Winter school on Quantum Computing | Sunday 23/10/2025, BKU

Research trends in quantum computing

PhD. Vu Tuan Hai

Email: haivt@uit.edu.vn

Affiliation

Lecturer at SE-UIT-VNUHCM

Adjunct Researcher at FRIS, Tohoku University, Japan

Co-Investigator at AISeQ lab

About me

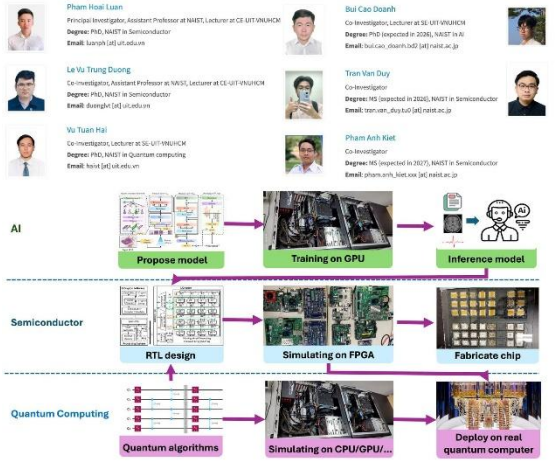
Lecturer, SE-UIT-VNUHCM: 10/2025 - now

PhD, NAIST, Japan: 9/2025

Ms, CS-UIT-VNUHCM, Vietnam: 2023

Bs, SE-UIT-VNUHCM, Vietnam: 2021

Quantum at VNUHCM

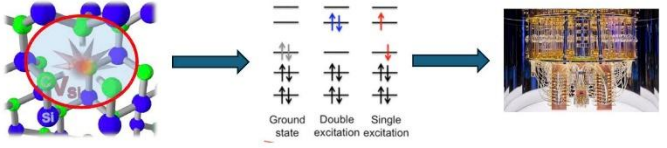


AISeQ

QuantumLab HCM US

Realistic materials

$$\hat{H}^{\mathrm{act}}_{\mathrm{ext}} = \sum_{pq\in\mathrm{act}} \bar{h}^{p}_{q}\hat{a}^{q}_{p} + \frac{1}{2}\sum_{pqrs\in\mathrm{act}} g^{pr}_{qs}\hat{a}^{qs}_{pr}$$



Correlated mean-field

Quantum downfolding

Effective Hamiltonian

Quantum computing solvers

Quantar Lab

Software Package

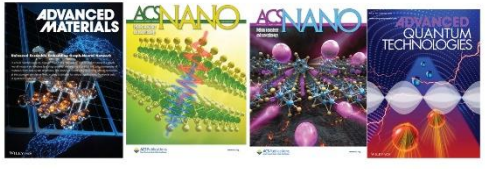
tqi

Qsun

One circuit

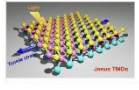
Nguyen lab

Dr. Nguyen Tuan Hung



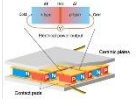
AI/ML for Materials

Keywords: Graph neural network, Diffusion model, High-throughput calculation



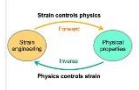
Photonics in Materials

Keywords: Raman spectra, Photovoltaics, Photocatalysis, Second harmonic generation, Excitons effect



Thermoelectricity

Keywords: Seebeck effect, Electrical conductivity, Thermal conductivity, Carrier mobility, Low-dimensional materials



Strain Engineering

Keywords: Artificial muscles, Ideal strength, Nanomechanics, Electromechanics, Optomechanics

Quantum at VIET NAM NATIONAL UNIVERSITY HO CHI MINH CITY

Research

- Quantum computing for machine learning (from 2021)
- Machine learning for quantum computing
- Hardware accelerator
- Functional programming

Paper: journal [14], conference [28], book [2] (September 2025)

Fund: NAIST Granite, Unitary Fund, …

Reviewer: SOICT, SoftwareX, Quantum Information Processing, …

See more: https://aiseqlab.github.io/team/haivt

Outline

1. Quantum optimization
2. Quantum machine learning
3. Quantum simulation

1. Quantum optimization

1.1. Parameterized quantum circuit (PQC)

1.2. Circuit ansatz

1.3. Compilation

1.4. Gradient and optimizer

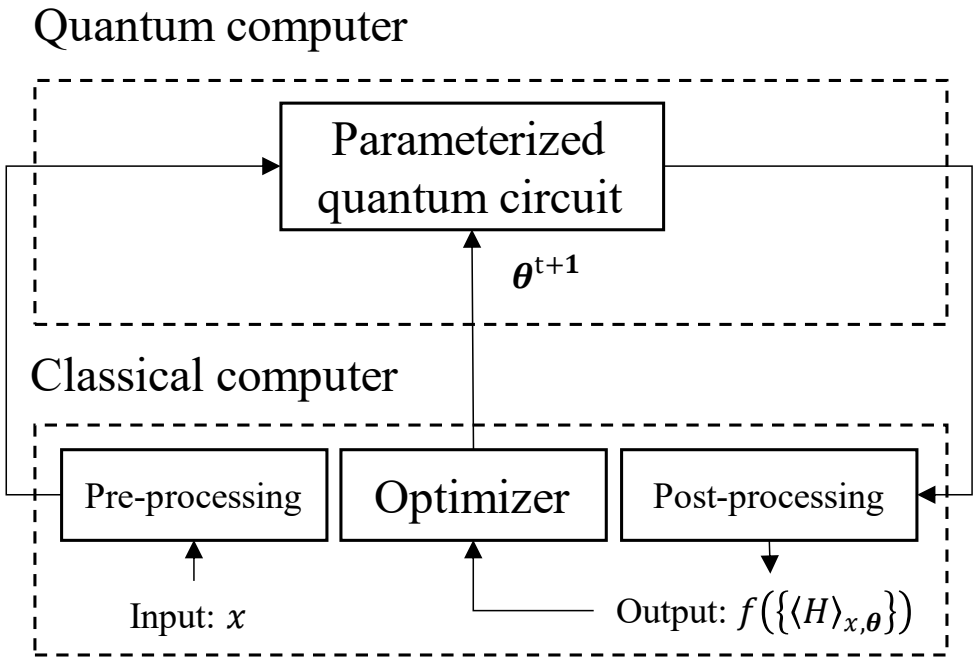
1.5. Encoding

1.6. Quantum optimization

1.7. Additional informations

1. Quantum optimization

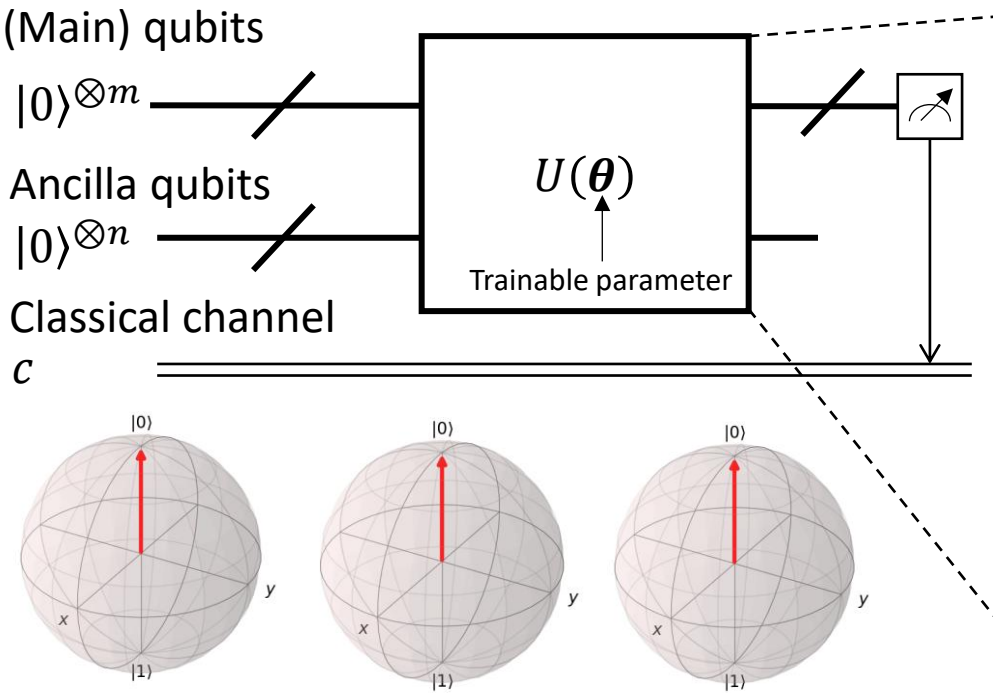
Hybrid quantum - classical model



Hybrid model [1.1]

[1.1] M. Schuld and F. Petruccione, Machine learning with quantum computers (Springer, 2021)

1.1. Parameterized quantum circuit (PQC)

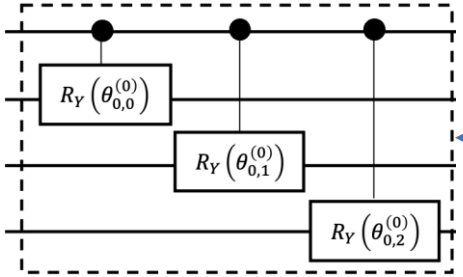


Gate $R_i(\theta)$ rotate state θ^o about the i-axis where $i \in \{X, Y, Z\}$ and $\theta \in [0, 2\pi]$

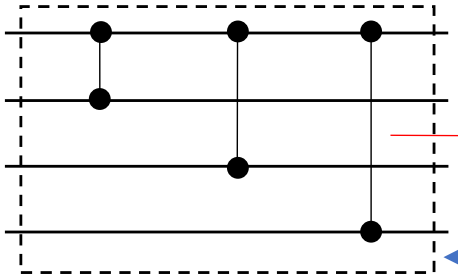
[1.2] Benedetti, M et al, Parameterized quantum circuits as machine learning models. Quantum Science and Technology, 4(4), 043001 (2019)

