Solving Nonlinear PDEs with Variational Quantum Computing

Mathew Pareles*
McMahon Lab, Cornell University
(Dated: March 2, 2024)

Recently, there has been increased interest on the applications of Variational Quantum Computing (VQC). Lubasch et al. created a flagship paper on solving PDEs [1] using a 2-qubit ansatz. However, there are many challenges to scaling circuits in size, and creating circuits that better represent relevant parts of Hilbert state while remaining fully accurate. Here we propose several novel Quantum circuits to be used in solving the Heat and Advection Equations, and discuss methods of scaling our approach to more advanced equations including the Inviscid Burgers' Equation.

I. INTRODUCTION

The world is in the era of noisy intermediate-scale quantum computing (NISQ) - there has been a recent a boom in the use of Variational Quantum Computing methods (VQC) to simulate classical equations. A challenge is efficiently creating low-depth circuits of the same quality as high-depth ones, especially since one must avoid making simplifications that could also be made with a digital computer [7]. We use VQC to solve PDEs, and focus on several tasks whose completion would progress our ability to produce a quantum computer speedup. 1) designing and testing low-depth and accurate ansaetze that might allow for a quantum speedup 2) determining the best finite difference method and optimizer to use to reduce the complexity of the cost function and allow for its effective optimization 3) creating low-depth initial condition operators to append to our variational ansatz and 4) the read-out of useful information from high-qubit circuits, whose valuable result cannot be simulated classically. This paper focuses on the first two of these points.

Solving PDEs using quantum computers is highly related to classical methods. We consider 1D functions $f(x,t)$ subject to PDEs of the form

$$\partial f(x,t)/\partial t = F(x)f(x,t) \tag{1}$$

where F consists of functions of x and partial derivative operators wrt x only. We approximate $f(x,t)$ discretely in space and time, with N uniformly distributed points in space with difference h, and unbounded in time with difference τ. We can use integers $n, m \geq 0$ to describe the function discretely, $f(x,t) \simeq f(x_n,t_m) = f(hn,\tau m) = f_n^m$; at the mth time, our function $\mathbf{f}^m = (f_0^m, f_1^m, ..., f_{N-1}^m)$. Finite difference methods on on Eq. (1) can be used to obtain $N \times N$ matrices T_-, T_+ on x alone which satisfy $T_-\mathbf{f}^{m+1} = T_+\mathbf{f}^m$, which we give examples for in the next section. The two sides' difference is ideally 0, and so we create a cost function which satisfies,

$$0 \approx |T_-\mathbf{f}^{m+1} - T_+\mathbf{f}^m|^2 \equiv C \tag{2}$$

By starting with a known $\mathbf{f}^m$, we can find $\mathbf{f}^{m+1}$ by minimizing the cost function C; $\mathbf{f}^{m+1} = \text{argmin}_{\mathbf{f}^{m+1}} C(\mathbf{f}^m, \mathbf{f}^{m+1})$.

In trying to use quantum computing to do these same computations, an important assumption is that one can equate the amplitudes of a quantum state with the amplitudes of the mathematical function f in Eq. (1). As a result, the quantum-mechanical analogue to Eq. (2) is straightforward, and we simply begin with

$$C(\boldsymbol{\lambda}_t, \boldsymbol{\lambda}_{t+1}) \equiv |\hat{T}_- |f(\boldsymbol{\lambda}_{t+1})\rangle - \hat{T}_+ |f(\boldsymbol{\lambda}_t)\rangle|^2 \tag{3}$$

where the $|f\rangle$ are not normalized, and depend on variational parameters $\boldsymbol{\lambda}$.

To account for the unit norm of states and motivate use of a quantum computer, we equate $|f(\boldsymbol{\lambda})\rangle = \lambda_0 |\psi(\vec{\lambda})\rangle = \lambda_0 \hat{U}(\vec{\lambda}) |\mathbf{0}\rangle$, introducing a new scaling parameter λ_0 and unitary operator $\hat{U}$ which we call the (variational) ansatz. Here you can see the power in the quantum approach is taking advantage of the quantum state $|\psi(\vec{\lambda})\rangle$, which represents $N = 2^n$ gridpoints with only n qubits. We are essentially done explaining the core assumptions here, and there are a few remaining details in its implementation.

