6. Here, we study both digital and sliced digital-analog quantum computing (sDAQC) [28] versions of the HEA, as shown in Figs. 6(a) and 6(b), respectively. In the sDAQC approach, the entangling operations are the fixed-duration Hamiltonian evolution of the form $\exp(i\hat{n}_k\hat{n}_l\pi)$ between neighboring pairs of qubits (k, l). After all FMs and ansatz layers are applied, a final state $|\psi\rangle$ is produced. The output [e.g., the predicted value of $y(x)$ or $\psi(x, y, t)$ or $p(x, y, t)$] of all models is obtained via the expectation value $\langle\psi|\hat{C}|\psi\rangle$, where the cost operator $\hat{C} = \sum_{m=1}^{N}\hat{Z}^m$ is an equally weighted total magnetization across the N qubits. This combination of constant-depth ansatz and one-local observables is known to avoid cost-function-induced barren plateaus [50].

All models in this work were trained using the Adam optimizer.

In Fig. 2(a) the model was trained with hyperparameters: qubits $N = 4$, $L = 4$ layers of the ansatz in Fig. 6(a), training iterations $N_i = 2000$, batch size $b_s = 1$, learning rate $\eta = 10^{-3}$.

In Fig. 2(b), both models were trained with $N = 4$, $L = 4$ layers of the ansatz in Fig. 6(a), $N_i = 6000$, $b_s = 2$, and $\eta = 10^{-3}$.

In Fig. 3 all models were trained with $N = 4$, $L = 8$ layers of the ansatz in Fig. 6(a), $N_i = 4000$, $b_s = 2$, and $\eta = 10^{-3}$.

In Fig. 5 and Table I, all models were trained per QNN with $N = 6$, $L = 10$ layers of the sDAQC ansatz in Fig. 6(b), $N_i = 5000$, $b_s = 600$, and $\eta = 10^{-2}$.

In Appendix C, Fig. 7, and Table II, all models were trained per QNN with $N = 4$, $L = 64$ layers of the ansatz in Fig. 6(a), $N_i = 5000$, $b_s = 600$, and $\eta = 10^{-2}$.

APPENDIX C: NAVIER-STOKES RESULTS IN THE OVERPARAMETERIZED REGIME

In this section we repeat the experiments in Sec. V using an overparameterized model, such that the number of trainable parameters is larger than the dimension of the Hilbert space [51]. Furthermore, here, we use a serial TFFM in which each dimension (x, y, t) is encoded serially in separate blocks, separated by an ansatz layer which acts to change the encoding basis to avoid loss of information. This is theoretically preferential to the parallel encoding strategy since it produces a quantum model with more unique frequencies. The TFFM uses the simple parametrization $\hat{G}_\theta = \sum_{m=1}^{N}\theta_m\hat{Y}^m/2$ for each dimension, while the FF model uses the same generator without trainable parameters $\hat{G} = \sum_{m=1}^{N}\hat{Y}^m/2$. After the FM, the model has $L = 64$ ansatz layers bisected by a data-reuploading FM. In total, each QNN has 804 trainable ansatz parameters.

Figure 7 presents a visualization of the experiment's results, evaluating the pressure field at a specific time. In this deeper regime, the TF QNN achieves excellent agreement with the reference solution, successfully capturing more features such as the presence of two interconnected negative-pressure bubbles on the left. In contrast, despite its increased

TABLE II. MAERM error predicting u, v, and p averaged over 10 runs. The minimum, maximum, and mean are with respect to the 11 time points between $t = 0$ and $t = 5.5$. For each observable, the lowest error per row is highlighted in bold.

	u		v		p	
	FF	**TF**	FF	**TF**	FF	**TF**
Min	7.2 ± 0.5	$\mathbf{5.3 \pm 0.6}$	23.1 ± 0.8	$\mathbf{16.8 \pm 0.9}$	39.2 ± 1.9	$\mathbf{31.2 \pm 3.9}$
Max	8.5 ± 0.6	$\mathbf{8.1 \pm 1.3}$	30.0 ± 0.7	$\mathbf{22.3 \pm 1.8}$	54.2 ± 4.1	$\mathbf{47.4 \pm 5.7}$
Mean	7.7 ± 0.3	$\mathbf{6.2 \pm 0.4}$	26.0 ± 0.5	$\mathbf{18.9 \pm 0.7}$	45.2 ± 3.3	$\mathbf{36.8 \pm 2.2}$

depth, the FF QNN solution is only approximately accurate and fails to correctly identify the separation between the pressure bubbles in the middle and right panels. The numerical performance of the models is given in Table II.

[1] A. W. Harrow, A. Hassidim, and S. Lloyd, Quantum algorithm for linear systems of equations, Phys. Rev. Lett. **103**, 150502 (2009).