1.2. Circuit ansatz

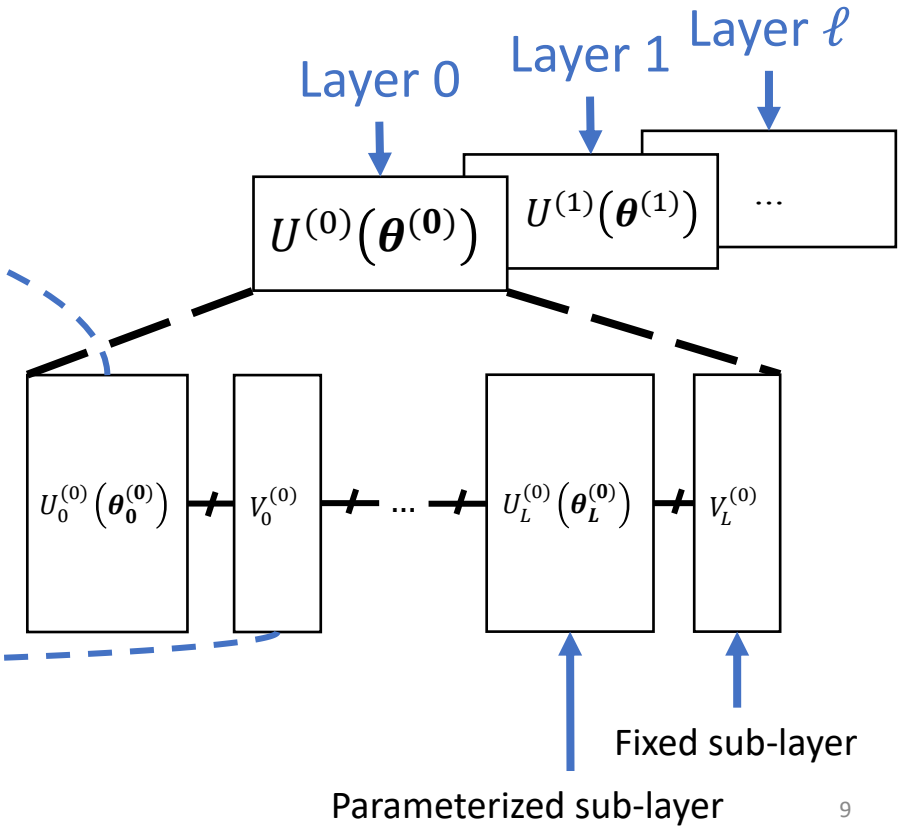
Structures



Example $U_k^{(j)}(.)$: **Graph – star [16]**



Example $V_k^{(j)}$: **Control – Z [16]**

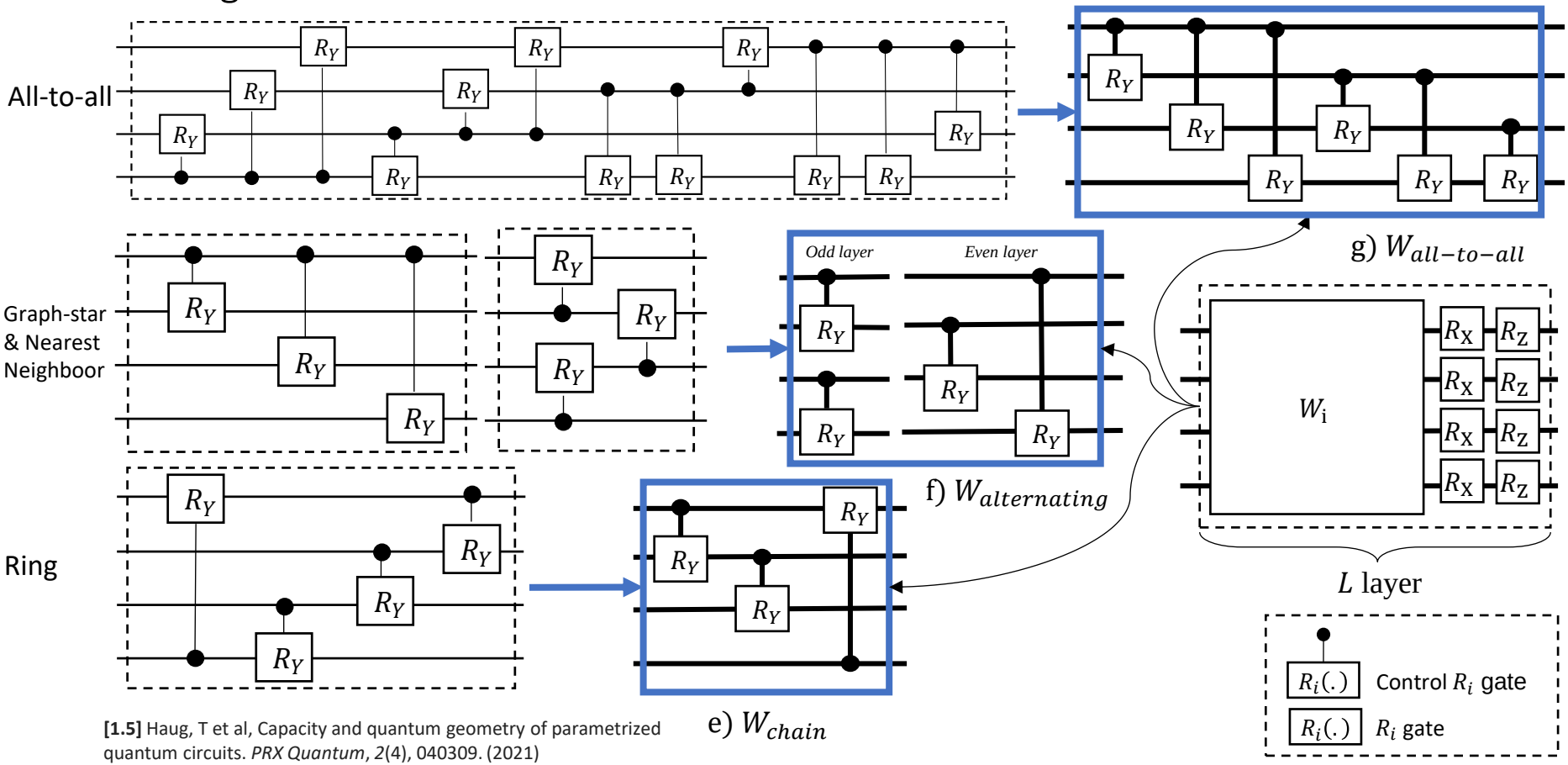


[1.3] Haug, T et al, Capacity and quantum geometry of parametrized quantum circuits. *PRX Quantum*, *2*(4), 040309. (2021)

[1.4] https://pennylane.ai/qml/glossary/circuit_ansatz

1.2. Circuit ansatz

Existing ansatzes

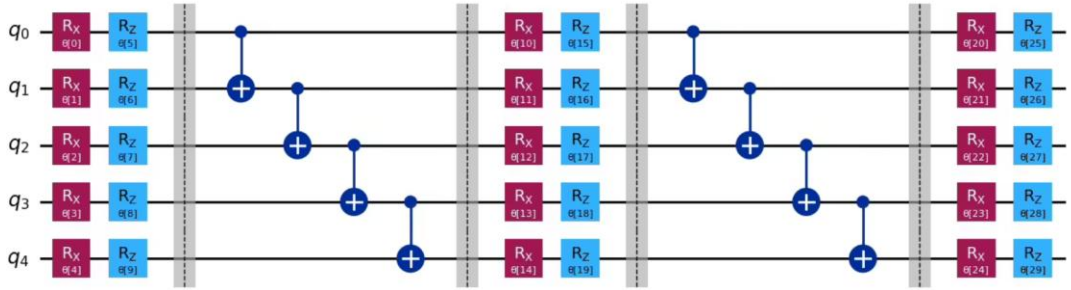


[1.5] Haug, T et al, Capacity and quantum geometry of parametrized quantum circuits. *PRX Quantum*, *2*(4), 040309. (2021)

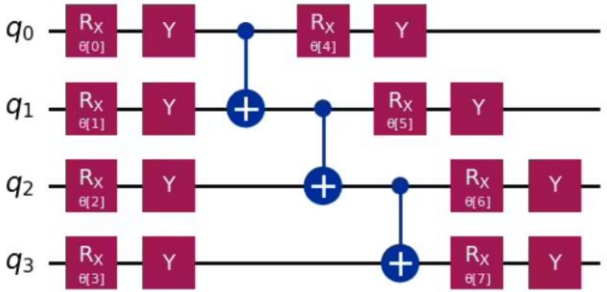
1.2. Circuit ansatz

Existing ansatzes

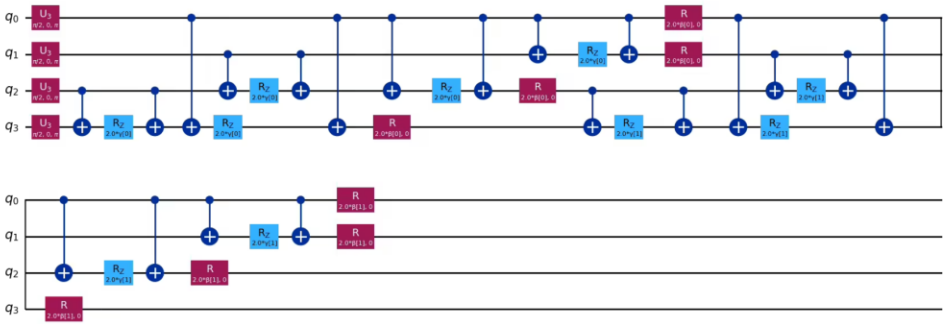
N local → Two local ansatz



Efficient SU2 ansatz



(Domain-specific) QAOA ansatz



[**1.6**] https://quantum.cloud.ibm.com/learning/en/courses/variational-algorithm-design/ansaetze-and-variational-forms

1.2. Circuit ansatz

Using descriptors to measure PQC

How do we know which ansatz is better?