We want to make use of a quantum computer to compute the terms in Eq. (3). To do this, we can expand out the square power, and get a sum of different quantum expectations.

$$\begin{aligned} C = \langle f(\boldsymbol{\lambda}_{t+1})| \hat{T}_-^\dagger \hat{T}_- |f(\boldsymbol{\lambda}_{t+1})\rangle + \langle f(\boldsymbol{\lambda}_t)| \hat{T}_+^\dagger \hat{T}_+ \\ |f(\boldsymbol{\lambda}_t)\rangle - 2\Re\{\langle f(\boldsymbol{\lambda}_{t+1})| \hat{T}_-^\dagger \hat{T}_+ |f(\boldsymbol{\lambda}_t)\rangle\} \end{aligned} \tag{4}$$

These can be be computed with the Hadamard test (Fig. 1c). The only trouble now is $\hat{T}_\pm$ is generally not unitary, but we can just decompose them into sums of several unitary operators, as we will demonstrate. By the linearity of expectations, these expectations can be computed independently, and this simply creates more expectations to evaluate and sum together in the cost function. Overall, the quantum computer computes expectations which are functions of variational parameters, while the classical computer keeps track these parameters and optimizes the cost function with respect to them.

* mjp327@cornell.edu

II. FINITE DIFFERENCE METHODS AND THEIR OPERATIONAL CIRCUITS

A. Heat and advection equations

We have studied closely the advection and heat equations to understand the circuitry required for them and more complicated equations. We consider only real solutions, which allow us to consider simpler cost functions and only take real-valued Hadamard test expectations.

1. The Heat equation forward and backward euler

Begin with the heat equation,

$$\frac{\partial u}{\partial t} = v\frac{\partial^2 u}{\partial x^2} = \hat{F}u$$

For simplicity, we first use the forwards-euler method, though it requires $O(N)$ timesteps and is exponentially intractable for high numbers of qubits. With this method, the function is discretized as $\frac{u_n^{m+1}-u_n^m}{\tau} = [\hat{F}u]_n^m = v\frac{u_{n+1}^m+u_{n-1}^m-2u_n^m}{h^2}$, which is equivalently written as $u_n^{m+1} = (1-2g)u_n^m + g(u_{n+1}^m + u_{n-1}^m)$. To express this in terms of quantum operators,

$$T_- = \mathbb{1} \qquad T_+ = (1-2g)\mathbb{1} + g(A + A^\dagger)$$

Where $g = \tau v/h^2$. Here, A periodically maps $|i\rangle \to |i+1\rangle$; since it is unitary, $A^\dagger$ does the opposite (See Fig. 1a). A is known as the adder operator and its operator circuit widely known in the literature [1].

Doing similar work for the heat equation with the backwards euler method, one obtains instead $\frac{u_n^{m+1}-u_n^m}{\tau} = [\hat{F}u]_n^{m+1} = v\frac{u_{n+1}^{m+1}+u_{n-1}^{m+1}-2u_n^{m+1}}{h^2}$, yielding

$$T_- = (1+2g)\mathbb{1} - g(A + A^\dagger) \qquad T_+ = \mathbb{1}$$

While the forward euler method doesn't scale well with many qubits, the backwards euler method here is computationally more intensive, as one can see by using Eq. (4), and the fact that the previous timestep expectation must be completed only once. As you can see from Fig. 3b, the cost function is remains relatively high at all iterations. There is a tradeoff between the computational depth of the cost function and the optimizer's ability to find an accurate minimum.

2. The Advection equation and generalities

The advection equation takes advantage of the same quantum operators. The forward euler method is not stable with this equation, and the Lax-Wendroff discretization is instead preferred, in which

