Supplementary Information for "Learning Parameterized Quantum Circuits with Quantum Gradient"

Keren Li,[1,2,*] Yuanfeng Wang,[2] Pan Gao,[3,†] and Shenggen Zheng[2,‡]

[1] *College of Physics and Optoelectronic Engineering, Shenzhen University, Shenzhen 518060, China*
[2] *Quantum Science Center of Guangdong-Hong Kong-Macao Greater Bay Area (Guangdong), Shenzhen 518045, China*
[3] *Beijing Academy of Quantum Information Sciences, Beijing 100193, China*

* likr@szu.edu.cn
† gaopan@baqis.ac.cn
‡ zhengshenggen@quantumsc.cn

Supplementary Information A: Details on Result 1

Proof for Result 1 — First, we introduce Lemma 1, which give a probability bound that two random vectors have overlap.

Lemma 1. *Suppose $\boldsymbol{u}$, $\boldsymbol{v}$ be two d-dimension random vectors whose elements are chosen from $\mathcal{N}(0,1/d)$, for some $\epsilon \in (0,1)$ there is*

$$Pr(||\langle \boldsymbol{u}|\boldsymbol{v}\rangle|| \geq \epsilon) \leq 4e^{-\frac{d\epsilon^2}{8}} \tag{A1}$$

where $\langle\cdot|\cdot\rangle$ denotes the inner product and $||\cdot||$ denotes the absolute value.

Proof. Since $\boldsymbol{u}$ and $\boldsymbol{v}$ are both chosen randomly from $\mathcal{N}(0,1/d)$, so $\boldsymbol{w} = [w_1, \cdots, w_d] = \frac{\boldsymbol{u}+\boldsymbol{v}}{\sqrt{2}} \sim \mathcal{N}(0,1/d)$, too.

We have the probability

$$\begin{aligned}
&\Pr(||\boldsymbol{w}||^2 - 1 \geq \epsilon) = \Pr(e^{\lambda(||\boldsymbol{w}||^2-1)} \geq e^{\lambda\epsilon}) \\
&\leq \min_{\lambda>0} e^{-\lambda\epsilon}\mathbb{E}[e^{\lambda(||\boldsymbol{w}||^2-1)}] \\
&= \min_{\lambda>0} e^{-\lambda(\epsilon+1)}\mathbb{E}[e^{\lambda(||\boldsymbol{w}||^2)}] \\
&= \min_{\lambda>0} e^{-\lambda(\epsilon+1)}\prod_i \mathbb{E}[e^{\lambda(w_i^2)}] \\
&= \min_{\lambda>0} e^{-\lambda(\epsilon+1)}(\frac{d}{d-2\lambda})^{d/2} \\
&\leq e^{d(\log(1+\epsilon)-\epsilon)/2} \leq e^{-d\epsilon^2/8}
\end{aligned} \tag{A2}$$

for any positive λ and $\mathbb{E}[\cdot]$ be the expectation. The second line in above equation uses the fact that

$$\mathbb{E}[x] = \int_0^\infty xp(x) \geq \int_a^\infty ap(x) = aP(x \geq a). \tag{A3}$$

Similarly to Eq. (A2), we have $\Pr(1 - ||\boldsymbol{w}||^2 \geq \epsilon) \leq e^{-d\epsilon^2/8}$, too. And therefore,

$$\begin{aligned}
\Pr(\langle u|v\rangle \geq \epsilon) \quad &\leq \Pr(||\boldsymbol{w}||^2 - 1 \geq \epsilon) + \Pr(1 - ||\boldsymbol{w}||^2 \geq \epsilon) \\
&\leq 2e^{-d\epsilon^2/8},
\end{aligned} \tag{A4}$$

in the same way,

$$\Pr(-\langle u|v\rangle \geq \epsilon) \leq 2e^{-d\epsilon^2/8}, \tag{A5}$$

too. So $\Pr(||\langle \boldsymbol{u}|\boldsymbol{v}\rangle|| \geq \epsilon) \leq 4e^{-d\epsilon^2/8}$ is proved. ■

Then we start our proof for Result 1.

Proof. In our case, we focus on

$$\nabla_{\boldsymbol{\theta}} f = \frac{\partial \boldsymbol{z}}{\partial \boldsymbol{\theta}} \cdot \nabla_{\boldsymbol{z}} f, \quad \text{where} \quad \frac{\partial \boldsymbol{z}}{\partial \boldsymbol{\theta}} = \begin{pmatrix} \frac{\partial z_1}{\partial \theta_1} & \cdots & \frac{\partial z_d}{\partial \theta_1} \\ \vdots & \ddots & \vdots \\ \frac{\partial z_1}{\partial \theta_m} & \cdots & \frac{\partial z_d}{\partial \theta_m} \end{pmatrix}. \tag{A6}$$

We want to study how large the norm of $\nabla_{\boldsymbol{\theta}} f$ can be. Statistically, the gradient $\boldsymbol{\nabla}_{\boldsymbol{z}} f$ can be obtained as a vector randomly chosen from d dimension complex space. We also denote the i-th row of the matrix $\partial \boldsymbol{z}/\partial \boldsymbol{\theta}$ as a d dimension vector v_i.

First, let us consider $\langle v_i|\boldsymbol{\nabla}_{\boldsymbol{z}} f\rangle$. We represent the complex variables $\boldsymbol{v}_i$ and $\boldsymbol{\nabla}_{\boldsymbol{z}} f$ in real and imaginary parts as $\boldsymbol{v}_i = \partial \boldsymbol{z}/\partial\theta_i = \alpha_i + i\beta_i$, and $\boldsymbol{\nabla}_{\boldsymbol{z}} f = \gamma + i\delta$, where $\alpha, \beta, \gamma, \delta$ can be treated as independent d-dimension random real vectors.

Given assumptions that the norm of $\boldsymbol{\nabla}_{\boldsymbol{z}} f$ and v_i are bounded. We have that $||\alpha||, ||\beta||, ||\gamma||, ||\delta||$ is nearly independent to the Hilbert space dimension d. If their elements are sampled from independent identically distribution

$\mathcal{N}(0,K)$, the independence of $||\alpha|| = \sqrt{\sum_{i=1}^{d}\alpha_i^2}$ to d indicates that $\alpha_i \propto 1/\sqrt{d}$ and therefore $K \sim C/d$ for some constant C. That is $\alpha,\beta,\gamma,\delta$'s elements are chosen randomly from $\mathcal{N}(0,C/d)$ with C being some bounded positive value so that the norm is bounded. Therefore $\alpha/\sqrt{C},\beta/\sqrt{C},\gamma/\sqrt{C},\delta/\sqrt{C}$ are random vectors whose elements are sampling from $\mathcal{N}(0,1/d)$. Then we have

$$\langle \boldsymbol{v}_i|\boldsymbol{\nabla} f\rangle = \langle\alpha|\gamma\rangle + \langle\beta|\delta\rangle - i(\langle\alpha|\delta\rangle - \langle\beta|\gamma\rangle) \tag{A7}$$

and the probability

$$\begin{aligned}
&\Pr(||\langle \boldsymbol{v}_i|\boldsymbol{\nabla} f\rangle|| \geq \epsilon C)\\
&\leq 1 - \Pr(||\langle\alpha|\gamma\rangle + \langle\beta|\delta\rangle|| \leq \frac{\epsilon}{\sqrt{2}}C)\\
&\times\Pr(||\langle\alpha|\delta\rangle - \langle\beta|\gamma\rangle|| \leq \frac{\epsilon}{\sqrt{2}}C)\\
&= 1 - \Pr(|\langle(\alpha,\beta)|(\gamma,\delta)\rangle| \leq \frac{\epsilon}{\sqrt{2}}C)\\
&\times\Pr(||\langle(\alpha,\beta)|(\delta,-\gamma)\rangle|| \leq \frac{\epsilon}{\sqrt{2}}C)\\
&\sim 1 - \Pr(||\langle(\frac{\alpha}{2\sqrt{C}},\frac{\beta}{2\sqrt{C}})|(\frac{\gamma}{2\sqrt{C}},\frac{\delta}{2\sqrt{C}})\rangle|| \leq \frac{\epsilon}{4\sqrt{2}})\\
&\times\Pr(||\langle(\frac{\alpha}{2\sqrt{C}},\frac{\beta}{2\sqrt{C}})|(\frac{\delta}{2\sqrt{C}},-\frac{\gamma}{2\sqrt{C}})\rangle|| \leq \frac{\epsilon}{4\sqrt{2}})\\
&\leq 1 - (1-4e^{-\frac{d\epsilon^2}{128}})(1-4e^{-\frac{d\epsilon^2}{128}})\\
&\leq 8e^{-\frac{d\epsilon^2}{128}}
\end{aligned} \tag{A8}$$

where we have the notation (α,β) means a $2d$-dimension vector whose first d-dimension half part is α and the last half is β, similarly the notation (γ,δ) and $(\delta,-\gamma)$. In the last two lines in Eq. (A8) we use the fact that $||(\alpha,\beta)|| = ||\boldsymbol{v}_i||$ and $||(\gamma,\delta)|| = ||(\delta,-\gamma)|| = ||\boldsymbol{\nabla}_{\boldsymbol{z}} f||$, as well as the result of Lemma.1. The result of Eq. (A8) means in high dimension d, the vector $\boldsymbol{v}_i$ and $\boldsymbol{\nabla}_{\boldsymbol{z}} f$ will orthogonal to each other with probability near to 1.