[2] N. Pirnay, R. Sweke, J. Eisert, and J.-P. Seifert, A superpolynomial quantum-classical separation for density modeling, Phys. Rev. A **107**, 042416 (2023).

[3] Y. Liu, S. Arunachalam, and K. Temme, A rigorous and robust quantum speed-up in supervised machine learning, Nat. Phys. **17**, 1013 (2021).

[4] S. Lloyd, S. Garnerone, and P. Zanardi, Quantum algorithms for topological and geometric analysis of data, Nat. Commun. **7**, 10138 (2016).

[5] M. Benedetti, E. Lloyd, S. Sack, and M. Fiorentini, Parameterized quantum circuits as machine learning models, Quantum Sci. Technol. **4**, 043001 (2019).

[6] K. Mitarai, M. Negoro, M. Kitagawa, and K. Fujii, Quantum circuit learning, Phys. Rev. A **98**, 032309 (2018).

[7] E. Farhi and H. Neven, Classification with quantum neural networks on near term processors, arXiv:1802.06002.

[8] V. Havlíček, A. D. Córcoles, K. Temme, A. W. Harrow, A. Kandala, J. M. Chow, and J. M. Gambetta, Supervised learning with quantum-enhanced feature spaces, Nature (London) **567**, 209 (2019).

[9] J. S. Otterbach *et al.*, Unsupervised machine learning on a hybrid quantum computer, arXiv:1712.05771.

[10] J. Bausch, Recurrent quantum neural networks, Adv. Neural Inf. Process. Syst. **33**, 1368 (2020).

[11] W. Guan, G. Perdue, A. Pesah, M. Schuld, K. Terashi, S. Vallecorsa, and J.-R. Vlimant, Quantum machine learning in high energy physics, Mach. Learn.: Sci. Technol. **2**, 011003 (2021).

[12] N. Heim, A. Ghosh, O. Kyriienko, and V. E. Elfving, Quantum model-discovery, arXiv:2111.06376.

[13] O. Kyriienko, A. E. Paine, and V. E. Elfving, Protocols for trainable and differentiable quantum generative modelling, arXiv:2202.08253.

[14] M. Schuld, R. Sweke, and J. J. Meyer, Effect of data encoding on the expressive power of variational quantum-machine-learning models, Phys. Rev. A **103**, 032430 (2021).

[15] D. Heimann, G. Schönhoff, and F. Kirchner, Learning capability of parametrized quantum circuits, arXiv:2209.10345.

[16] E. Peters and M. Schuld, Generalization despite overfitting in quantum machine learning models, Quantum **7**, 1210 (2023).

[17] O. Kyriienko and V. E. Elfving, Generalized quantum circuit differentiation rules, Phys. Rev. A **104**, 052417 (2021).

[18] O. Kyriienko, A. E. Paine, and V. E. Elfving, Solving nonlinear differential equations with differentiable quantum circuits, Phys. Rev. A **103**, 052416 (2021).

[19] A. Sedykh, M. Podapaka, A. Sagingalieva, N. Smertyak, K. Pinto, M. Pflitsch, and A. Melnikov, Quantum physics-informed neural networks for simulating computational fluid dynamics in complex shapes, arXiv:2304.11247.

[20] F. J. Gil Vidal and D. O. Theis, Input redundancy for parameterized quantum circuits, Front. Phys. **8**, 297 (2020).

[21] A. W. Lei, Comparisons of input encodings for quantum neural networks, master's thesis, University of Tartu, 2020.

[22] A. Pérez-Salinas, A. Cervera-Lierta, E. Gil-Fuster, and J. I. Latorre, Data re-uploading for a universal quantum classifier, Quantum **4**, 226 (2020).

[23] E. Ovalle-Magallanes, D. E. Alvarado-Carrillo, J. G. Avina-Cervantes, I. Cruz-Aceves, and J. Ruiz-Pinales, Quantum angle encoding with learnable rotation applied to quantum–classical convolutional neural networks, Appl. Soft Comput. **141**, 110307 (2023).

[24] L.-P. Henry, S. Thabet, C. Dalyac, and L. Henriet, Quantum evolution kernel: Machine learning on graphs with programmable arrays of qubits, Phys. Rev. A **104**, 032416 (2021).

[25] A. E. Paine, V. E. Elfving, and O. Kyriienko, Quantum kernel methods for solving regression problems and differential equations, Phys. Rev. A **107**, 032428 (2023).

[26] M. John, J. Schuhmacher, P. Barkoutsos, I. Tavernelli, and F. Tacchino, Optimizing quantum classification algorithms on classical benchmark datasets, Entropy **25**, 860 (2023).

[27] G. Gentinetta, D. Sutter, C. Zoufal, B. Fuller, and S. Woerner, Quantum kernel alignment with stochastic gradient descent, in *Proceedings of the 2023 IEEE International Conference on Quantum Computing and Engineering (QCE)* (IEEE, 2023), Vol. 1, pp. 256–262.