$$T_- = \mathbb{1} \qquad T_+ = aA + bA^\dagger + c\mathbb{1}$$

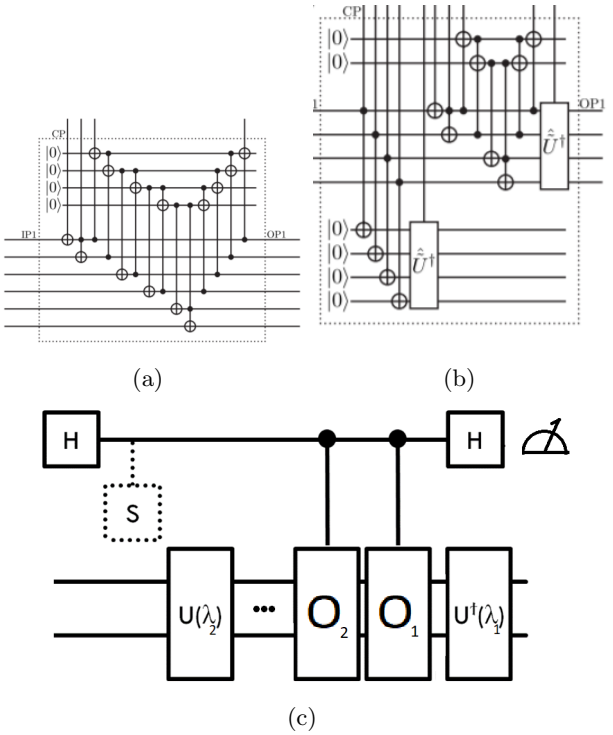


FIG. 1: a) An adder circuit for 6 qubits [1]. The top 'ancilla' or helper qubits are used to realize lower-depth circuits. b) Example of a diagonal operator (left) and adder operator (right). c) Our implementation for measuring the real part of quantum expectations (include the S gate for the imaginary part). We measure spin-Z, counting $|0\rangle$ as 1 and $|1\rangle$ as -1. Through many measurements we obtain $\langle\psi_1| O_1 O_2 |\psi_2\rangle$ = (number of $|0\rangle$ - number of $|1\rangle$) / (number of $|0\rangle$ + number of $|1\rangle$). The leftmost inputs are of course all $|0\rangle$.

where $g = v\tau/h$, $a = g^2 - g/2$, $b = g^2 + g/2$, $c = 1 - 2g^2$. By using simplifications,
$C(|f_t\rangle, |f_{t+1}\rangle) =$
$\langle f_t| (a^2 + b^2 + c^2)\mathbb{1} + 2(abAA + (a+b)cA) |f_t\rangle$
$+ \langle f_{t+1}| |f_{t+1}\rangle$
$- 2(a \langle f_{t+1}| A |f_t\rangle + b \langle f_{t+1}| A^\dagger |f_t\rangle + c \langle f_{t+1}| |f_t\rangle)$. This form is surprisingly concise, and can be used to solve the forwards euler heat equation as well. To obtain the cost function with the backwards euler method, we can substitute $|f_t\rangle \leftrightarrow |f_{t+1}\rangle$ and $g \to -g$. We see it will require extra variational terms. Due to this general form, the heat and advection equations essentially only differ by classical constants.

3. Nonlinearities and the Inviscid Burgers' Equation

Work has been done in this area, and the Inviscid Burgers' Equation follows a similar approach to the advection equation. We will not discuss this more, as it is not a current focus in testing.