We estimate the probability that $||\boldsymbol{\nabla}_{\boldsymbol{\theta}} f|| = \sqrt{\sum_{i=1}^{poly(\log(d))}||\langle\frac{\partial \boldsymbol{z}}{\partial \boldsymbol{\theta}_i}|\boldsymbol{\nabla}_{\boldsymbol{z}} f\rangle||^2}$ is larger than $\epsilon C\sqrt{poly(\log(d))}$. Thus,

$$\begin{aligned}
&\Pr(||\boldsymbol{\nabla}_{\boldsymbol{\theta}} f|| \geq \epsilon C\sqrt{poly(\log(d))})\\
&\leq 1 - \prod_{i=1}^{poly(\log(d))}\Pr(||\langle \boldsymbol{v}_i|\boldsymbol{\nabla}_{\boldsymbol{z}} f\rangle|| \leq \epsilon C))\\
&\leq 1 - \prod_{i=1}^{poly(\log(d))}(1-\Pr(||\langle \boldsymbol{v}_i|\boldsymbol{\nabla}_{\boldsymbol{z}} f\rangle|| \geq \epsilon C))\\
&= 1 - (1-8e^{-\frac{d\epsilon^2}{128}})^{poly(\log(d))}\\
&\leq 8poly(\log(d))\cdot e^{-\frac{d\epsilon^2}{128}}
\end{aligned} \tag{A9}$$

Remarkably, we make a variable substitution to transfer C and $\log(d)$ to the right side formula. ■

Supplementary Information B: the Pseudo-code of NOM

In this section, the pseudo-code of NOM is presented in Table 1, corresponding to the theory and Figure 1 of the manuscript.

Table 1 The NOM.

Input: an initial ansatz $U(\boldsymbol{\theta}_{z_0})$ with its structure and therein parameters to be optimized, m, the maximum iteration number, ϵ, the terminate threshold, and ξ, the learning rate.

Set $\Delta f \leftarrow 1$
while $t-1 \in [m] \cap \Delta f \geq \epsilon$ **do**
  function QUANTUM_ALGORITHM_PART($U(\boldsymbol{\theta}_{z_{t-1}})$)
    return $|\boldsymbol{z}_t'\rangle \leftarrow U(\boldsymbol{\theta}_{z_{t-1}})|0\rangle - \xi \mathcal{D}U(\boldsymbol{\theta}_{z_{t-1}})|0\rangle$ ▷ $|\boldsymbol{z}_{t-1}\rangle = U(\boldsymbol{\theta}_{z_{t-1}})|0\rangle$
  end function
  Calculate $\Delta f \leftarrow 1 - \| \langle \boldsymbol{z}_t'| U(\boldsymbol{\theta}_{z_{t-1}})|0\rangle \|^2$
  function CLASSIC_LEARNING_PART($|\boldsymbol{z}_t'\rangle$, $U(\boldsymbol{\theta}_{z_{t-1}})$)
    return $U(\boldsymbol{\theta}_{z_t})$
  end function
end while
return: $U(\boldsymbol{\theta}_{z_t})$

Steps of sub-function of quantum loop (depicted in Hamiltonian Simulation method)

Require: The principal register is initialized with $U(\boldsymbol{\theta}_{z_{t-1}})$, and the three other ancillary systems, $|0\rangle_{lcu}|0\rangle_d|0\rangle_e$
Driving the entire system through the circuit of quantum gradient algorithm in Figure 1.
Post-select on $|0\rangle_{lcu}|0\rangle_d$ and the principal register is output, i.e., $|\boldsymbol{z}_t'\rangle$

Steps of sub-function of classic loop

Require: $U(\boldsymbol{\theta}z_{t-1})$, and $|\boldsymbol{z}_t'\rangle$ is repeatedly produced
function 1ST STEP($U(\boldsymbol{\theta}z_{t-1})$, $|\boldsymbol{z}_t'\rangle$)
  Transport $|\boldsymbol{z}_t'\rangle$ to the inverse circuit of $U(\boldsymbol{\theta}_{z_{t-1}})$
  Measure the possibility on $|0\rangle\langle 0|$
  return: $\boldsymbol{\theta}$, which is tuned to max the aforementioned possibility
end function
If the cost function satisfies the specified criteria, we terminate the process; otherwise, we proceed to the next function
function 2ND STEP($U(\boldsymbol{\theta}z_{t-1})$, $|\boldsymbol{x}'\rangle$) ▷ This step can be realize with various method, we refer to reinforcement learning and depict it later
  Transport $|\boldsymbol{z}_t'\rangle$ to a initialized a slice of circuit $T(\boldsymbol{\alpha})$ and the inverse circuit of $U(\boldsymbol{\theta}z_{t-1})$
  Measure the possibility on $|0\rangle\langle 0|$
  return: $T(\boldsymbol{\alpha})$, which is tuned to max the aforementioned possibility
end function
$U(\tilde{\boldsymbol{\theta}}) \leftarrow U(\boldsymbol{\theta})$ or $T(\boldsymbol{\alpha})U(\boldsymbol{\theta})$ is returned

Supplementary Information C: Quantum gradient algorithm in complex domain

First, we review Result 2. Given that $f(\boldsymbol{z})$ is defined as a polynomial, mapping from $\mathbb{C}^{d+1} \to \mathbb{R}$, the effective gradient operator at $\boldsymbol{z}$ can be expressed as

$$\mathcal{D}(\boldsymbol{z}) = \mathrm{Tr}_{p-1}\left[\mathbb{I} \otimes \rho_{\boldsymbol{z}}^{\otimes p-1} \mathcal{M}_{\mathcal{D}}\right], \tag{C1}$$

where $\rho_{\boldsymbol{z}} = |\boldsymbol{z}\rangle\langle\boldsymbol{z}|$, and $\mathcal{M}_{\mathcal{D}} = \sum_{k=1}^{p} \mathcal{P}_k \mathcal{F} \mathcal{P}_k$, with $\mathcal{P}_k$ denoting the permutation of the first and k-th subsystems in the p-fold tensor product $|\boldsymbol{z}\rangle^{\otimes p}$.

Proof. As $f(z, \bar{z})$ is real valued at every moment, it can be regarded as a real function $f(x, y)$ with two real variables,

$$z = x + iy, \quad \bar{z} = x - iy, \tag{C2}$$

where $x, y \in \mathbb{R}$. By definition of the gradient descent algorithm in the real domain, the iterative formula can be

$$(x', y') = \left(x - \frac{\partial f}{\partial x}\Delta x, y - \frac{\partial f}{\partial y}\Delta y\right) \tag{C3}$$

where x', y' are updated variables. This implies that an updated z' can be expressed as,

$$z' = x + iy - \left[\frac{\partial f}{\partial x}\Delta x + i\frac{\partial f}{\partial y}\Delta y\right]. \tag{C4}$$

Meanwhile,

$$\frac{\partial f}{\partial \bar{z}} = \frac{1}{2}\left(\frac{\partial f}{\partial x} + i\frac{\partial f}{\partial y}\right). \tag{C5}$$

When the learning rate is set to $\Delta x = \Delta y = \xi/2$, we obtain,

$$z' = z - \xi\frac{\partial f}{\partial \bar{z}}, \tag{C6}$$

which is the gradient descent formula used in our case.