[28] A. Parra-Rodriguez, P. Lougovski, L. Lamata, E. Solano, and M. Sanz, Digital-analog quantum computation, Phys. Rev. A **101**, 022305 (2020).

[29] L. Fan and H. Situ, Compact data encoding for data re-uploading quantum classifier, Quantum Inf. Process. **21**, 87 (2022).

[30] G. E. Crooks, Gradients of parameterized quantum gates using the parameter-shift rule and gate decomposition, arXiv:1905.13311.

[31] J. Kübler, S. Buchholz, and B. Schölkopf, The inductive bias of quantum kernels, Adv. Neural Inf. Process. Syst. **34**, 12661 (2021).

[32] A. Ghosh, A. A. Gentile, M. Dagrada, C. Lee, S.-H. Kim, H. Cha, Y. Choi, B. Kim, J.-I. Kye, and V. E. Elfving, Harmonic (quantum) neural networks, Proc. Mach. Learn. Res. **202**, 11340 (2023).

[33] A. Skolik, S. Jerbi, and V. Dunjko, Quantum agents in the Gym: A variational quantum algorithm for deep q-learning, Quantum **6**, 720 (2022).

[34] T. Hubregtsen, D. Wierichs, E. Gil-Fuster, P.-J. H. S. Derks, P. K. Faehrmann, and J. J. Meyer, Training quantum embedding kernels on near-term quantum computers, Phys. Rev. A **106**, 042431 (2022).

[35] S. Jerbi, L. J. Fiderer, H. Poulsen Nautrup, J. M. Kübler, H. J. Briegel, and V. Dunjko, Quantum machine learning beyond kernel methods, Nat. Commun. **14**, 517 (2023).

[36] J. J. Meyer, M. Mularski, E. Gil-Fuster, A. A. Mele, F. Arzani, A. Wilms, and J. Eisert, Exploiting symmetry in variational quantum machine learning, PRX Quantum **4**, 010328 (2023).

[37] R. LaRose and B. Coyle, Robust data encodings for quantum classifiers, Phys. Rev. A **102**, 032420 (2020).

[38] M. Raissi, P. Perdikaris, and G. E. Karniadakis, Physics informed deep learning (Part II): Data-driven discovery of nonlinear partial differential equations, arXiv:1711.10566.

[39] C. Rao, H. Sun, and Y. Liu, Physics-informed deep learning for incompressible laminar flows, Theor. Appl. Mech. Lett. **10**, 207 (2020).

[40] S. Xu, Z. Sun, R. Huang, D. Guo, G. Yang, and S. Ju, A practical approach to flow field reconstruction with sparse or incomplete data through physics informed neural network, Acta Mech. Sin. **39**, 322302 (2023).

[41] C. D. Cantwell *et al.*, Nektar++: An open-source spectral/hp element framework, Comput. Phys. Commun. **192**, 205 (2015).

[42] G. Karniadakis and S. Sherwin, *Spectral/hp Element Methods for Computational Fluid Dynamics* (Oxford University Press, USA, 2013).

[43] B. Jaderberg, L. W. Anderson, W. Xie, S. Albanie, M. Kiffner, and D. Jaksch, Quantum self-supervised learning, Quantum Sci. Technol. **7**, 035005 (2022).

[44] A. Mari, T. R. Bromley, J. Izaac, M. Schuld, and N. Killoran, Transfer learning in hybrid classical-quantum neural networks, Quantum **4**, 340 (2020).

[45] J. Liu, K. H. Lim, K. L. Wood, W. Huang, C. Guo, and H.-L. Huang, Hybrid quantum-classical convolutional neural networks, Sci. China Phys. Mech. Astron. **64**, 290311 (2021).

[46] J. Liu, F. Tacchino, J. R. Glick, L. Jiang, and A. Mezzacapo, Representation learning via quantum neural tangent kernels, PRX Quantum **3**, 030323 (2022).

[47] T. Hur, I. F. Araujo, and D. K. Park, Neural quantum embedding: Pushing the limits of quantum supervised learning, arXiv:2311.11412.

[48] T. Hubregtsen, F. Wilde, S. Qasim, and J. Eisert, Single-component gradient rules for variational quantum algorithms, Quantum Sci. Technol. **7**, 035008 (2022).

[49] A. Kandala, A. Mezzacapo, K. Temme, M. Takita, M. Brink, J. M. Chow, and J. M. Gambetta, Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets, Nature (London) **549**, 242 (2017).

[50] M. Cerezo, A. Sone, T. Volkoff, L. Cincio, and P. J. Coles, Cost function dependent barren plateaus in shallow parametrized quantum circuits, Nat. Commun. **12**, 1791 (2021).

[51] X. You, S. Chakrabarti, and X. Wu, A convergence theory for over-parameterized variational quantum eigensolvers, arXiv:2205.12481.