III. THE ANZATZ

A. IC-Compatible Ansatz Design

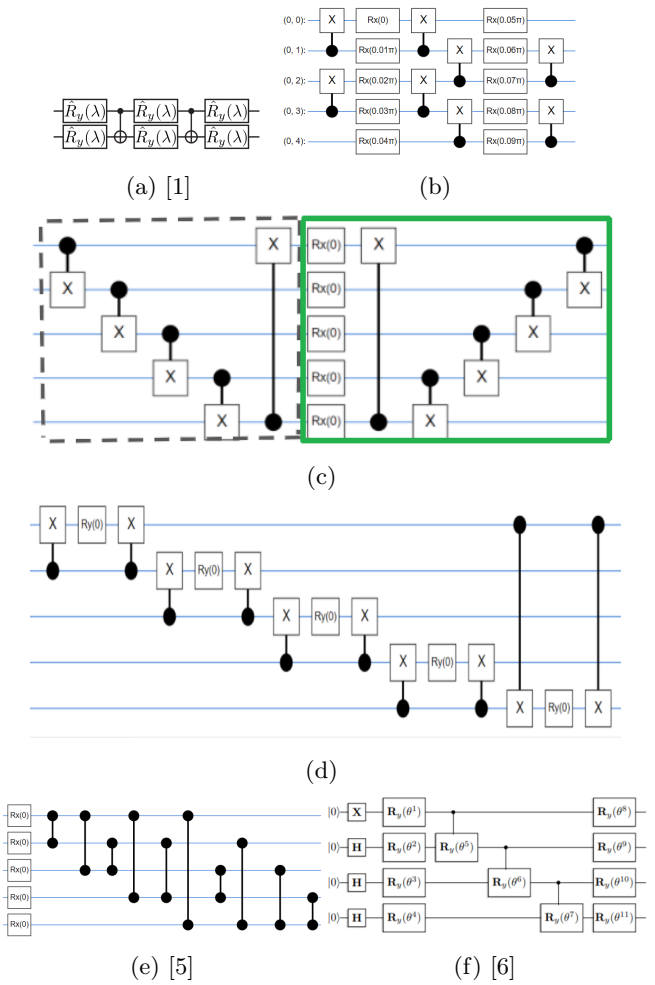


FIG. 2: a) The 2-qubit ansatz used by Lubasch[1]. c) Green part: a 5-qubit version of our initial multiple-qubit ansatz, based on (a). We have had success using this circuit on the order of approx. Appending the dashed part, we get an IC-compatible circuit. However, this was very unsuccessful in testing. b) A version of the ansatz that is compatible with an initial condition circuit prepended to it. The 'sandwich' CNOT format allows the initial condition not to get scrambled as it passes through the circuit, when the rotation angle is zero degrees. d) A circuit motivated by [2]. e) An ansatz used with layerwise learning [5]. f) An ansatz used in a one-step method for solving effectively the heat equation [6]. Note that there is not a consensus on which ansatz to use, and many are not compatible with an IC, which is not useful for us.

A naive approach to creating an ansatz is to use a universal ansatz that spans all of Hilbert space. Such an ansatz would require 2^n parameters and would have to digitally keep track of this many parameters. A classical computer can directly do computations with these amplitudes, which would likely be much faster compared to the increased computations needed when using a hybrid quantum computer approach. Factors include the many observations needed in the Hadamard test, and the fact that optimizing over such an ansatz would be a similar task for digital and quantum computers, but quantum computers would have many extra slowdowns.

Clearly, it is important to consider ansaetze which are equation-specific, and to make relevant simplifications to filter out irrelevant parts of Hilbert space. The equations we consider all have real solutions, and so one assumption is to use only real gates, such as Ry. A simple 2-qubit ansatz was tested on IBM's quantum computer [1] successfully [1] (Fig. 1a). Though an ansatz circuit that spans all Hilbert space is quite simple to create with two qubits, the complexity increases vastly with more qubits [1]. We have been attempting to find an ansatz that scales with many qubits, and uses just enough parameters to represent the hilbert space of the equation considered.

When using a low amount of qubits in an ansatz, it is possible for a digital computer to brute-force solve for initial parameters for a given state. However, starting at around with 5 qubits this becomes intractable (See Fig. 2a). Naturally, it would be useful to use an initial condition circuit which is acted on rightwards by a variational ansatz. This would allow small changes to be made gradually, and would maintain the history of the initial state even with the presence of noise. There have been algorithms developed on representing arbitrary states with circuits, which we use to generate low-depth Gaussian and sinusoidal initial conditions [8].

Previously, the ansaetze we used were incompatible with an initial condition; there is no trivially known $\vec{\lambda}$ for which the variational circuit acts as the unit operator (placing the IC circuit after the variatonal circuit is a very unintuitive solution). As a result, the initial circuit will be scrambled when passing through the circuit.

To solve these issues, we have proposed using circuits which 'sandwich' rotation gates between two CNOT gates, as seen in but not limited to Fig. 2b-e. The results of such testing are shown in Fig. 3. This more naturally centers the circuit about $\vec{\lambda} = \vec{0}$, though it we still have not accounted for equation-specific constraints. A common pattern is that the function becomes jagged, as the low degrees of freedom go into raising and lowering individual amplitudes, as opposed to smoothly doing this in groups. As a result, in the future we will consider further constraints on the ansatz so that it acts more smoothly and physically.

B. IC-Compatible Ansatz Results

We have developed and used a general code framework for testing the heat and advection equations. Here we

test 5-qubit configurations. We refer to the green part of Fig. 2c as the alternating layer ansatz (ALA), which has worked the best so far. In Fig. 3b, we graph of the cost at each iteration in testing the advection equation, for different configurations of ansaetze. The heat equation tends to only optimize the scaling parameter λ_0, which hides interesting behavior. The configurations of each plot shown are: red) Result of optimizing ansatz circuit in Fig. 2d with an initial condition prepended to it, and starting with $\vec{\lambda} = 0$. As you can see, the solution tended to get stuck, likely due to the complexity created by the initial condition. blue) Testing the same ansatz but without an initial condition. It begins with parameters λ found through IC-optimization. yellow) testing the ALA with layerwise optimization. green) Testing the ALA with regular optimization.

We have also tested the performance of the ALA circuit over 5 optimizers (imfil, COBYLA, mgd, Powell, Nelder-Mead). Their performance is quite similar, but we found that imfil is marginally the best (supplemental section).