On the other side, we substitute $f(\boldsymbol{z}) = \boldsymbol{z}^{\dagger\otimes p}\mathcal{F}\boldsymbol{z}^{\otimes p} = f(\boldsymbol{z}, \bar{\boldsymbol{z}}) = \bar{\boldsymbol{z}}^{T\otimes p}\mathcal{F}\boldsymbol{z}^{\otimes p}$ into partial derivative, then

$$\frac{\partial f}{\partial \bar{z}} = \sum_{\alpha=1}^{k}\sum_{i=1}^{p} c_{\alpha,i}(\boldsymbol{z})F_{\alpha,i}\boldsymbol{z} \doteq \mathcal{D}(\boldsymbol{z})\boldsymbol{z}, \tag{C7}$$

This holds on the fact that

$$\mathcal{F} = \sum_{\alpha=1}^{k} \otimes_{j=1}^{p} a_{\alpha,j}F_{\alpha,j} \tag{C8}$$

such that $c_{\alpha,i}(\boldsymbol{z}) = [\prod_{j=1}^{p}(a_{\alpha,j}\boldsymbol{z}^\dagger F_{\alpha,j}\boldsymbol{z})]/\boldsymbol{z}^\dagger F_{\alpha,i}\boldsymbol{z}$. Therefore,

$$\mathcal{D}(\boldsymbol{z}) = (\mathbb{I}\otimes\boldsymbol{z}^{\dagger\otimes p-1})\sum_{k=1}^{p} P_k A P_k(\mathbb{I}\otimes\boldsymbol{z}^{\otimes p-1}), \tag{C9}$$

where $\mathcal{P}_k$ permutes the first and the k-th subsystems. Up to this point, the problem formulates as a problem encountered in [1–3].

After encoding $\boldsymbol{z}$ into a quantum state, we can get

$$\mathcal{D}(\boldsymbol{z}) = Tr_{p-1}\left[\mathbb{I}\otimes\rho_{\boldsymbol{z}}^{\otimes p-1}\mathcal{M}_{\mathcal{D}}\right], \tag{C10}$$

which can be realized by LCU or Hamiltonian simulation-based methods with the time complexity of $\mathcal{O}(poly(p, \mathcal{F}, \log d))$. ■

Supplementary Information D: Implementation of Quantum Gradient

This section provides detailed instructions to implement the following iterative steps,

$$|\boldsymbol{z}\rangle \leftarrow |\boldsymbol{z}\rangle \pm \xi\mathcal{D}|\boldsymbol{z}\rangle, \tag{D1}$$

where $\mathcal{D}$ is the parameter-dependent operator $\mathcal{D}(\boldsymbol{z})$, and $|\boldsymbol{z}\rangle$ is assumed to encode the variable.

1. Implement Quantum Gradient with Oracles to Query Coefficient Matrix

In this method, the algorithm is achieved through the quantum matrix inversion method via Hamiltonian simulation [1, 4, 5]. Specifically, $\mathcal{D}$ is defined as,

$$\mathcal{D} = \mathrm{Tr}_{p-1}\left[\mathbb{I}\otimes\rho_{\boldsymbol{z}}^{\otimes p-1}\mathcal{M}_{\mathcal{D}}\right]. \tag{D2}$$

The relevant notations are as follows,

$$\rho_{\boldsymbol{z}} = |\boldsymbol{z}\rangle \langle \boldsymbol{z}|,$$
$$\mathcal{M}_{\mathcal{D}} = \sum_{k=1}^{p} \mathcal{P}_k \mathcal{F} \mathcal{P}_k, \tag{D3}$$

where $\mathcal{M}_{\mathcal{D}}$ and $\mathcal{F}$ have relations by $\mathcal{P}_k$, which swaps the first and the k-th subsystems of the entire space $|\boldsymbol{z}\rangle^{\otimes p}$, as

$$\mathcal{S} |\boldsymbol{z}^1\rangle \otimes \ldots |\boldsymbol{z}^{k-1}\rangle \otimes |\boldsymbol{z}^k\rangle \otimes |\boldsymbol{z}^{k+1}\rangle \cdots = |\boldsymbol{z}^k\rangle \otimes \cdots |\boldsymbol{z}^{k-1}\rangle \otimes |\boldsymbol{z}^1\rangle \otimes |\boldsymbol{z}^{k+1}\rangle \cdots .$$

Algorithm 2 The quantum gradient algorithm

1: Initialization:

$$|0\rangle_{lcu} |0\rangle_d |0\rangle_e |0\rangle \to |0\rangle_{lcu} |0\rangle_d |0\rangle_e |\boldsymbol{z}\rangle - \xi |1\rangle_{lcu} |0\rangle_d |0\rangle_e |\boldsymbol{z}\rangle \tag{D4}$$

where $|0\rangle_{lcu}$ is applied by

$$R_y(\xi) = \begin{pmatrix} 1/\sqrt{1+\xi^2} & \xi/\sqrt{1+\xi^2} \\ -\xi/\sqrt{1+\xi^2} & 1/\sqrt{1+\xi^2} \end{pmatrix}. \tag{D5}$$

2: Matrix Inversion on $|1\rangle_{lcu}$:

- First, Phase estimation of $e^{-i\mathcal{D}t}$,

$$|1\rangle_{lcu} |0\rangle_d |0\rangle_e |\boldsymbol{z}\rangle \to |1\rangle_{lcu} |0\rangle_d \sum_n |\widetilde{\lambda}_n\rangle_e \beta_n |n\rangle, \tag{D6}$$

  where $\widetilde{\lambda}_n$ is the binary approximation of λ_n, the eigenvalue of applied $\mathcal{D}$.

- Then, controlled rotation is applied and only $|0\rangle_d$ part is kept,

$$|1\rangle_{lcu} |0\rangle_d \sum_n |\widetilde{\lambda}_n\rangle_e \beta_n |n\rangle \xrightarrow{C_R} |1\rangle_{lcu} \sum_n \widetilde{\lambda}_n |0\rangle_d |\widetilde{\lambda}_n\rangle_e \beta_n |n\rangle \tag{D7}$$

- Finally, phase estimation is inversed,

$$|1\rangle_{lcu} |0\rangle_d \sum_n \widetilde{\lambda}_n |\widetilde{\lambda}_n\rangle_e \beta_n |n\rangle \to |1\rangle_{lcu} |0\rangle_d |0\rangle_e \sum_n \widetilde{\lambda}_n \beta_n |n\rangle \to |1\rangle_{lcu} |0\rangle_d |0\rangle_e \mathcal{D} |\boldsymbol{z}\rangle, \tag{D8}$$

  which accurately established if $\widetilde{\lambda}_n = \lambda_n$.

3: With post-selection, output $|\boldsymbol{z}\rangle \pm \xi \mathcal{D} |\boldsymbol{z}\rangle$, which is based on $|0\rangle_d |0\rangle_e$, which is either result or input of next iteration.

The procedure for conducting the quantum gradient algorithm is outlined in Table 2, where the implementation of the effective gradient $\mathcal{D} = Tr_{p-1}\left[\mathbb{I} \otimes \rho_{\boldsymbol{x}}^{\otimes p-1} \mathcal{M}_{\mathcal{D}}\right]$ is the central component. In this section, we will provide further details on its implementation.

First, we present all temporary states in Table 2. Usually, $\mathcal{D}$ is hermitian but not unitary, therefore can not be directly applied in circuit models. But if the evolution $e^{-i\mathcal{D}t}$ can be efficiently realized, we can effectively implement

$\mathcal{D}\left|\psi\right\rangle$ with the help of an ancillary quantum register in a 'HHL-like' process as