Expressibility

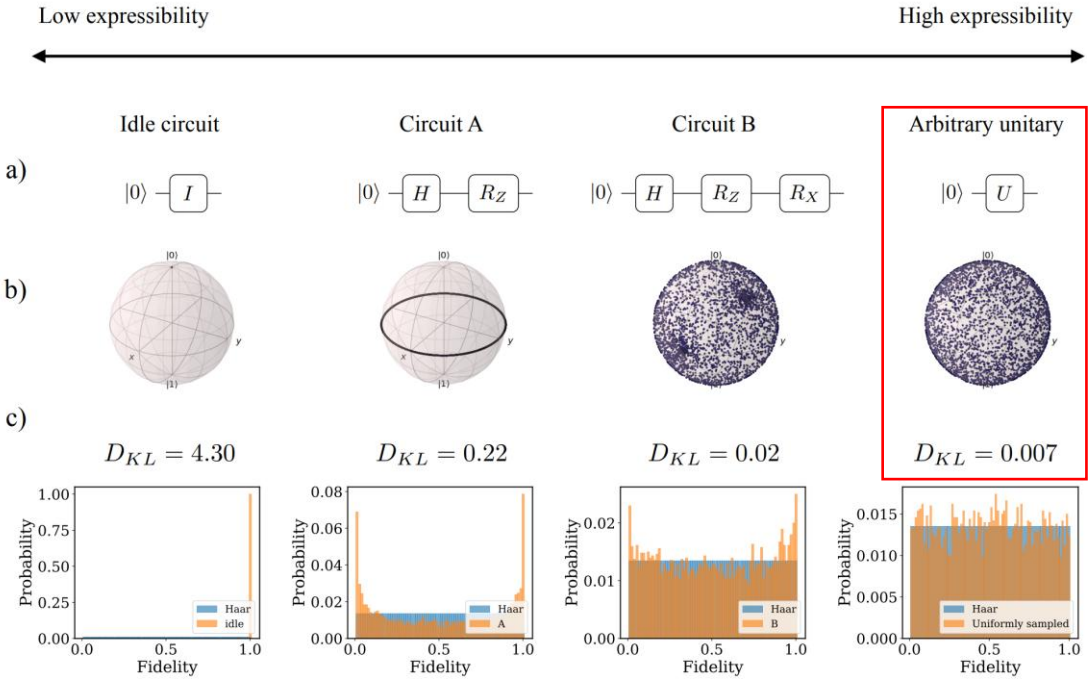
$$Expr = D_{KL}(\hat{P}_{PQC}(F;\theta)||P_{Haar}(F)),$$

where $\hat{P}_{PQC}$ is the estimated probability distribution of fidelities resulting from sampling states from a PQC.

Entangling capability

$$Ent = \frac{1}{|S|}\sum_{\boldsymbol{\theta}_i \in S} Q\left(|\psi_{\boldsymbol{\theta}_i}\rangle\right)$$

where $S = \{\boldsymbol{\theta}_i\}$ is the set of sampled circuit parameter vectors and Q is the Meyer-Wallach entanglement measure



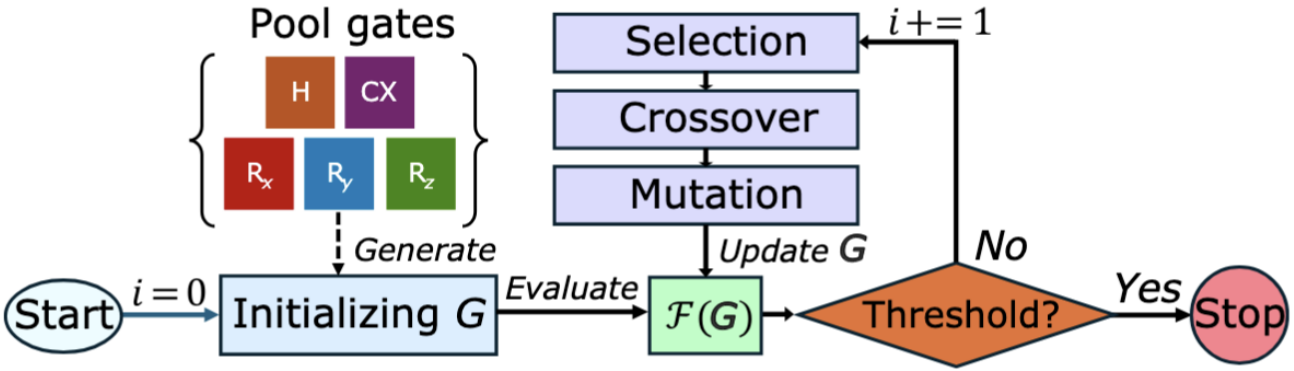
[1.7] Sim, S., Johnson, P. D., & Aspuru-Guzik, A. (2019). Expressibility and entangling capability of parameterized quantum circuits for hybrid quantum-classical algorithms. Advanced Quantum Technologies, 2(12), 1900070.

1.2. Circuit ansatz

Searching problem

Question: Can we automatically find the best ansatz?

A easiest approach is the gradient-free method, such as Bayesian opt **[1.8]** and genetic algorithm (GA) **[1.9].**



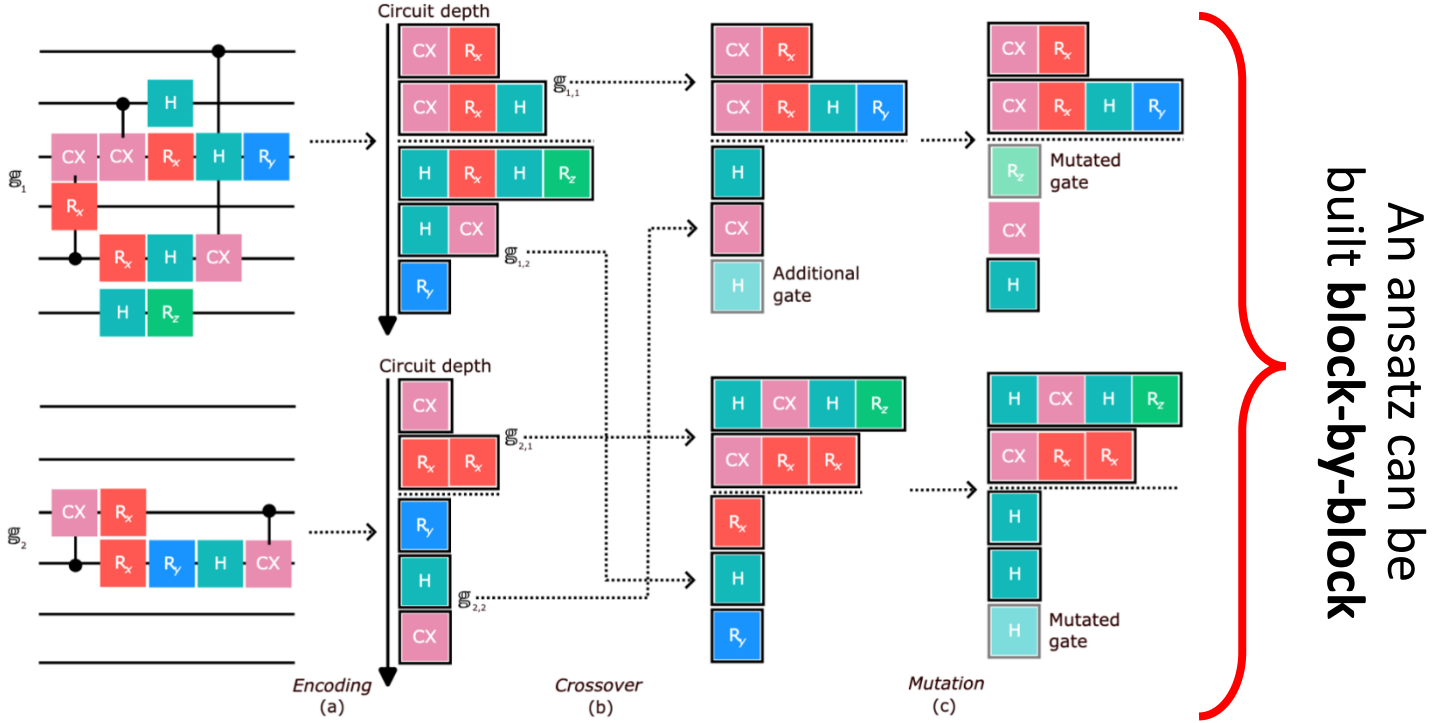
[1.8] Duong, T., Truong, S. T., Tam, M., Bach, B., Ryu, J. Y., & Rhee, J. K. K. (2022). Quantum neural architecture search with quantum circuits metric and bayesian optimization. *ICML 2022* Workshop AI4Science.

[1.9] Hai, V. T., Viet, N. T., Urbaneja, J., Linh, N. V., Tran, L. N., & Ho, L. B. (2024). Multi-target quantum compilation algorithm. *Machine Learning: Science and Technology*, *5*(4), 045057.

1.2. Circuit ansatz

Searching problem

Question: How does an ansatz construct?



[1.10] Duc (2025). Flexible Genetic Algorithm for Quantum Support Vector Machines (unpublished).

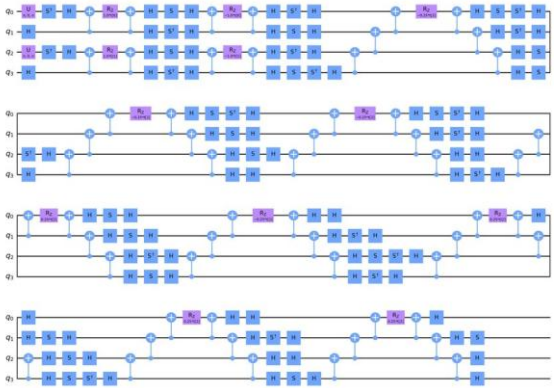
1.2. Circuit ansatz

Searching problem

Let's take a look on UCCSD ansatz

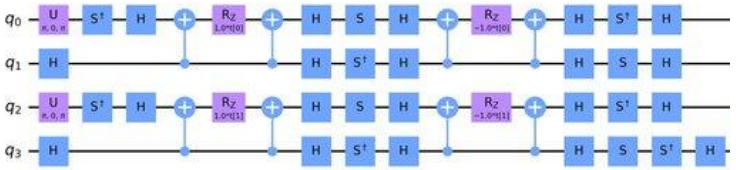
We map from $U_{UCCSD}(\boldsymbol{\theta}) = \prod e^{\theta_k}(\mathrm{A}_k - A_k^\dagger)$

where each $e^{\theta_k}(\mathrm{A}_k - A_k^\dagger)$ is equivalent to a series of CX and R_i gate



$\Rightarrow$

Pruning through optimization process (ADAPT-VQE) [*]



An adaptive ansatz for each systems

[1.11] Grimsley, H. R., Economou, S. E., Barnes, E., & Mayhall, N. J. (2019). An adaptive variational algorithm for exact molecular simulations on a quantum computer. *Nature communications*, *10*(1), 3007.