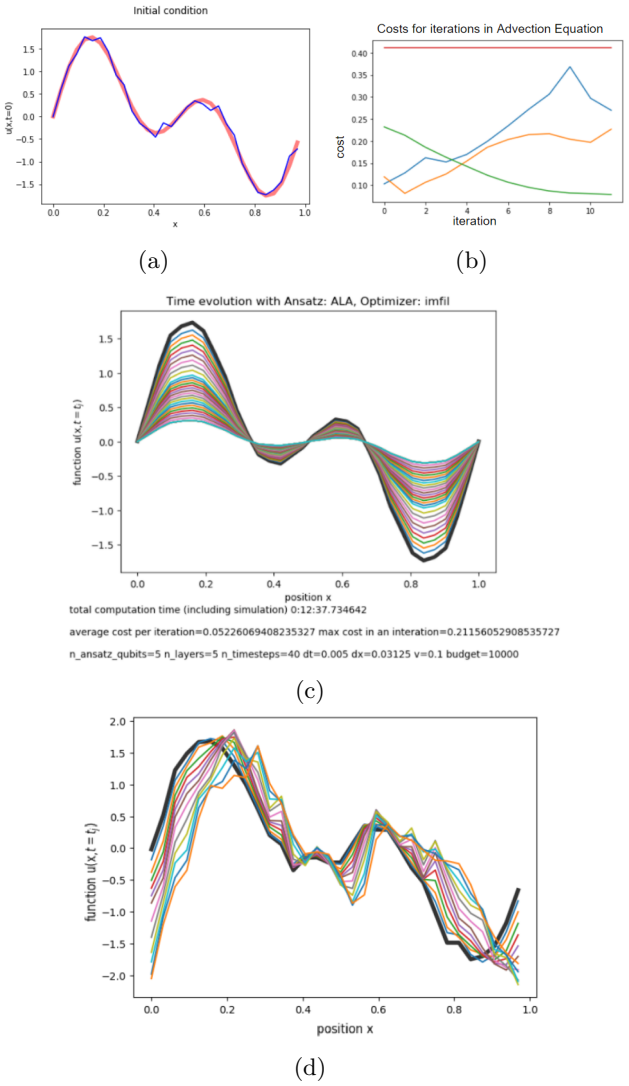


FIG. 3: This entire figure is some of our testing with different 5-qubit ansatz circuits. a) classical brute-force optimization. Here we find the initial condition for the function $cos(x) + cos(2x)$, by minimizing the difference between the function and its represented quantum state $|\lambda_0 U(\vec{\lambda})\,|0\rangle - |f(x)\rangle\,|$ (we call this 'initial-condition optimization', which is only tractable for low amounts of qubits). The optimization (blue) often does not perfectly represent the desired function (red). This is highly dependent on the variational parameter guess used, which is completely arbitrary b) Explained in body text. 3) no-IC ALA optimized in heat equation 4) No-IC ALA optimized in advection equation.

IV. FUTURE WORK

A. Ansatz design

While a lot of work has been done on desining an effective ansatz, there are two main areas that can be improved upon. Since we aim to use an IC circuit with our ansatz, a challenge is creating an IC-compatible circuit that is effective, given the fact that doing this has only worsened optimization performance, as one can see in Fig. 3b.

To solve this problem, I see two criterion we can try to sort through. First, CNOTs and entanglers in quantum states tend to scatter the amplitudes chaotically, rather than in an organized fashion. As a result, when using low numbers of parameters compared to gridpoints, we see jagged results. Somehow smoothing out the results, and allowing parameters to only act toward groups of amplitudes would be beneficial. Easier said than done.

Another approach I can think of is to use ML to sort through ansaetze good at performing a certain function. Perhaps there are certain configurations of circuits good at 'shifting a state over' or 'flattening a state'. In trying to make the ansatz scalable, we could try something modular or functional-based like this.

B. Layerwise learning and optimization

It has been shown that certain alternating layer circuits run into the vanishing gradient problem, making them difficult or impossible to optimize. [4]. We have yet to implement the Adam optimizer and use gradient descent, and test this. However, it seems the only way to proceed in testing high number of qubits is to use a gradient-based optimizer. In Fig. 3d, it is a struggle to converge to a solution. Using gradient descent, and thinking about circuit designs that have scalable gradients, is a goal.

C. 2D and 3D ansaetze

By considering multiple 1D states separately, we can solve higher dimensional separable equations, such as Poisson's equation. We are able to act component-wise, as required. However, we must entangle all of the qubits we use in order to create non-trivial states in the higher dimension we are considering. While this is an easy task, it is not immediately clear how one would act on a singular component of such an entangled state. With more dimensions, we are effectively turning a flat array into multiple dimensions. For instance, our representation of 2D space would be a 1D vector of size 2^{2n}, which could be mapped to a $2^n x 2^n$ grid. However, how would one operate on a single component of such a vector, given we can only consider this 'flattened' 1D array?