$$
\begin{aligned}
\left|0\right\rangle\otimes\left|0\right\rangle\otimes\left|\psi\right\rangle \quad & \xrightarrow{H} \left|0\right\rangle\otimes\sum_{j=0}^{N-1}\left|j\right\rangle\otimes\left|\psi\right\rangle \\
& \xrightarrow{C-e^{-i\mathcal{D}t}} \left|0\right\rangle\otimes\sum_{j=0}^{N-1}\left|j\right\rangle\otimes e^{-i\frac{2\pi}{N}\mathcal{D}j}\left|\psi\right\rangle=\left|0\right\rangle\otimes\sum_{k}\sum_{j=0}^{N-1}\beta_k\left|j\right\rangle\otimes e^{-i\frac{2\pi}{N}\lambda_k j}\left|k\right\rangle \\
& \xrightarrow{QPE} \left|0\right\rangle\otimes\sum_{k}\beta_k\left|\lambda_k\right\rangle\otimes\left|k\right\rangle \\
& \xrightarrow{C-rotation} \sum_{k}\beta_k(\frac{\lambda_k}{C}\left|0\right\rangle+\sqrt{1-|\frac{\lambda_k}{C}|^2}\left|1\right\rangle)\otimes\left|\lambda_k\right\rangle\otimes\left|k\right\rangle \\
& \xrightarrow{QPE^{-1}} \sum_{k}\sum_{j=0}^{N-1}\beta_k(\frac{\lambda_k}{C}\left|0\right\rangle+\sqrt{1-|\frac{\lambda_k}{C}|^2}\left|1\right\rangle)\otimes e^{-i\frac{2\pi}{N}\lambda_k j}\left|j\right\rangle\otimes\left|k\right\rangle \\
& = \sum_{k}\sum_{j=0}^{N-1}\beta_k(\frac{\lambda_k}{C}\left|0\right\rangle+\sqrt{1-|\frac{\lambda_k}{C}|^2}\left|1\right\rangle)\otimes\left|j\right\rangle\otimes e^{-i\frac{2\pi}{N}\mathcal{D}j}\left|k\right\rangle \\
& \xrightarrow{C-e^{i\mathcal{D}t}} \sum_{k}\sum_{j=0}^{N-1}\beta_k(\frac{\lambda_k}{C}\left|0\right\rangle+\sqrt{1-|\frac{\lambda_k}{C}|^2}\left|1\right\rangle)\otimes\left|j\right\rangle\otimes\left|k\right\rangle \\
& \xrightarrow{H} \sum_{k}\beta_k(\frac{\lambda_k}{C}\left|0\right\rangle+\sqrt{1-|\frac{\lambda_k}{C}|^2}\left|1\right\rangle)\otimes\left|0\right\rangle\otimes\left|k\right\rangle \\
& \xrightarrow{P=\left|0\right\rangle\left\langle 0\right|} \sum_{k}\beta_k\frac{\lambda_k}{C}\left|0\right\rangle\otimes\left|0\right\rangle\otimes\left|k\right\rangle\propto\left|0\right\rangle\otimes\left|0\right\rangle\otimes\mathcal{D}\left|\psi\right\rangle. \qquad \text{(D9)}
\end{aligned}
$$

In a further step, the evolution $e^{-i\mathcal{D}t}$ can be approximately constructed with another evolution $e^{-i\mathcal{M}_{\mathcal{D}}t}$ and the help of $m(p-1)$ copies of state $\rho_{\boldsymbol{x}}=\left|\boldsymbol{x}\right\rangle\left\langle\boldsymbol{x}\right|$, within accuracy $\mathcal{O}(\frac{t^2p^2||\mathcal{F}||_{max}^2}{m})$, in a 'quantum principal component analysis(QPCA)' way as

$$
\begin{aligned}
& \underbrace{Tr_{p-1}[e^{-i\mathcal{M}_{\mathcal{D}}\frac{t}{m}}Tr_{p-1}[e^{-i\mathcal{M}_{\mathcal{D}}\frac{t}{m}}\cdots Tr_{p-1}[}_{m-trace}e^{-i\mathcal{M}_{\mathcal{D}}\frac{t}{m}}\rho^{\otimes p}e^{i\mathcal{M}_{\mathcal{D}}\frac{t}{m}}]\cdots\rho^{\otimes p-1}e^{i\mathcal{M}_{\mathcal{D}}\frac{t}{m}}]\rho^{\otimes p-1}e^{i\mathcal{M}\frac{t}{m}}] \\
& =(e^{-i\mathcal{D}\frac{t}{m}})^m\rho(e^{i\mathcal{D}\frac{t}{m}})^m+\mathcal{O}(m||\mathcal{D}||_{max}^2\frac{t^2}{m^2}) \\
& =e^{-i\mathcal{D}t}\rho e^{i\mathcal{D}t}+\mathcal{O}(\frac{t^2p^2||\mathcal{F}||_{max}^2}{m}), \qquad \text{(D10)}
\end{aligned}
$$

since we have $\mathcal{D}=Tr_{p-1}[\mathbb{I}\otimes\rho_{\boldsymbol{x}}^{\otimes p-1}\mathcal{M}_{\mathcal{D}}]$ and the evolution $e^{-i\mathcal{M}_{\mathcal{D}}\frac{t}{m}}$ can be approximated by Trotter expansion

$$
e^{-i\mathcal{M}_{\mathcal{D}}\frac{t}{m}}=\prod_{k=1}^{p}\mathcal{P}_k e^{-i\mathcal{F}\frac{t}{m}}\mathcal{P}_k+\mathcal{O}(\frac{t^2p^2||\mathcal{F}||_{max}^2}{m^2}), \qquad \text{(D11)}
$$

which is shown in Figure 2.

Lastly, as known in general quantum singling processing(QSP) Hamiltonian simulation[6], $e^{-i\mathcal{F}\frac{t}{m}}$ can be simulated within accuracy ϵ_h at the cost of $\mathcal{O}(sp||\mathcal{F}||_{max}\frac{t}{m}+\frac{\log 1/\epsilon_h}{\log\log 1/\epsilon_h})$ queries of U_1, U_2 and $\mathcal{O}\big((sp||\mathcal{F}||_{max}\frac{t}{m}+\frac{\log 1/\epsilon_h}{\log\log 1/\epsilon_h})(d^p+qpoly(\log q))\big)$ extra basic quantum gates, where s denotes the sparsity of $\mathcal{F}$, q is the bit-accuracy of elements of $\mathcal{F}$, and the two Oracles,

$$
\begin{aligned}
U_1\left|j,k\right\rangle\left|0\right\rangle &= \left|j,k\right\rangle\left|\mathcal{F}_{j,k}\right\rangle, \\
U_2\left|j,l\right\rangle &= \left|j,k_{\mathcal{F}}(j,l)\right\rangle. \qquad \text{(D12)}
\end{aligned}
$$

To sum up, we implement the effective gradient $\mathcal{D}$ in the following logical flow,

$$U_1, U_2 \xrightarrow{QSP} e^{-i\mathcal{F}\frac{t}{m}} \xrightarrow{Trotter} e^{-i\mathcal{M}_{\mathcal{D}}\frac{t}{m}} \xrightarrow{QPCA} e^{-i\mathcal{D}t} \xrightarrow{HHL-like} \mathcal{D}. \tag{D13}$$

To visualize the algorithm, we present a circuit diagram in Figure 1 and Figure 2. More details have been described in related work in [1, 5].

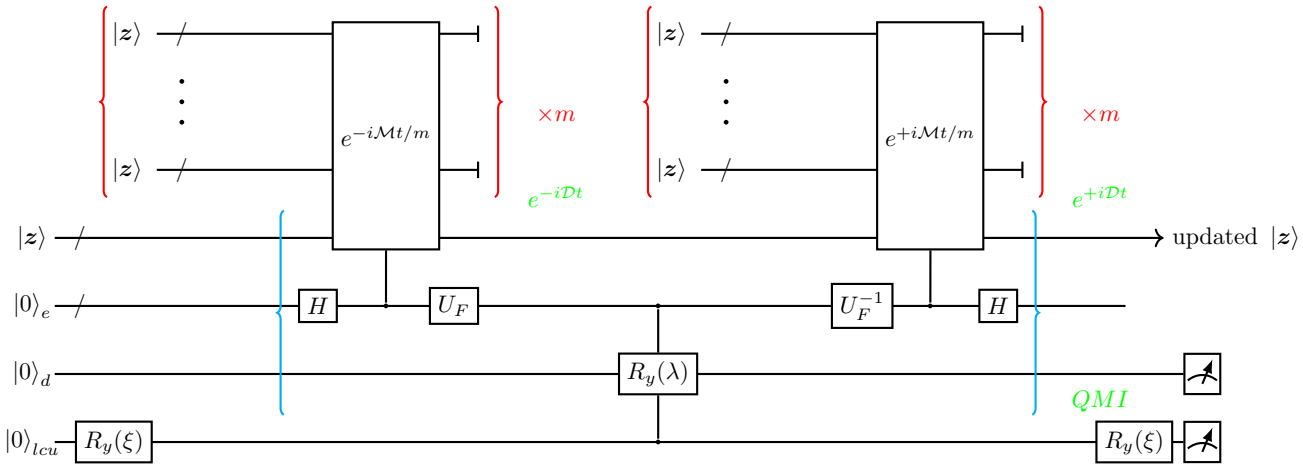

FIG. 1. Circuit diagram to process Eq. (D1). Component in blue bracket is for Quantum Matrix Inverse and Components in red bracket is for simulating $e^{-i\mathcal{D}t}$.