1.3. Compilation

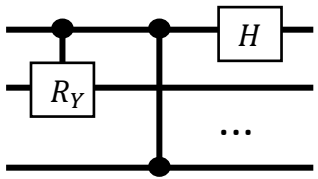
There is a large gap between algorithms and set of instructions we can perform on real qc

Problem 1 (a): The native gate set is diff

No CX, R_X or CR_Y $\{CZ, X, R_Z, SX, ID\}$ IBM series

No CX, R_Y or CR_Y $\{R_X, R_Z, ISWAP\}$ Rigetti Ankaa-3

If we have this example circuit:



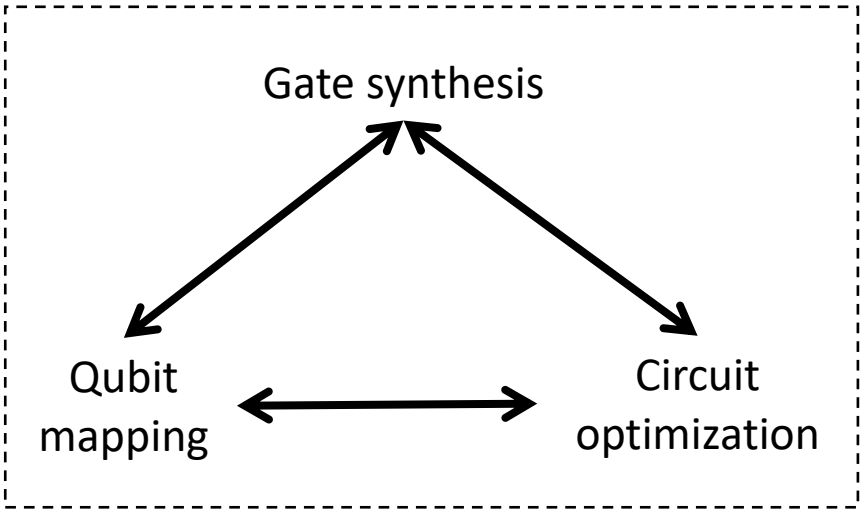
$\equiv$ $U_{\{u\}}$

$\Downarrow$ Compile

$U'_{\{u'\}}$

$u = \{R_{X,Y,Z}, CX\}$

$u' = \{R_{X,Z}, ISWAP\}$

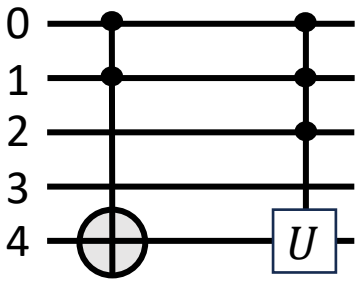


[1.12] Bäumer, E., Tripathi, V., Wang, D. S., Rall, P., Chen, E. H., Majumder, S., ... & Minev, Z. K. (2024). Efficient long-range entanglement using dynamic circuits. *PRX Quantum*, *5*(3), 030339.

1.3. Compilation

There is a large gap between algorithms and set of instructions we can perform on real qc

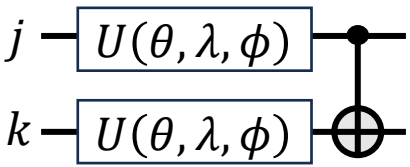
Problem 1 (b): Some gates are not exist

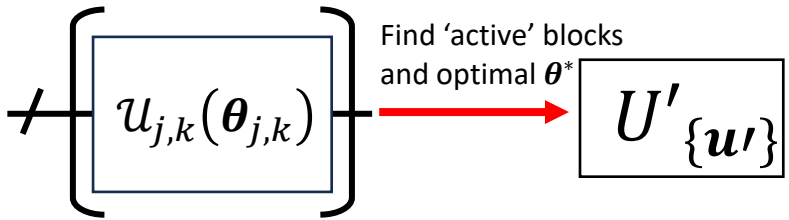


Toffoli gate Multi-control gate

- Native gate set $\{\boldsymbol{u}'\}$
- Qubit layout

Number of qubits	2	3	4	5	n_{CX}
Iterative unentangling [21]	8	62	344	1642	$2 \cdot 4^n - (2n+3) \cdot 2^n + 2n$
Givens rotations [20, 27]	4	64	536	4156	$\approx 8 \cdot 4 \cdot 4^n$
Recursive CSD [22]	14	92	504	2544	$\frac{1}{2} \cdot n \cdot 4^n - \frac{1}{2} \cdot 2^n$
Recursive CSD (optimized) [23]	4	26	118	494	$\frac{1}{2} \cdot 4^n - \frac{1}{2} \cdot 2^n - 2$
QSD [21, 27]	6	36	168	720	$\frac{3}{4} \cdot 4^n - \frac{3}{2} \cdot 2^n$
QSD (optimized)[21]	3	20	100	444	$(23/48) \cdot (4^n) - \frac{3}{2} \cdot 2^n + \frac{4}{3}$
Sequential optimization[a]	**3**	**15**	**63**	**267**	Not available
Theoretical Lower Bounds[18]	3	14	61	252	$\frac{1}{4}(4^n - 3n - 1)$





SQUANDER package **[1.13]**

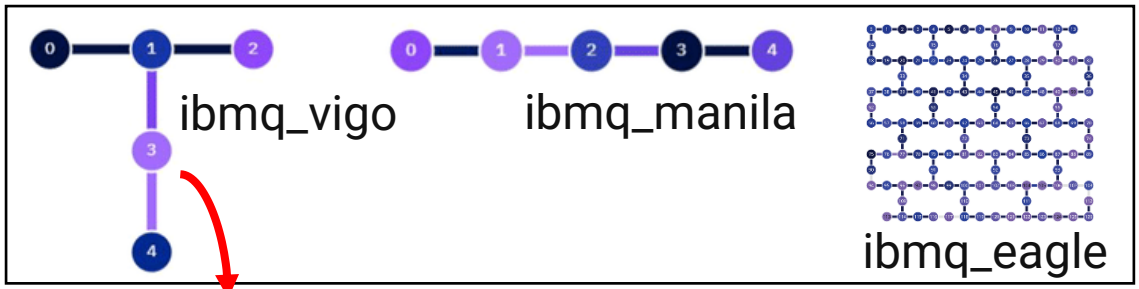
Note that there is a theoretical limit (lower bound) for synthesize circuit!

[**1.13**] Rakyta, P., & Zimborás, Z. (2022). Approaching the theoretical limit in quantum gate decomposition. *Quantum*, *6*, 710.

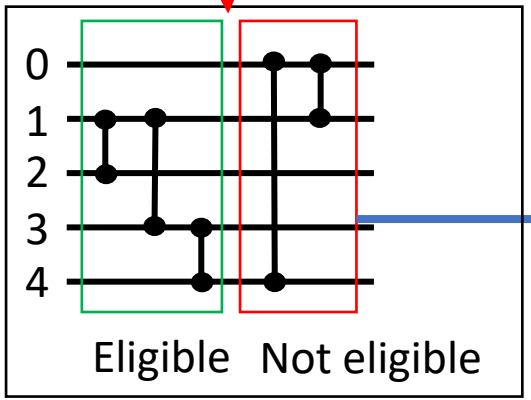
1.3. Compilation

There is a large gap between algorithms and set of instructions we can perform on real qc

Problem 2: Not all qubits are connected together



Limited topologies



Unlimited topologies

Transpile

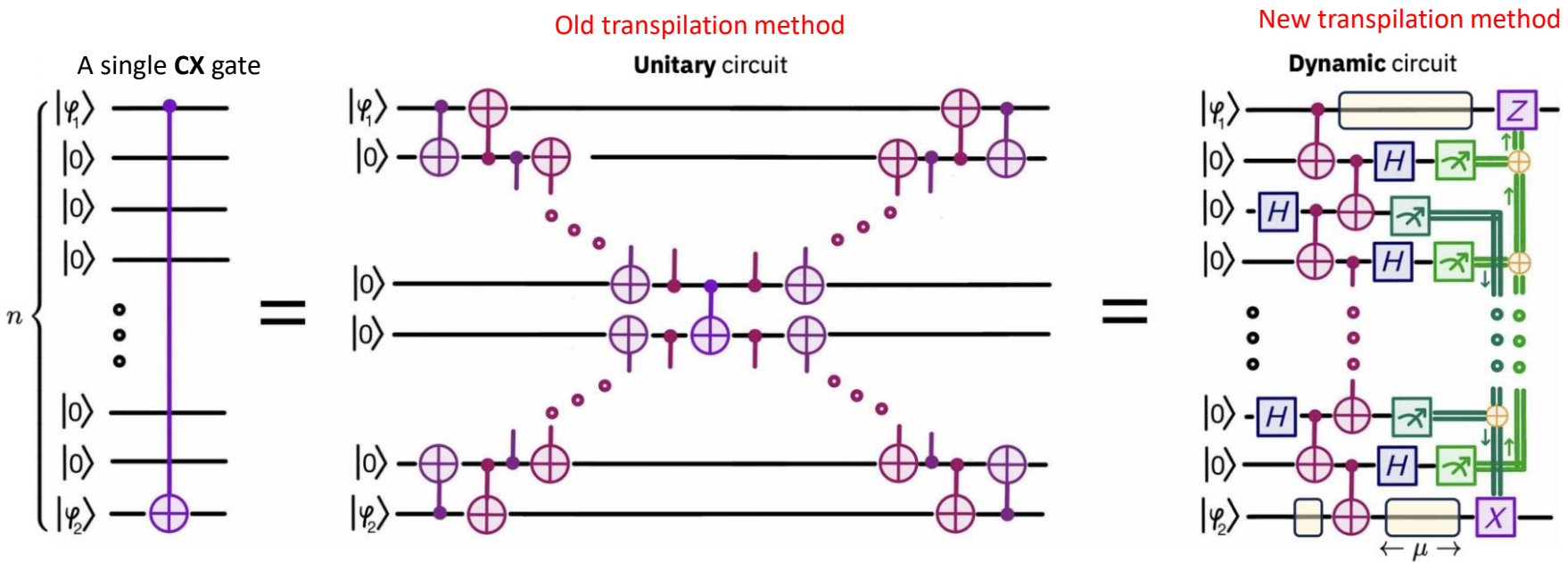
Dynamic circuit [*]

[1.14] https://quantum.cloud.ibm.com/computers

[1.15] Bäumer, E., Tripathi, V., Wang, D. S., Rall, P., Chen, E. H., Majumder, S., ... & Minev, Z. K. (2024). Efficient long-range entanglement using dynamic circuits. *PRX Quantum*, *5*(3), 030339.