D. Higher-order speedups

When implementing these equations, one improvement would be to have a 1-step method that generates a large timestep, rather than doing many small timesteps which are more intensive. From $T_- |f(x, t+\tau)\rangle = T_+ |f(x,t)\rangle$, we can compound these operators together, by raising both sides to some power γ. The cost function then becomes.

$$C(\boldsymbol{\lambda}_t, \boldsymbol{\lambda}_{t+1}) \equiv |\hat{T}_-^{\gamma} |f(\boldsymbol{\lambda}_{t+1})\rangle - \hat{T}_+^{\gamma} |f(\boldsymbol{\lambda}_t)\rangle|^2$$

While one might think this produces a circuit depth too large for NISQ computers, as it uses at most A^{γ} operators, there is a simple solution to using individual repeated adder operators. If one uses the adder operator to some power of two (A^{2^j}), this is equivalent to using the adder operator beginning on the least significant jth bit, as bits valued lower than that power of two are unaffected. As a result, we require only $\log_2(\gamma)$ of these power-of-two adders to implement A^{γ}. Such a simplification may be used as a higher-order speedup in the future.

V. ACKNOWLEDGEMENTS

I would like to thank Prof. McMahon for the opportunity to work in his lab. I would also like to thank my group members for their support.

VI. SUPPLEMENTAL CHARTS

[1] Michael Lubasch, Jaewoo Joo, Pierre Moinier, Martin Kiffner and Dieter Jaksch. Variational quantum algorithms for nonlinear problems, 2019, Phys. Rev. A 101, 010301 (2020); arXiv:1907.09032. DOI: 10.1103/PhysRevA.101.010301.

[2] Vojtech Havlicek, Antonio D. Córcoles, Kristan Temme, Aram W. Harrow, Abhinav Kandala, Jerry M. Chow and Jay M. Gambetta. Supervised learning with quantum enhanced feature spaces, 2018, Nature. vol. 567, pp. 209-212 (2019); arXiv:1804.11326. DOI: 10.1038/s41586-019-0980-2.

[3] Kaining Zhang, Min-Hsiu Hsieh, Liu Liu and Dacheng Tao. Toward Trainability of Quantum Neural Networks, 2020; arXiv:2011.06258.

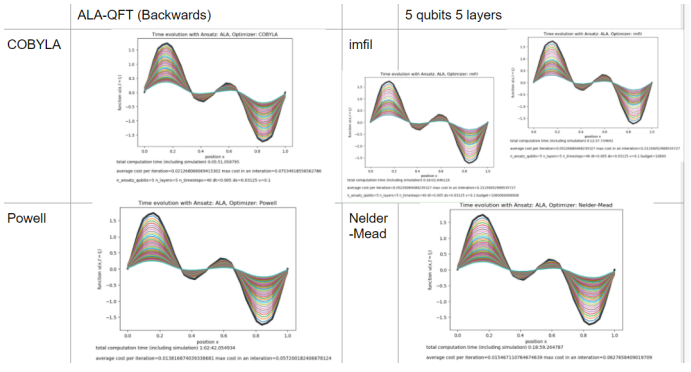
[4] M. Cerezo, Akira Sone, Tyler Volkoff, Lukasz Cincio and Patrick J. Coles. Cost-Function-Dependent Barren Plateaus in Shallow Quantum Neural Networks, 2020; arXiv:2001.00550.

[5] Andrea Skolik, Jarrod R. McClean, Masoud Mohseni, Patrick van der Smagt and Martin Leib. Layerwise learning for quantum neural networks, 2020; arXiv:2006.14904.

[6] Filipe Fontanela, Antoine Jacquier and Mugad Oumgari. A Quantum algorithm for linear PDEs arising in Finance, 2019; arXiv:1912.02753.

[7] John Preskill. Quantum Computing in the NISQ era and beyond, 2018, Quantum 2, 79 (2018); arXiv:1801.00862. DOI: 10.22331/q-2018-08-06-79.

[8] Mikko Mottonen, Juha J. Vartiainen, Ville Bergholm and Martti M. Salomaa. Transformation of quantum states using uniformly controlled rotations, 2004, Quant. Inf. Comp. 5, 467 (2005); arXiv:quant-ph/0407010.



(a)