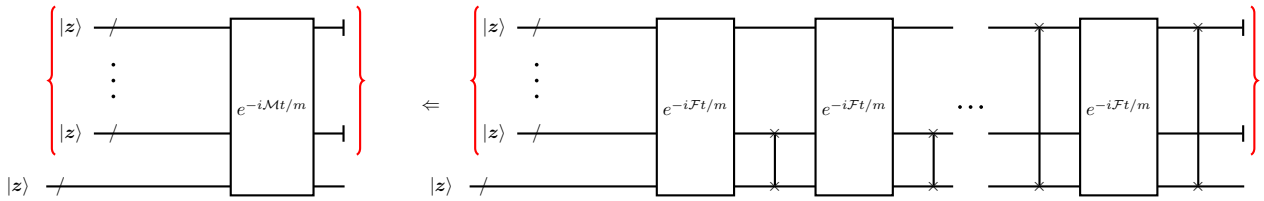

FIG. 2. Schematic of implementation of $e^{-i\mathcal{M}\frac{t}{m}}$. The right hand side shows the series performance of $\{\mathcal{P}_k\}$ with k from 1 to p. The "/" means multi qubits and "|" means dropped of registers in the last of the circuits.

2. Quantum Gradient algorithm with $\mathcal{F}$ Efficiently Decomposed

If $\mathcal{F}$, the coefficient matrix can be decomposed efficiently, there is alternative method. Consider that $\mathcal{F}$ can be represented by a tensor product of unitaries, which is geometry k-local, we have,

$$\begin{aligned} \mathcal{F} &= \sum_{\alpha=1}^{p} \otimes_{j=1}^{m} P_{\alpha,j}, \\ \mathcal{D} &= \sum_{\alpha=1}^{p} \prod_{\beta\neq\alpha} \langle \boldsymbol{z}| \otimes_{j=1}^{m} P_{\beta,j} |\boldsymbol{z}\rangle \otimes_{j=1}^{m} P_{\alpha,j}, \end{aligned} \tag{D14}$$

where $d = \prod_{j=1}^{m} \mathrm{d}(P_{\alpha,j})$ and $\mathrm{d}(P_{\alpha,j})$ is the dimension of $P_{\alpha,j}$ which should $\leq 2^k$. Under the condition that ξ is small, we can interpret $I - \xi\mathcal{D}$, which is equivalent to realize Eq. (D1), as,

$$e^{-\xi\mathcal{D}} \sim \prod_{\alpha=1}^{p} \otimes_{j}^{m} e^{-\xi \prod_{\beta\neq\alpha} \langle \boldsymbol{z}| \otimes_{j=1}^{m} P_{\beta,j} |\boldsymbol{z}\rangle \otimes_{j=1}^{m} P_{\alpha,j}}. \tag{D15}$$

Here, we introduce two method to simulate this $e^{-\xi\mathcal{D}}$.

The first method is based on the LCU. If $P_{\alpha,j}$ in Eq. (D14) can be presented by Pauli matrix, then the LCU method can be employed to implement the target process with a specified probability. The gradient operator can be rewritten as

$$\mathcal{D} = \sum_{\alpha=1}^{p}\sum_{i=1}^{m} \frac{\prod_j \langle \boldsymbol{z}| P_{\alpha,j} |\boldsymbol{z}\rangle}{\langle \boldsymbol{z}| F_{\alpha,i} |\boldsymbol{z}\rangle} F_{\alpha,i} = \sum_{\alpha=1}^{p}\sum_{i=1}^{m} c_{\alpha,i} F_{\alpha,i}, \tag{D16}$$

where $c_{\alpha,i}$ are parameters to be pre-determined. In additional experiments, $\mathcal{O}(kp)$ measurements on $P_{\alpha,j}$ are required to obtain $c_{\alpha,i}$. Each decomposition of $\mathcal{F}$ must be measured to evaluate the cost function. To implement $I - \xi\mathcal{D}$, two ancillary registers are needed. The first is a single qubit to apply $I - \xi\mathcal{D}$, and the second is a $\log(kp)$-qubit register for implementing $\mathcal{D}$. The circuit shown in Fig. 3 illustrates this method, where the principal register outputs $|\boldsymbol{z}\rangle$ if post-selections are applied. This setup has a success probability of $\mathcal{O}(1/(kp)^2)$ for observing the $|0\rangle_e |0\rangle_d$ subsystem. Further details are provided in related work [3].

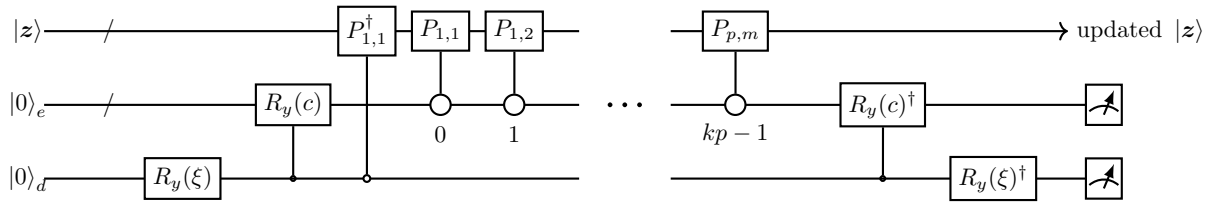

FIG. 3. Circuit to process Eq. (D1) with method of linear combination of unitaries.

The second method is from Ref.[7]. If $|\boldsymbol{z}\rangle$ is chosen as a tensor product state, $\prod_{\beta\neq\alpha} \langle \boldsymbol{z}| \otimes_{j=1}^{m} P_{\beta,j} |\boldsymbol{z}\rangle \equiv b_\beta$ can be efficiently evaluated with $\mathcal{O}(2^L)$ times calculations, where $L \geq k$ is a correlation length. We set $e^{-i\xi D} = \prod_\alpha e^{-i\xi D_\alpha}$ and $D_\alpha = \sum_k a_{\alpha,k}\sigma_{\alpha,k}$. By approximating $e^{-i\xi D_\alpha} |z\rangle = \frac{1}{c} \otimes_j^m e^{-\xi b_\beta P_{\alpha,j}} |z\rangle$, a system of linear equation formulates as

$$\sum_k a_{\alpha,k} \langle \boldsymbol{z}| \sigma_{\alpha,k'}^{\dagger}\sigma_{\alpha,k} |\boldsymbol{z}\rangle = \frac{-ib_\beta}{c} \langle \boldsymbol{z}| \sigma_{\alpha,k'}^{\dagger} \otimes_j^m P_{\alpha,j} |\boldsymbol{z}\rangle, \tag{D17}$$

where $c = \sqrt{\langle \boldsymbol{z}| e^{-2\xi\mathcal{D}} |\boldsymbol{z}\rangle}$ is the re-normalization factor. In the initial step $e^{-i\xi D_1}$, a tensor product state can be chosen, allowing the coefficients in Eq. (D17) to be efficiently computed. Solving this system provides an efficient determination of $e^{-i\xi D_1}$. For $e^{-i\xi D_\alpha}$ with $\alpha > 1$, since $\prod_{\beta=1}^{\alpha-1} e^{-i\xi D_\beta}$ has already been applied, $|\psi\rangle$ becomes non-local, and the correlation length increases with each subsequent α. This growth in correlation length accelerates with α, necessitating iterative updates with gradient adjustments. Therefore, an approximate approach is recommended, as referred in related work [7].

Supplementary Information E: Details on Numerical simulation

The first demonstration problem is the Max-Cut problem. Suppose there are n vortex in the graph, the observable that we aim to maximize is

$$H_c = \sum_{\langle i,j\rangle} \frac{1 - \sigma_{z_i}\sigma_{z_j}}{2},$$

where σ_{z_i} and σ_{z_j} represent the Pauli-Z operators acting on qubits i and j, $\langle i,j\rangle$ exists only there is an edge. We are noticed of the results in [8]. The mixer Hamiltonian can be chosen as

$$H_b = \sum_{i=0}^{n-1} \sigma_{x_i}, \tag{E1}$$

and can be used as QAOA method, where σ_{x_i} is the Pauli-X operators acting on qubits i. The results indicate that if the graph is a ring(that is, regular-2 graph), the maximum value of the objective function $\langle H_c\rangle$, which is estimated under state output by a QAOA circuit is lower bounded as $n(2p+1)/(2p+2)$, where p is the depth of QAOA circuits. Similarly, increasing p will produce a better approximate ratio for more general graphs.