1.3. Compilation

Problem 2: Not all qubits are connected together



[1.16] Bäumer, E., Tripathi, V., Wang, D. S., Rall, P., Chen, E. H., Majumder, S., ... & Minev, Z. K. (2024). Efficient long-range entanglement using dynamic circuits. *PRX Quantum*, *5*(3), 030339.

1.4. Gradient and optimizer

Considering a PQC:

$$U(\boldsymbol{\theta}) = U_{m-1}U_{m-2}\dots U_0$$

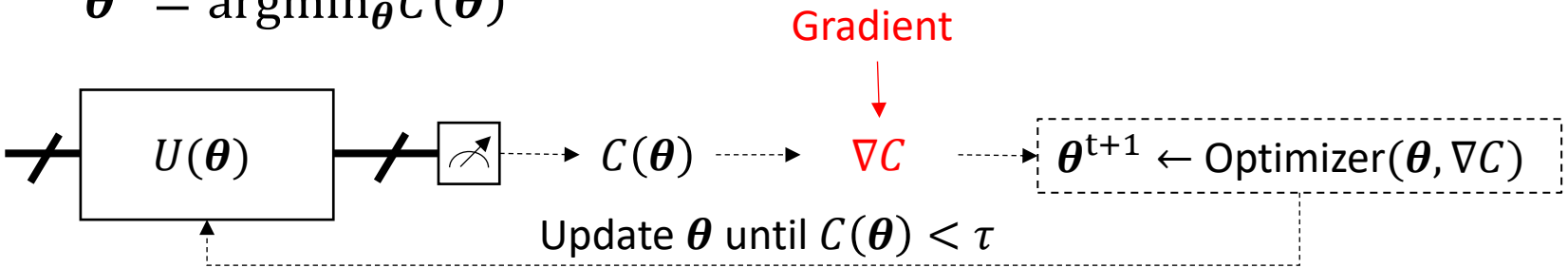
can be splitted into m sub-circuits with $|\boldsymbol{\theta}| = m$

and cost function:

$$C(\boldsymbol{\theta}) = \langle \mathbf{0} | U^{\dagger}(\boldsymbol{\theta})\hat{B}U(\boldsymbol{\theta}) | \mathbf{0} \rangle$$

We want to find:

$$\boldsymbol{\theta}^* = \mathrm{argmin}_{\boldsymbol{\theta}} C(\boldsymbol{\theta})$$



1.4. Gradient

Computing PQC's gradient is the key in quantum optimization:

$$\nabla C = [\partial_{\theta_0} C \quad \dots \quad \partial_{\theta_{m-1}} C]^\intercal$$

The partial derivative can be computed through parameter-shift rule **[1.17]**:

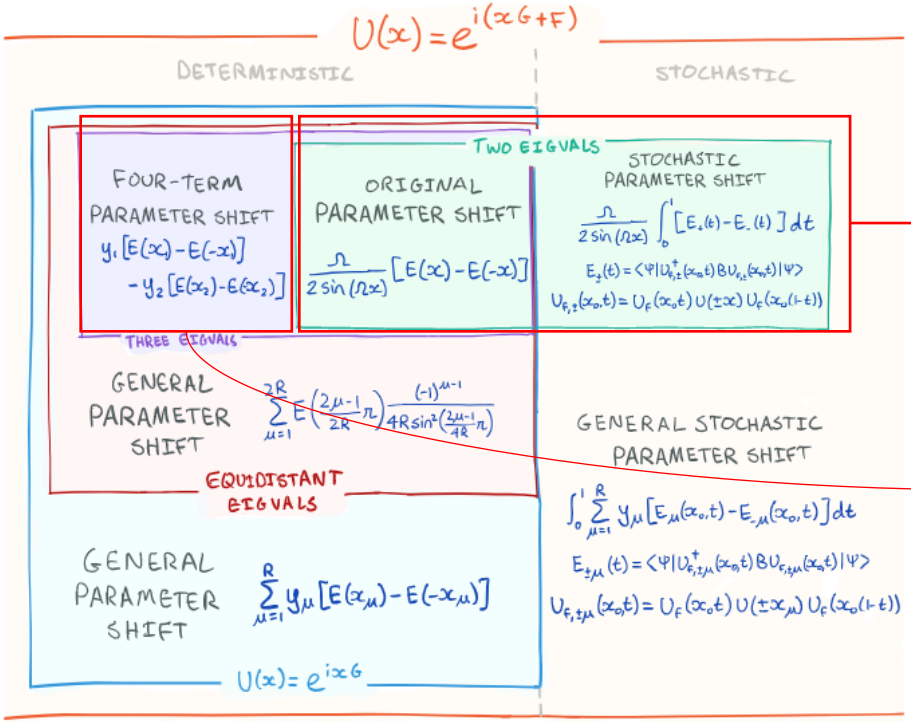
$$\frac{\partial C}{\partial \theta_j} = \sum_{k=1}^{R} d_k \left(C\left(\theta_j^{+\epsilon_k}\right) - C\left(\theta_j^{-\epsilon_k}\right) \right)$$

and $C\left(\theta_j^{+\epsilon_k}\right) \equiv C(\boldsymbol{\theta} + \mathbf{e_j}\epsilon_k)$, $\mathbf{e_j}$ is the j^{th} unit vector, ϵ_k is the shift-value

[1.17] D. Wierichs, J. Izaac, C. Wang, and C. Y.-Y. Lin, "General parameter-shift rules for quantum gradients," Quantum, vol. 6, p. 677, Mar. 2022

1.4. Gradient

Parameter-shift rule



Commonly, we only use 2term-PSR, in case R_i.

$$\partial_{\theta_j} C = \frac{1}{2}\left[C\left(\theta_j^{+\frac{\pi}{2}}\right) - C\left(\theta_j^{-\frac{\pi}{2}}\right)\right]$$

4term-PSR is used in case control-R_i.

$$\partial_{\theta_j} C = d^+\left[C\left(\theta_j^{+\alpha}\right) - C\left(\theta_j^{-\alpha}\right)\right] - d^-\left[C\left(\theta_j^{+\beta}\right) - C\left(\theta_j^{-\beta}\right)\right]$$

Take $2m/4m$ quantum evaluations for a single gradient!

$$f(x) = \sin x, f'(x) = \frac{1}{2}\left[\sin\left(x+\frac{\pi}{2}\right) - \sin\left(x-\frac{\pi}{2}\right)\right] = \frac{1}{2}[\cos(x)+\cos(x)] = \cos(x)$$

1.4. Gradient

Parameter-shift rule with backpropagation

$$\mathcal{U}_{i:j} = U_j U_{j-1} \dots U_i$$

Quantum backpropagation [**1.18,1.19**] requires only a ***single*** quantum evaluations

For any pair $\left\{U\left(\boldsymbol{\theta}_j^{\pm\epsilon_k}\right)\right\}$, there is a duplication:

$$U\left(\boldsymbol{\theta}_j^{+\epsilon_k}\right) = \mathcal{U}_{j+1:m-1} U\left(\theta_j^{+\epsilon_k}\right) \mathcal{U}_{0:j-1}$$

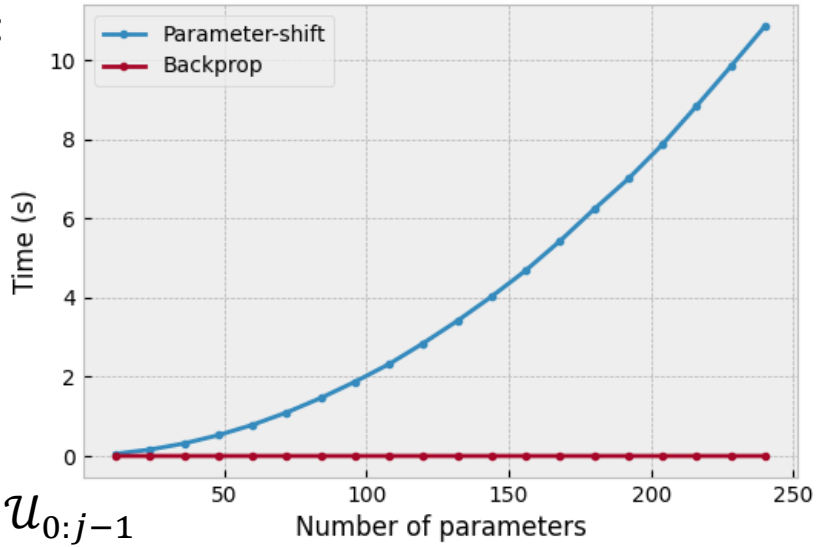
$$U\left(\boldsymbol{\theta}_j^{-\epsilon_k}\right) = \mathcal{U}_{j+1:m-1} U\left(\theta_j^{-\epsilon_k}\right) \mathcal{U}_{0:j-1}$$

Furthermore:

$$U\left(\boldsymbol{\theta}_{j+1}^{\pm\epsilon_k}\right) = \mathcal{U}_{j+2:m-1} U\left(\theta_{j+1}^{\pm\epsilon_k}\right) \mathcal{U}_{0:j}$$

We already know:

$$\mathcal{U}_{j+1:m-1} = U_{j+1}\mathcal{U}_{j+2:m-1} \text{ and } \mathcal{U}_{0:j} = U_j \mathcal{U}_{0:j-1}$$



[1.18] https://pennylane.ai/qml/demos/tutorial_backprop

[1.19] Hai, V. T., Luan, P. H., & Nakashima, Y. (2024, October). Efficient Parameter-Shift Rule Implementation for Computing Gradient on Quantum Simulators. *ATC* (pp. 449-454). IEEE.

1.4. Gradient

Parameter-shift rule with backpropagation

j \ i	0	1	2	3	...	$m-1$	$\|\psi_j\rangle$
0	U_0	$\mathbb{I}_{2^n}$	$\mathbb{I}_{2^n}$	$\mathbb{I}_{2^n}$	...	$\mathbb{I}_{2^n}$	$\|\psi_0\rangle$
1	$\mathcal{U}_{0:1}$	U_1	$\mathbb{I}_{2^n}$	$\mathbb{I}_{2^n}$	...	$\mathbb{I}_{2^n}$	$\|\psi_1\rangle$
2	$\mathcal{U}_{0:2}$	$\mathcal{U}_{1:2}$	U_2	$\mathbb{I}_{2^n}$	...	$\mathbb{I}_{2^n}$	$\|\psi_2\rangle$
3	$\mathcal{U}_{0:3}$	$\mathcal{U}_{1:3}$	$\mathcal{U}_{2:3}$	U_3	...	$\mathbb{I}_{2^n}$	$\|\psi_3\rangle$
...	...	...	...	...	...	...	...
$m-1$	U	$\mathcal{U}_{1:m-1}$	$\mathcal{U}_{2:m-1}$	$\mathcal{U}_{3:m-1}$	...	U_{m-1}	$\|\psi_{m-1}\rangle$

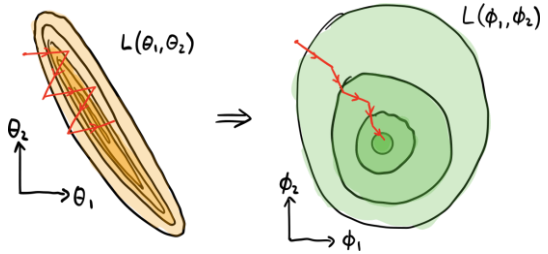
Table. Look up table (LUT) for $\{U_{i:j}\}_{i,j} \in [0, m-1]$. Red and Blue elements are required for MM-based and state-based simulators, respectively. Violet elements are required for both.

To fill this LUT, only take $\sum_{j=0}^{m-1} j = \frac{(m-1)\times m}{2}$ matrix multiplication

To achive $\left\{U\left(\boldsymbol{\theta}_j^{\pm\epsilon_k}\right)\right\}$, this method only take $\frac{(m-1)\times m}{2} + 4R < \left(2 \times R \times (m-1)\right) \times m$

1.4. Gradient

Tensor metric



The same idea as Hessian matrix

We can use Information Matrix (QFIM) F [**1.20**] or Tensor metric G [**1.21**] to 'transform' the normal gradient to the natural gradient:

$$F_{j,k}(\theta) = 4\Re e\left[\langle \partial_{\theta_j}\psi|\partial_{\theta_k}\psi\rangle - \langle \partial_{\theta_j}\psi|\psi\rangle\langle\psi|\partial_{\theta_k}\psi\rangle\right]$$

$$g^{(\ell)}_{j,k} = \langle\psi_{(\ell)-1}\left|K_jK_k\right|\psi_{(\ell)-1}\rangle - \langle\psi_{(\ell)-1}\left|K_j\right|\psi_{(\ell)-1}\rangle\langle\psi_{(\ell)-1}|K_k|\psi_{(\ell)-1}\rangle$$

This 'new' gradient can be used as normal gradient, for example SGD:

$$\boldsymbol{\theta}^{t+1} \leftarrow \boldsymbol{\theta}^{t} - \alpha\nabla_{\boldsymbol{\theta}}C$$

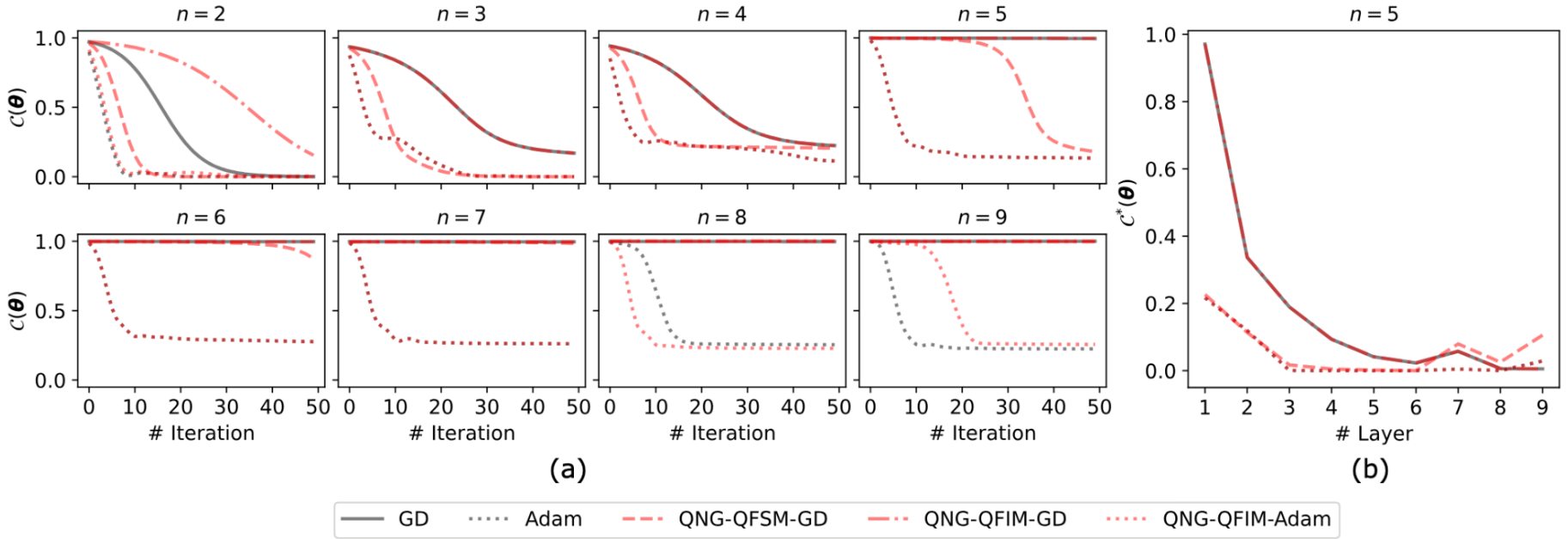
Pseudo-inverse of F

$$\boldsymbol{\theta}^{t+1} \leftarrow \boldsymbol{\theta}^{t} - \alpha(F^{+}\nabla_{\boldsymbol{\theta}}C)$$

[1.20] Meyer, J. J. (2021). Fisher information in noisy intermediate-scale quantum applications. *Quantum*, *5*, 539.
[1.21] Stokes, J., Izaac, J., Killoran, N., & Carleo, G. (2020). Quantum natural gradient. *Quantum*, *4*, 269.

1.4. Optimizer

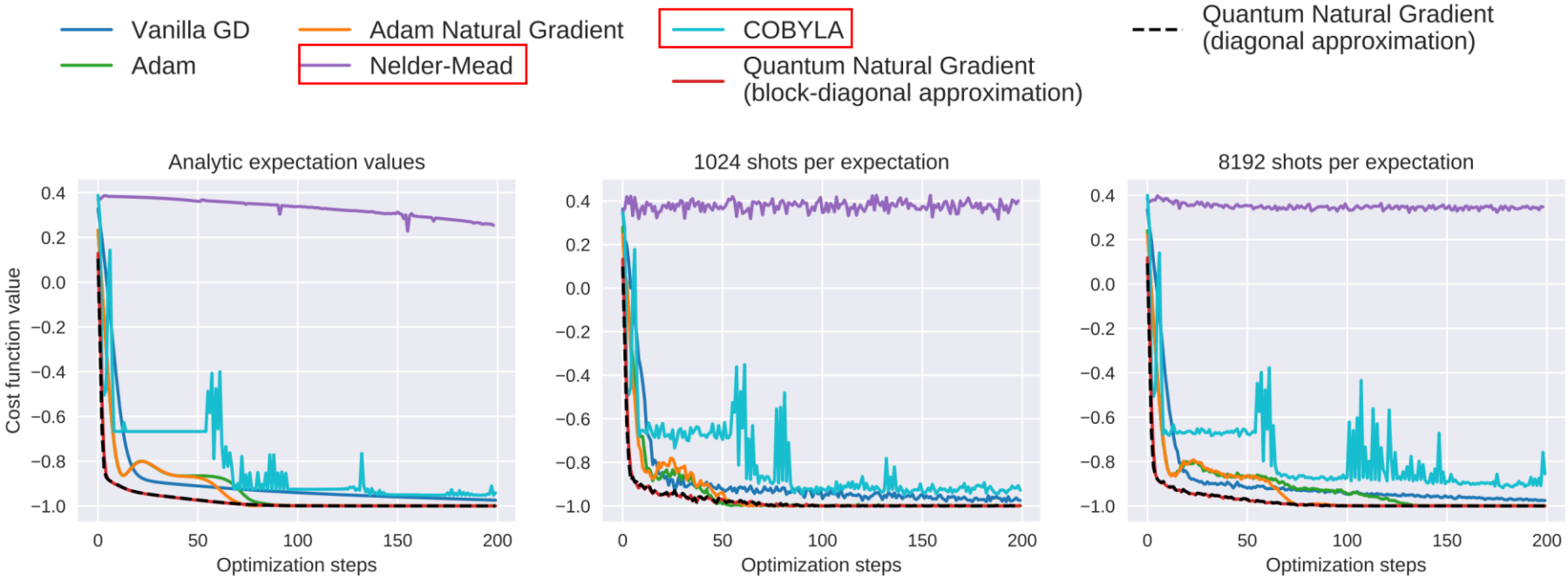
We can use classical optimizers such as Adam in quantum optimization problems.



[1.22] Hai, V. T., Luan, P. H., & Nakashima, Y. (2024, November). Benchmarking Classical and Quantum Optimizers for Quantum Simulator. In *2024 Twelfth International Symposium on Computing and Networking (CANDAR)* (pp. 238-244). IEEE.