Specifically, we consider a random graph with 4 vortex. The corresponding Hamiltonian is depicted above and $\langle i, j\rangle$

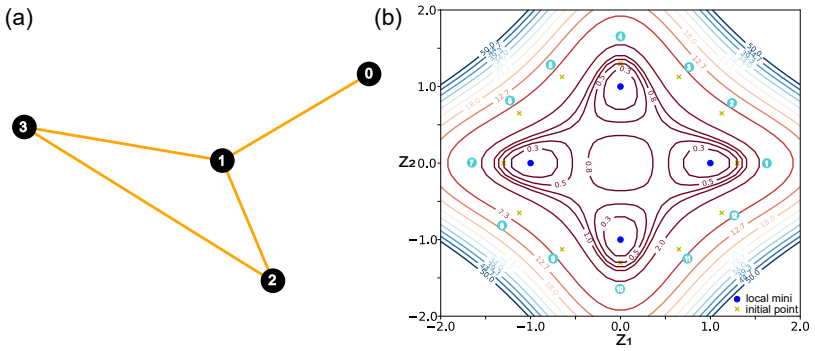

FIG. 4. (a) is a random graph with 4 vortex. (b) is the contour plot of the polynomial in the second example with $z_3 = 0$.

exists only there is an edge in Figure 4(a).

For initial ansatz construction, we set 4 quantum circuit. The first one formulates as a single layer QAOA ansatz,

$$e^{-iH_b\beta}e^{-iH_c\gamma},$$

where $H_b = \sum_{i=0}^{3}\sigma_{x_i}$. The second one is a modified single layer QAOA ansatz,

$$e^{-iH_b\beta}R_x(\alpha)e^{-iH_c\gamma},$$

where $R_x(\alpha)$ is a global rotation about x-axis with angle α. The third one is inspired by hardware efficient ansatz, which is

$$e^{-i\sigma_{z_0}\sigma_{z_3}\alpha}e^{-i\sigma_{z_1}\sigma_{z_2}\beta}e^{-i\sigma_{x_2}\sigma_{x_3}\gamma}e^{-i\sigma_{x_0}\sigma_{x_1}\eta}.$$

The fourth one is

$$e^{-i\sigma_{z_0}\sigma_{z_3}\alpha}e^{-i\sigma_{z_1}\sigma_{z_2}\beta}R_y(\alpha)e^{-i\sigma_{x_2}\sigma_{x_3}\gamma}e^{-i\sigma_{x_0}\sigma_{x_1}\eta},$$

where $R_y(\alpha)$ is a global rotation about y-axis with angle α. Then using typical gradient-based method we can find 4 corresponding sub-optimal parameter configurations which is not the optimal answer but with vanished gradients. Then we use these circuit and their parameter configurations as our initialization and learn our NOM. Based on this, we can get the results 2 in the main text.

The second simulation is to optimize a polynomial. Polynomials are a kind of function that has many applications. For example, nonlinear equations and linear regression. Here we show a case that is optimizing of a polynomial

$$\begin{aligned} f = \quad & z_1^2\overline{z_1}^2 + z_2^2\overline{z_2}^2 + z_3^2\overline{z_3}^2 + 2(\overline{z_1}^2 z_2^2 + z_1^2\overline{z_2}^2) \\ & +6z_1z_2\overline{z_1z_2} - 2z_1z_3\overline{z_1z_3} - 2z_2z_3\overline{z_2z_3} \\ & -2z_1\overline{z_1} - 2z_2\overline{z_2} + 6z_3\overline{z_3} \\ & +2(z_3^2 + \overline{z_3}^2) + 1 \end{aligned} \tag{E2}$$

which can be expressed as

$$f(Z) = Z^{\otimes 2}AZ^{\dagger\otimes 2}, \tag{E3}$$

where $Z = (1, z_1, z_2, z_3)$ and the coefficient matrix

$$A = X^{\otimes 4} + Y^{\otimes 4} + Z^{\otimes 4}. \tag{E4}$$

It is observed that the cost function features isolated local minima within the region $(z_1, z_2, z_3) \in [-2, 2]^{\otimes 3}$, and our simulation focuses on optimizing within this region. In Figure 4(b), we present a case where $z_3 = 0$. We start with 12 initial points, each positioned on a circle with a radius of 1.3. To generate these initial points, we use a parameterized circuit to approximate them via a state-to-state method. These circuit with their parameter configurations serve as our initialization for the method. Then we conduct the NOM as depicted in pseudo-codes. Based on this, we can get

the results of demonstration in the main text.

Influence by PQC Approximation Method — We additionally investigate how inaccuracy of PQC approximation method affects the protocol, which is RL method work for. In this simulation, we consider the same demonstration problems and the settings are the same as previous ones.

Figures 5 and 6 illustrate the deviation between the ideal scenario—where every updated state from the quantum gradient algorithm is assumed to be perfectly captured by a PQC—and the scenario where the state is approximated by a PQC with added random noise, weighted by the magnitude of disturbance. In these cases, we chose 0.2 as learning rate in simulation of quantum algorithm.

We examine the influence of disturbance magnitude on each iteration, as shown in sub-figures (a) of both scenarios. Snapshots at three disturbance magnitudes—0.02(0.02), 0.06(0.04), and 0.10(0.06)—are presented in sub-figures (b)–(d) of Figure 5 (Figure 6). The magnitude of disturbance reflects the accuracy of the classical training process in approximating the quantum gradient state with a PQC. The results in Figures 5 and 6 indicate that the protocol remains robust under a certain level of disturbance, but if the approximation quality deteriorates significantly, convergence is disrupted.

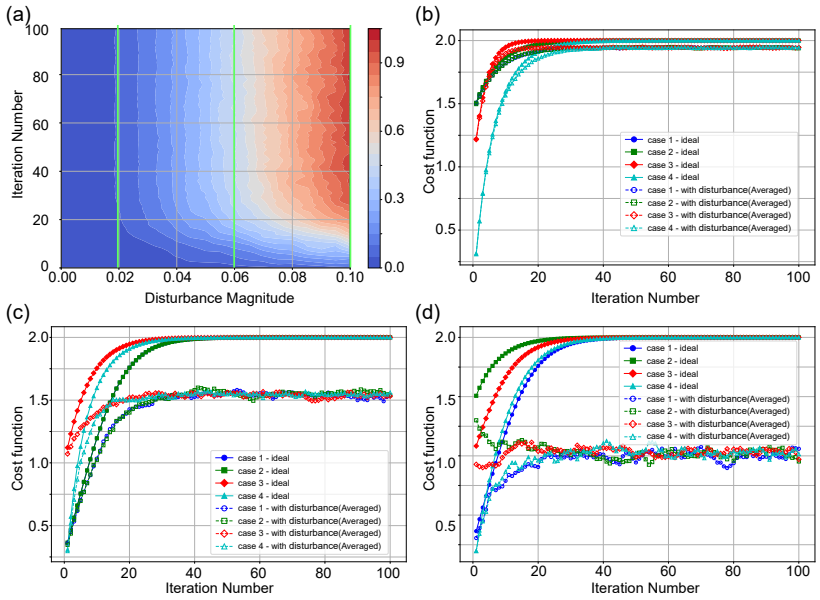

FIG. 5. Influence of disturbance magnitude on the Max-Cut problem: 4 initial ansatze are tested. Sub-figure (a) shows the overall effect of varying disturbance magnitudes across all iterations. Sub-figures (b), (c), and (d) provide detailed snapshots at specific disturbance magnitudes of 0.02, 0.06, and 0.10, respectively. The results are averaged over 50 runs to ensure statistical reliability.

Additional evidence on BP-Exhibiting Circuits — TensorFlow tutorials have provided examples of circuits that exhibit barren plateaus (BPs). In an additional set of simulations, we demonstrate that our indicator remains effective even when BPs emerge.

To make this more evident, we conduct sequential numerical simulations on a more general Max-Cut problem, with system sizes ranging from 4 to 10 qubits:

$$H = \sum_{i=0}^{K} \frac{Z_i Z_{i+1}}{2}, \tag{E5}$$

with periodic boundary condition, where Z_i is the Pauli-Z matrix acting on the i-th qubit.

For each system size, we simulate 200 random circuits with 50 layers following the structure outlined in the TensorFlow documentation. We evaluate the parameter gradients and record their variance, reproducing their analysis. In addition, for each output state of these random circuits, we apply our quantum gradient algorithm with a learning rate of 0.2 to produce a corresponding gradient state. We then compute the average value of our indicator, as defined in Eq.(11) in the main text, across the 200 circuits.