1.4. Optimizer

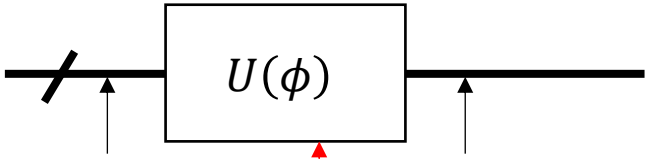
Note that we can also use gradient-free optimizers: Nelder-Mead, COBYLA [**1.23**]



[1.23] Stokes, J., Izaac, J., Killoran, N., & Carleo, G. (2020). Quantum natural gradient. *Quantum*, *4*, 269.

1.5. Encoding

Quantum state preparation



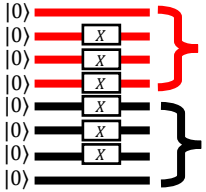
$|\psi_0\rangle = |0\rangle^{\otimes n}$ $|\psi'\rangle = U(\phi)|\psi_0\rangle$ with $|\psi'\rangle$ is the prepared/embedded state

How to prepare $U(\phi)$?

For example: $|\psi'\rangle = \begin{bmatrix} 1/\sqrt{5} & 2/\sqrt{5} \end{bmatrix}$

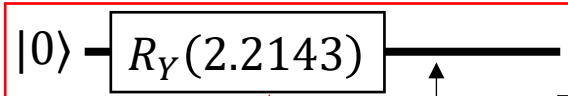
$0.447_{10} \approx 0.0111_2$

$0.8944_{10} \approx 0.1110_2$



$\longrightarrow M2^n$ qubits

Basic encoding



$|\psi'\rangle = \begin{bmatrix} 1/\sqrt{5} \\ 2/\sqrt{5} \end{bmatrix}$ $\longrightarrow n$ qubits

How to find ϕ?

Amplitude encoding

[1.24] Schuld, M., & Petruccione, F. (2018). Supervised learning with quantum computers. *Quantum science and technology*, *17*.

1.5. Encoding

Diff encoders

Encoder / Feature	Threshold Encoding - **TE**	Flexible Representation - **FRQI**	Novel enhanced quantum - **NEQR [1.25]**	Amplitude Encoding - **AE [1.26] & Quantum RAM [1.27]**	Divide – and - Conquer Encoder - **DCE [1.28]**	Universal Compilation – based variational quantum- **UC – VQA [1.29]**
#qubit	$\mathcal{O}(2^{2n})$	$\mathcal{O}(2n+1)$	$\mathcal{O}(2n+1)$	$\mathcal{O}(n)$	$\mathcal{O}(2^{2n})$	$\boldsymbol{\mathcal{O}(n)}$
Circuit depth	$\mathcal{O}(2^{2n})$	$\mathcal{O}(2^{4n})$	$\mathcal{O}(2^{2n})$	$\mathcal{O}(2^{2n})$	$\mathcal{O}(k)$	$\boldsymbol{\mathcal{O}(\mathrm{poly}(n))} + \boldsymbol{\mathcal{O}(\mathrm{poly}(n))}$

List of encoders

Circuit depth

Resource for $\boldsymbol{\theta_\phi}$

[1.25] Zhang, Y., Lu, K., Gao, Y., & Wang, M, NEQR: a novel enhanced quantum representation of digital images. Quantum information processing, 12, 2833-2860 (2013).

[1.26] M. Weigold et al, Data encoding patterns for quantum computing, in Proceedings of the 27th Conference on Pattern Languages of Programs PLoP '20, (The Hillside Group, USA, 2022).

[1.27] V. Giovannetti et al, Quantum random access memory, Phys. Rev. Lett. 100 (Apr 2008) p. 160501.

[1.28] I. F. Araujo et al, A divide-and-conquer algorithm for quantum state preparation, Scientific Reports 11 (Mar 2021) p. 6329.

[1.29] Hai, V. T., & Ho, L. B. (2023). Universal compilation for quantum state tomography. Scientific Reports, 13(1), 3750

1.5. Encoding

Time-evolution Unitary approximation

Implementing a time evolution unitary in a quantum circuit is difficult, so we use a Trotter-Suzuki decomposition to perform approximate time evolution. We then have:

Time-ordering operator

Target

$$U(T) = \mathcal{T} \exp\left[-i \int_0^T H(t)\, dt\right] \xrightarrow{discretization} \approx \mathcal{T} \exp\left[-i \sum_{k=0}^{T/\Delta t} H(k\Delta t)\Delta t\right]$$

$$\xrightarrow{\mathcal{T}} e^{-i\beta_n H_d \Delta t}\, e^{-i\beta_n H_c \Delta t} \ldots e^{-i\beta_1 H_d \Delta t} e^{-i\beta_1 H_c \Delta t}$$

Implemented by rotation gates

$$= U_d(\beta_n) U_C \ldots U_d(\beta_1) U_C$$

→ Quantum circuit

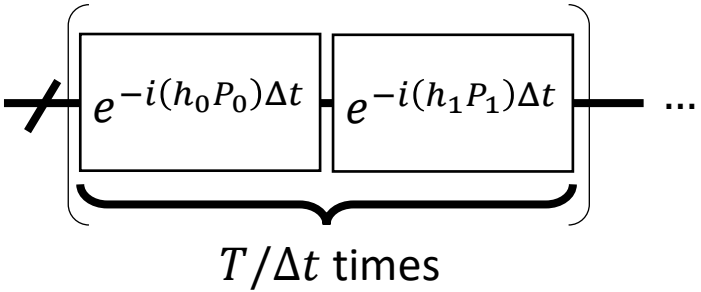
1.5. Encoding

Time-evolution Unitary approximation

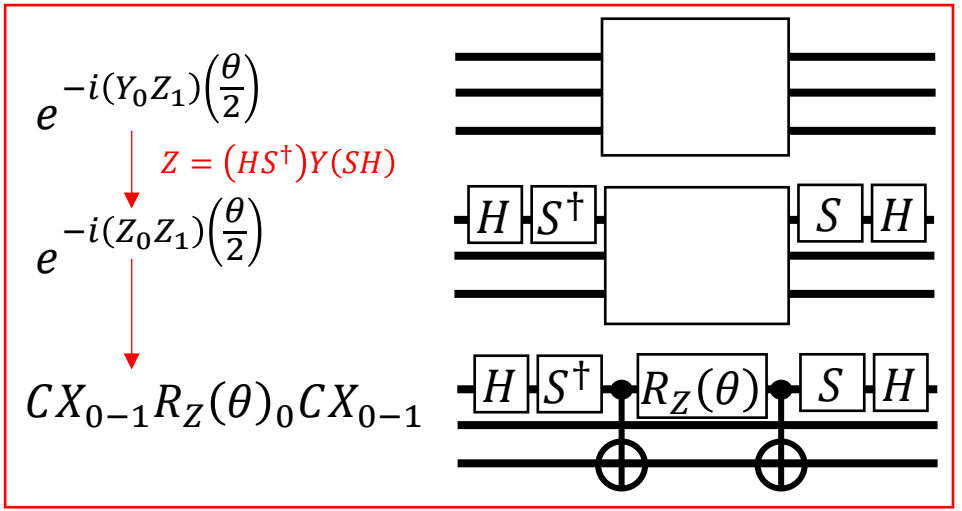
We already know that:

$H = \sum h_j P_j$ with coefficient $h_j \in \mathbb{C}$ and P_j is the n - qubit Pauli string

$$U(T) = \prod_{k=1}^{T/\Delta t} \prod_{j} e^{-i(h_j P_j)\Delta t}$$



Example



[1.30] qml.ApproxTimeEvolution

1.6. Quantum optimization

Considering a PQC:

$$U(\boldsymbol{\theta}) = U_{m-1}U_{m-2}\dots U_0$$

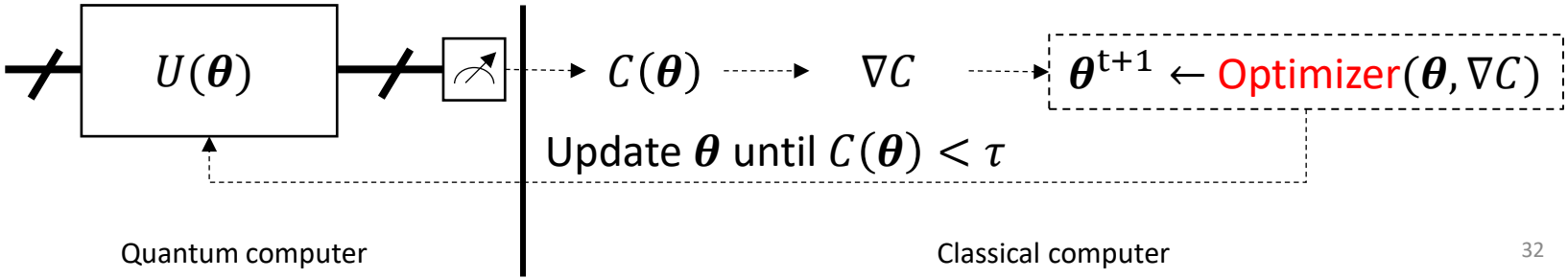
can be splitted into m sub-circuit with $|\boldsymbol{\theta}| = m$ → Trainable parameters

and cost function:

$$C(\boldsymbol{\theta}) = \langle \mathbf{0}|U^{\dagger}(\boldsymbol{\theta})\hat{B}U(\boldsymbol{\theta})|\mathbf{0}\rangle$$ → Objective associated with expectation values

We want to find:

$$\boldsymbol{\theta}^* = \text{argmin}_{\boldsymbol{\theta}} C(\boldsymbol{\theta})$$ → Optimal parameters