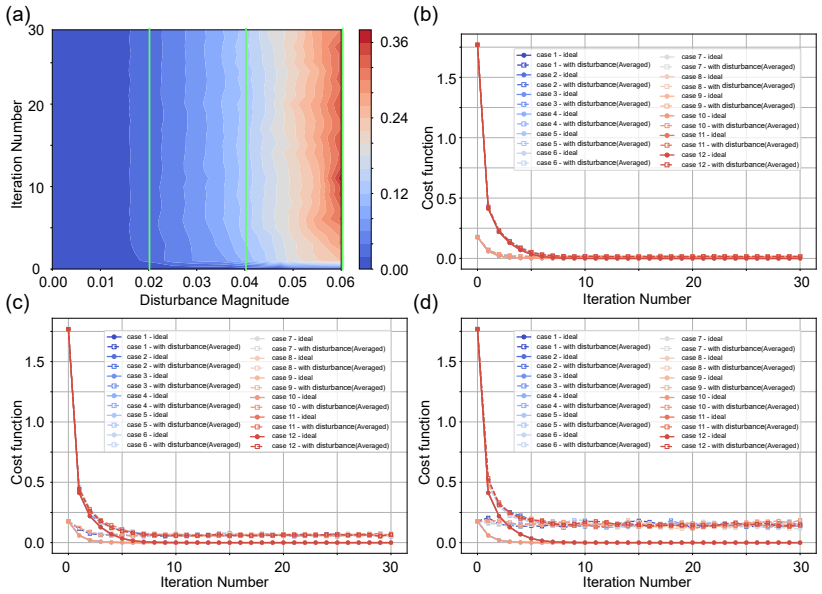

FIG. 6. Influence of disturbance magnitude on polynomial optimization: 12 initial points are tested. (a) depicts the overall effect of varying disturbance magnitudes across all iterations. (b), (c), and (d) present detailed snapshots at specific disturbance magnitudes of 0.02, 0.04, and 0.06, respectively. The results are averaged over 50 runs to ensure robustness.

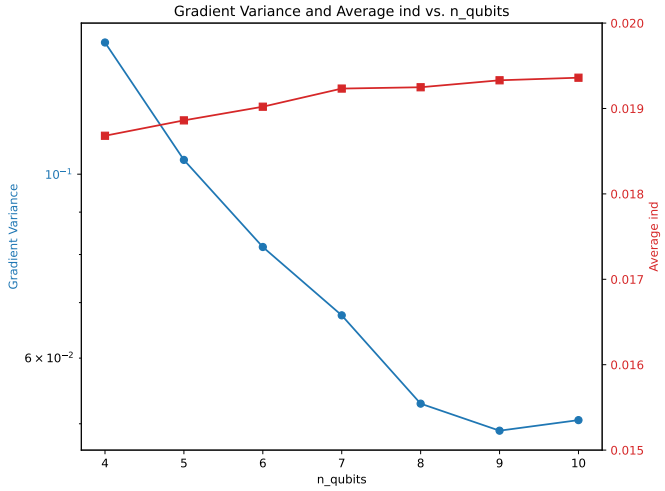

FIG. 7. Indicators helps to avoid Barren Plateau

The results are shown in Figure 7. As seen in the plot, while the gradient variance decreases rapidly with increasing problem size (i.e., number of qubits), the value of the indicator remains nearly constant. This suggests that although BPs may occur in standard PQCs, our indicator remains stable and serves as a reliable signal for learning progress.

Supplementary Information F: Quantum circuit synthesis with reinforcement learning

For a given initial state $|z\rangle$ and target state $|z'\rangle$, the RL algorithm learns a policy that generates a PQC (by adding gates sequentially), which transform $|z\rangle$ to approximate $|z'\rangle$. The policy is learned by Proximal Policy optimization

(PPO) algorithm [9] with the built-in MLP functional approximator. The reward is set to

$$r = \begin{cases} 10 & if \quad f \geq f_t \\ -5.0 & if \quad f < f_t \quad and \quad circuit\_len >= max\_circuit\_len \\ \max(\frac{f-f_{prev}}{f_t-f_{prev}}, -1.0) - p & elsewise \end{cases} \tag{F1}$$

where $f = \langle z'| U(\theta) |z\rangle$ is the fidelity of current step (circuit), f_{prev} is the fidelity from last step of the episode and f_t is a pre-defined fidelity threshold. p is a penalty of adding a gate to encourage shallow circuit which makes the agent always trying to find the shortest possible protocol to prepare the target state. The fidelity f is obtained by optimizing all parameters $\{\theta\}$ of the quantum circuit for current step. The optimization is done using the COBYLA algorithm in scipy optim package. The episode terminates if fidelity value is great than f_t or the circuit depth(number of gates) exceeds the maximum allowed circuit depth. This is to prevent episodes, especially at the beginning of training, become exceedingly long leading to unfeasible training times. All the quantum circuit simulation is done using the Qulacs python package [10].

The pseudo-code of reinforcement algorithm for PQC learning is concluded in Table 3.

Table 3 Reinforcement Learning Procedure for PQC

Input: Initial state $|z\rangle$, target state $|z\rangle'$, maximum circuit depth d_{max}, reward penalty p, fidelity threshold f_t, convergence threshold ϵ
Output: Learned policy for circuit synthesis and corresponding PQC
Initialize gym environment env($|z\rangle$, $|z\rangle'$, d_{max}, p, f_t)
$\theta_{\text{prev}} \leftarrow \theta$
Define actions (quantum gates set $g_i(\theta_i)$), reward function and termination condition.
repeat
    Learn PPO(γ, n_{epochs}, $clip\_range$, $learning\_rate$)
until maximum learning steps
Reset environment
repeat
    Predict action for the current state: $a = \text{PPO}_{opt}(state)$
    Take a step: $state$, $reward$, $done$, $info$ = env.step(a)
until $done$
Return: Learned policy PPO_{opt} and parameterized quantum circuit

For both examples (Max-Cut problem and polynomial optimization), we choose from the gate set consisting of single qubit Pauli rotation gates and two-bit Pauli rotation gates, namely $R_X(\theta)$, $R_Y(\theta)$, $R_Z(\theta)$, $R_{XX}(\theta)$, $R_{YY}(\theta)$, $R_{ZZ}(\theta)$ (by definition $R_X(\theta) = exp(-i\frac{\theta}{2}X)$, $R_{XX}(\theta) = exp(-i\frac{\theta}{2}X \times X)$ etc.). In the learning process, we set the maximum circuit depth to be 10. The parameter setting of PPO algorithm is $\gamma = 0.99$, $n_{epochs} = 4$, $clip\_range = 0.2$, $learning\_rate = 0.0001$.

We tested the algorithm on random 4-qubit state with different quantum gradient step size, resulting in different target state. Figure 8 shows the influence of number of learning episodes on the fidelity of approximate state with learned PQC and the target state. Result from two environments, corresponding to with different target states are shown. We can see that, in general, the RL algorithm can approximate target state better with increased learning episode, but reaching some plateau potentially restricted by the maximum episode length (e.g. circuit depth).

Supplementary Information G: Error Accumulation and Analysis

In this section, we depict the fact that the iterative method is insensitive to error of variables produced either in preparation or readout.

Proposition 1. *The iterative gradient algorithm is insensitive to errors from an inaccurate quantum state.*

The inaccurate quantum state is caused by either inaccuracy of devices or optimization methods, which leads a perturbation on output variable. As the quantum gradient algorithm faithfully performs calculation of gradients and the iterative operations.

We directly analyze its classical behavior. The iterative equations are listed,

$$\boldsymbol{x}' \leftarrow \boldsymbol{x} - \xi \nabla_{\boldsymbol{x}} f(\boldsymbol{x}), \tag{G1}$$

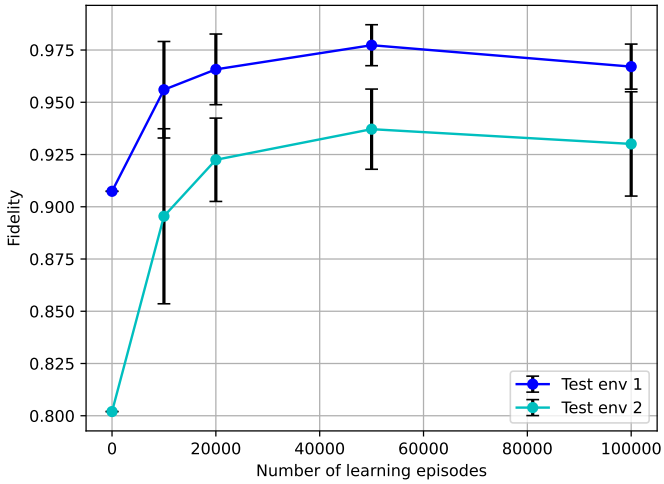

FIG. 8. Effects of number of learning episodes on learned policies: two environments with different target states are tested. Mean and standard deviation of fidelity values are calculated from five independent runs. The fidelity between the initial state and target state is shown at $x = 0$.

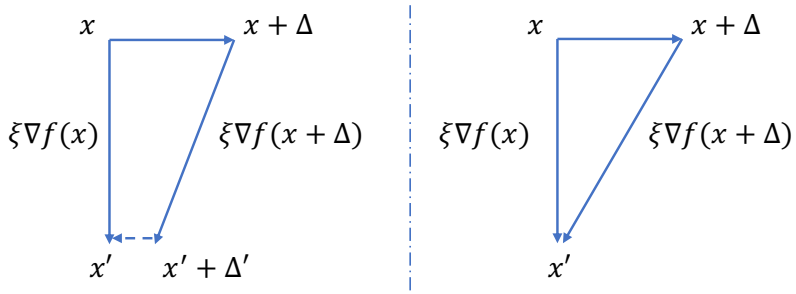

FIG. 9. Gradient calculation with errors.

If $\boldsymbol{x}$ has a small perturbation $\boldsymbol{x} + \Delta$, Eq. (G1) becomes,

$$\begin{aligned} \boldsymbol{x}' \leftarrow &\quad \boldsymbol{x} + \Delta - \xi\nabla_{\boldsymbol{x}} f(\boldsymbol{x} + \Delta) \\ = &\quad \boldsymbol{x} + \Delta - \xi(\nabla_{\boldsymbol{x}} f(\boldsymbol{x}) + \nabla^2_{\boldsymbol{x}} f(\boldsymbol{x})\Delta), \end{aligned} \tag{G2}$$

where $\Delta - \xi\nabla^2_{\boldsymbol{x}} f(\boldsymbol{x})\Delta$ implies the perturbation on updated point. We are noticed that the Hessian matrix $\nabla^2_{\boldsymbol{x}} f(\boldsymbol{x})$ is positive definite nearby local minimum, so Δ is curved back by $\xi\nabla^2_{\boldsymbol{x}} f(\boldsymbol{x})\Delta$, denoted as Δ' (shown in Fig.9). This is due to small value of ξ, and we can adjust this mitigation by ξ. An ideal case is in the right-side of the figure with proper ξ and strength of Δ,

$$\boldsymbol{x} \pm \xi\nabla_{\boldsymbol{x}} f(\boldsymbol{x}). \tag{G3}$$

Supplementary Information H: Details on Feasibility

Analysis of barren plateaus in classical training part— Classical gradient-based methods for training $T(\boldsymbol{\alpha})$ may encounter barren plateaus. In this section, we analyze this situation and clarify the mitigation for such barren plateaus.

We focus on $T(\boldsymbol{\alpha}) = \prod_{j=1}^{L} T_j(\alpha_j)$, where $T_j(\alpha_j)$ defines a single-layer circuit, and L is finite due to the choice of small ξ values that limits the norm of $\xi\mathcal{D}$.

The partial derivative with respect to the k-th parameter for training $T(\boldsymbol{\alpha})$ is given by,

$$\frac{\partial c_2}{\partial \alpha_k} = i \langle \boldsymbol{z}| T_-^\dagger \left[V_k,\ T_+^\dagger |\boldsymbol{z}'\rangle \langle \boldsymbol{z}'| T_+ \right] T_- |\boldsymbol{z}\rangle , \tag{H1}$$

where V_k denotes the derivative of a single-layer circuit $T_k(\alpha_k)$, $[\cdot]$ represents the commutator, and

$$T_- = \prod_{j=1}^{k} T_j(\alpha_j), \quad T_+ = \prod_{j=k+1}^{L} T_j(\alpha_j).$$

Let us observe the behavior at the initial moment. We initialize $T(\boldsymbol{\alpha}) = T_+ T_-$ to the identity I. In this case,

$$\begin{aligned} \frac{\partial c_2}{\partial \alpha_k} &= i \langle \boldsymbol{z}| \left(T_-^\dagger V_k T_+^\dagger |\boldsymbol{z}'\rangle \langle \boldsymbol{z}'| - |\boldsymbol{z}'\rangle \langle \boldsymbol{z}'| T_+ V_k T_- \right) |\boldsymbol{z}\rangle \\ &= i \langle \boldsymbol{z}| \left[|\boldsymbol{z}'\rangle \langle \boldsymbol{z}'|,\ V_k \right] |\boldsymbol{z}\rangle , \end{aligned} \tag{H2}$$

where the second line holds due to the initialization $T_+ = T_-^\dagger = I$. Thereby, Eq. (H2) equals zero only when the target state commutes with the observable, which is generally not the case [11].

When T_+ and T_- are not initialized from the identity, we have

$$\frac{\partial c_2}{\partial \alpha_k} = i \langle \boldsymbol{z}| \left(T_-^\dagger V_k T_+^\dagger |\boldsymbol{z}'\rangle \langle \boldsymbol{z}'| - |\boldsymbol{z}'\rangle \langle \boldsymbol{z}'| T_+ V_k T_- \right) |\boldsymbol{z}\rangle .$$

Given the precondition that the search region is centered around the identity, meaning $d(T(\boldsymbol{\alpha}), I) = d(T_+, T_-^\dagger)$ is bounded by ϵ_f, then $|d(T_+, I) - d(T_-, I)| < d_f < d(T_+, I) + d(T_-, I)$. Thus, without normalization, we have

$$\begin{aligned} T(\boldsymbol{\alpha}) |z\rangle &= |z\rangle + \epsilon U' |z\rangle , \\ T_- |z\rangle &= |z\rangle + k_- \epsilon U' |z\rangle , \\ T_+ |z\rangle &= |z\rangle + k_+ \epsilon U' |z\rangle , \end{aligned} \tag{H3}$$

where k_+ and k_- are related to the searching region of T_+ and T_-, restricted by their depth. We can then transform the partial derivative into

$$\begin{aligned} \frac{\partial c_2}{\partial \alpha_k} &= i \langle \boldsymbol{z}| \left(V_k |\boldsymbol{z}'\rangle \langle \boldsymbol{z}'| - |\boldsymbol{z}'\rangle \langle \boldsymbol{z}'| V_k \right) |\boldsymbol{z}\rangle \\ &+ ik_+ \epsilon \langle \boldsymbol{z}| \left(V_k U_+^\dagger |\boldsymbol{z}'\rangle \langle \boldsymbol{z}'| - |\boldsymbol{z}'\rangle \langle \boldsymbol{z}'| U_+ V_k \right) |\boldsymbol{z}\rangle \\ &+ ik_- \epsilon \langle \boldsymbol{z}| \left(U_-^\dagger V_k |\boldsymbol{z}'\rangle \langle \boldsymbol{z}'| - |\boldsymbol{z}'\rangle \langle \boldsymbol{z}'| V_k U_- \right) |\boldsymbol{z}\rangle . \end{aligned} \tag{H4}$$

This shows that the major part of the partial derivative is Eq. (H2), can be preserved by maintaining a finite search region for $T(\boldsymbol{\alpha})$, thereby mitigating the effects of barren plateaus.

[1] P. Rebentrost, M. Schuld, L. Wossnig, F. Petruccione, and S. Lloyd, New Journal of Physics **21**, 073023 (2019).
[2] P. Gao, K. Li, S. Wei, J. Gao, and G. Long, Phys. Rev. A **103**, 042403 (2021).
[3] L. Cheng, P. Gao, T. Wang, and K. Li, Phys. Rev. A **109**, 032429 (2024).
[4] K. Li, S. Wei, P. Gao, F. Zhang, Z. Zhou, T. Xin, X. Wang, P. Rebentrost, and G. Long, npj Quantum Information **7**, 16 (2021).
[5] P. Gao, K. Li, S. Wei, and G.-L. Long, Science China Physics, Mechanics & Astronomy **64**, 100311 (2021).
[6] G. H. Low and I. L. Chuang, Phys. Rev. Lett. **118**, 010501 (2017).
[7] M. Motta, C. Sun, A. T. Tan, M. J. O'Rourke, E. Ye, A. J. Minnich, F. G. Brandão, and G. K. Chan, Nature Physics **16**, 205 (2020).
[8] E. Farhi, J. Goldstone, and S. Gutmann, "A quantum approximate optimization algorithm," (2014), arXiv:1411.4028 [quant-ph].
[9] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," (2017),

arXiv:1707.06347 [cs.LG].
[10] Y. Suzuki, Y. Kawase, Y. Masumura, Y. Hiraga, M. Nakadai, J. Chen, K. M. Nakanishi, K. Mitarai, R. Imai, S. Tamiya, T. Yamamoto, T. Yan, T. Kawakubo, Y. O. Nakagawa, Y. Ibe, Y. Zhang, H. Yamashita, H. Yoshimura, A. Hayashi, and K. Fujii, Quantum **5**, 559 (2021).
[11] E. Grant, L. Wossnig, M. Ostaszewski, and M. Benedetti, Quantum **3**, 214 (2019).