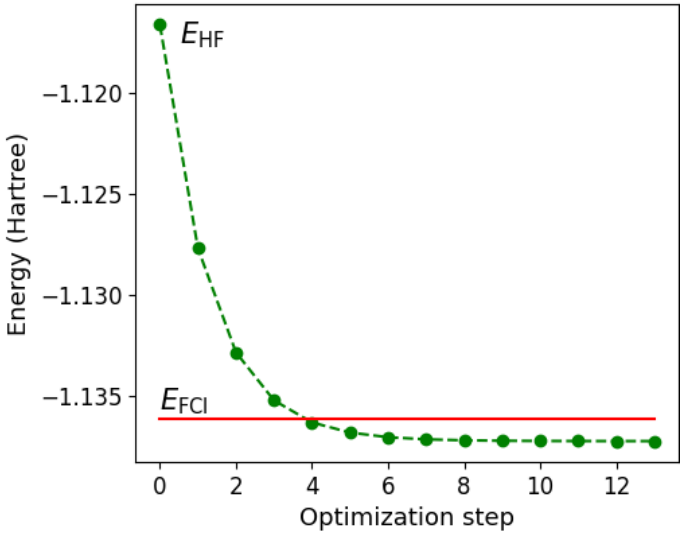


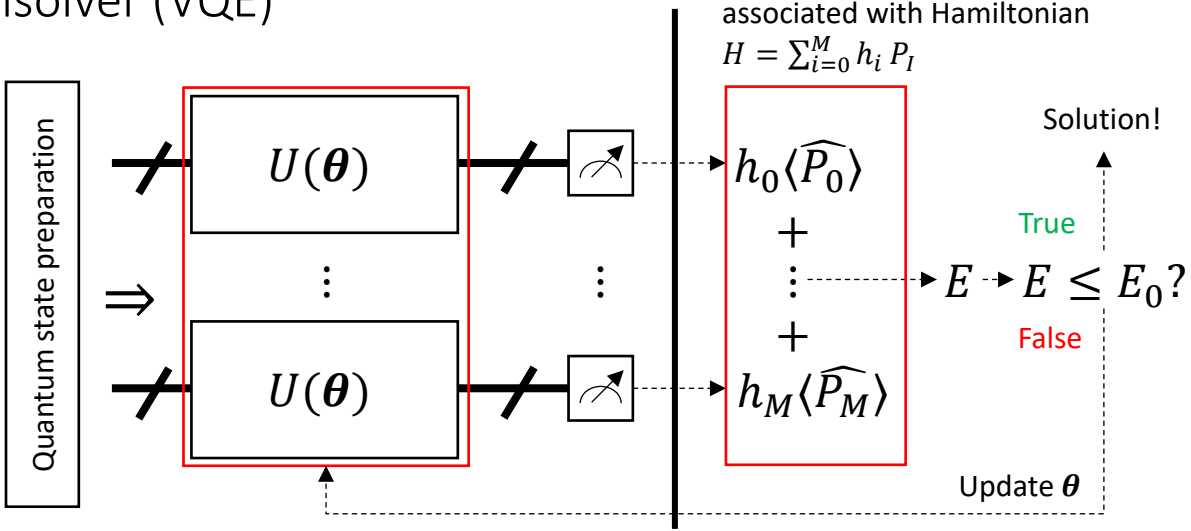
1.6.1. Quantum optimization

Variational Quantum Eigensolver (VQE)

Finding ground energy E_0 corresponding to ground state $|\psi_0\rangle$ of Hamiltonian H

$$\boldsymbol{H|\psi_0\rangle = E_0|\psi_0\rangle}$$





Simple Hamiltonian: $\widehat{H}_{Ising} = -\sum_i J_i Z_i Z_{i+1} - \sum_i h_i X_i$

Electronic structure → Basis set expansion (ex: sto3g)

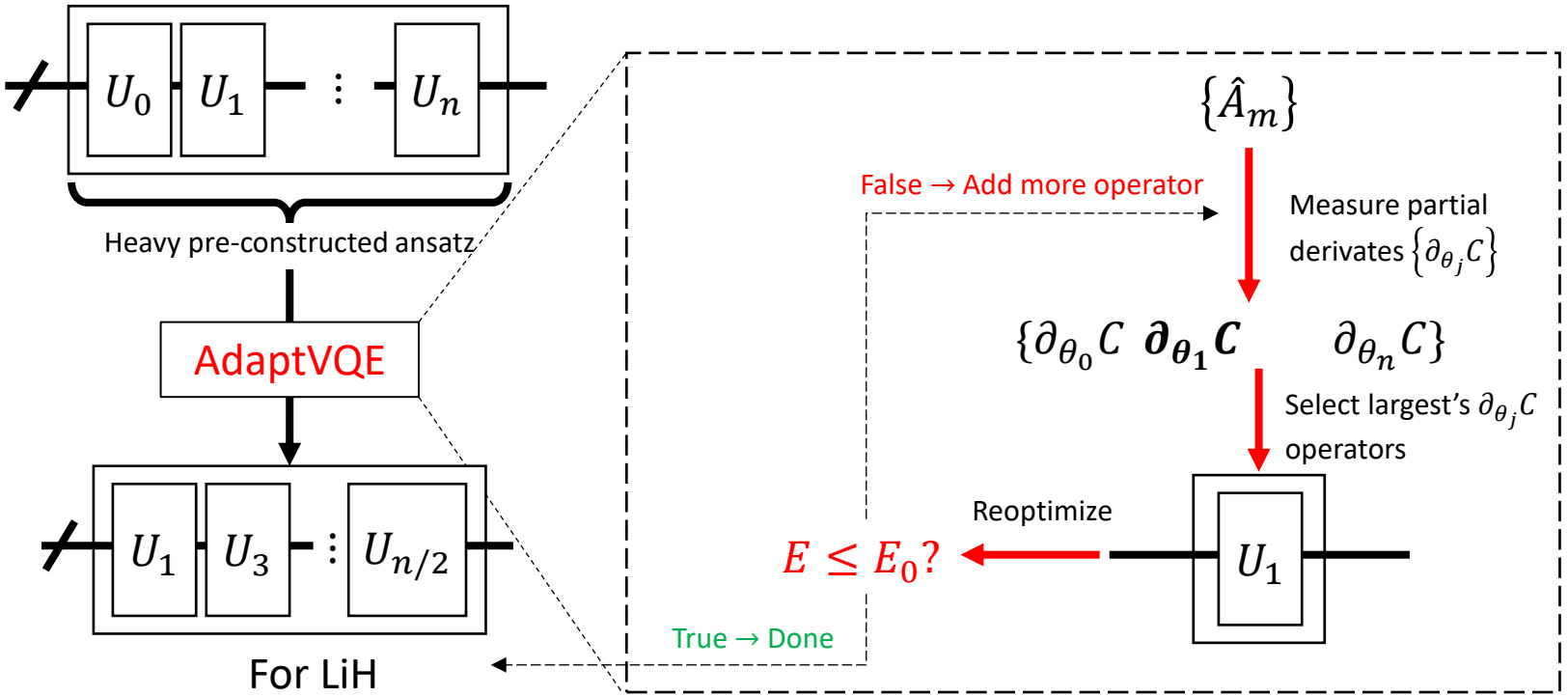
→ Mapping to qubits (ex: JW transform)

→ Pauli decomposition → $\widehat{H}$

[1.31] https://pennylane.ai/qml/demos/tutorial_vqe_intro

1.6.1. Quantum optimization

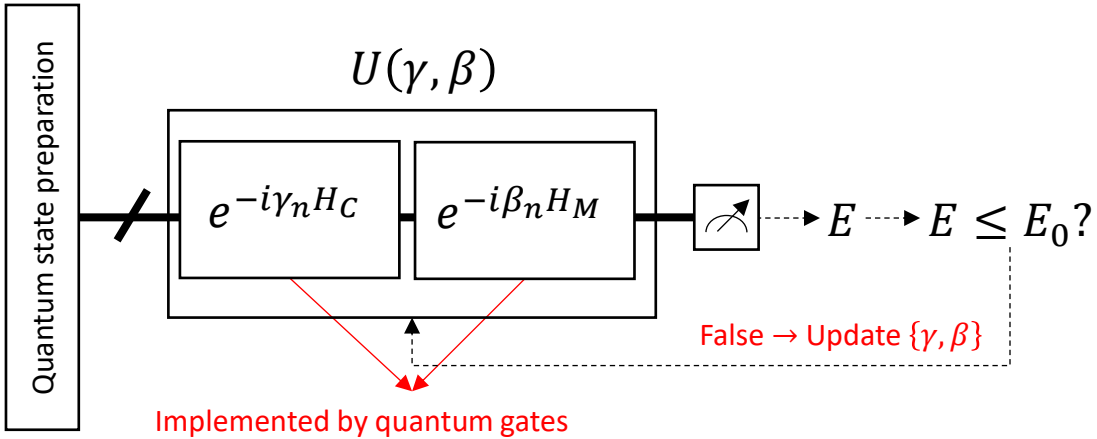
VQE → AdaptVQE



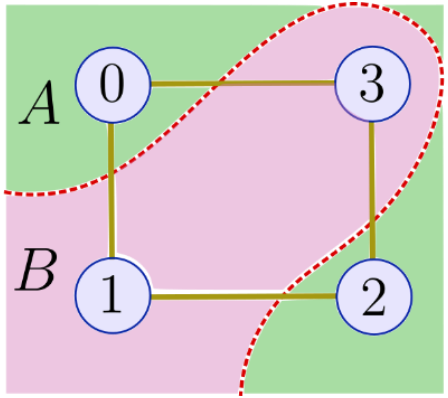
[1.32] Grimsley, H. R., Economou, S. E., Barnes, E., & Mayhall, N. J. (2019). An adaptive variational algorithm for exact molecular simulations on a quantum computer. *Nature communications*, *10*(1), 3007.

1.6.2. Quantum optimization

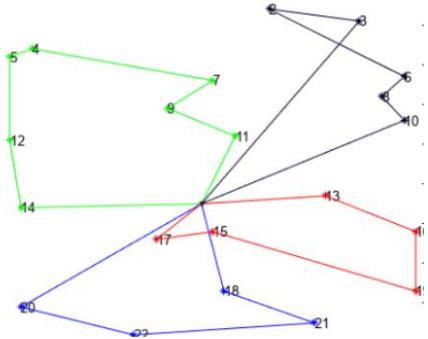
Quantum Approximate Optimization Algorithm (QAOA)



1. Problem Definition (QUBO)
2. Ising Hamiltonian: $\widehat{H}_{Ising} = -\sum_i J_i z_i z_{i+1} - h_i \sum z_i \quad (z_i \in \{\pm 1\})$
3. Find optimal $\{\gamma^*, \beta^*\}$



Min/Max cut

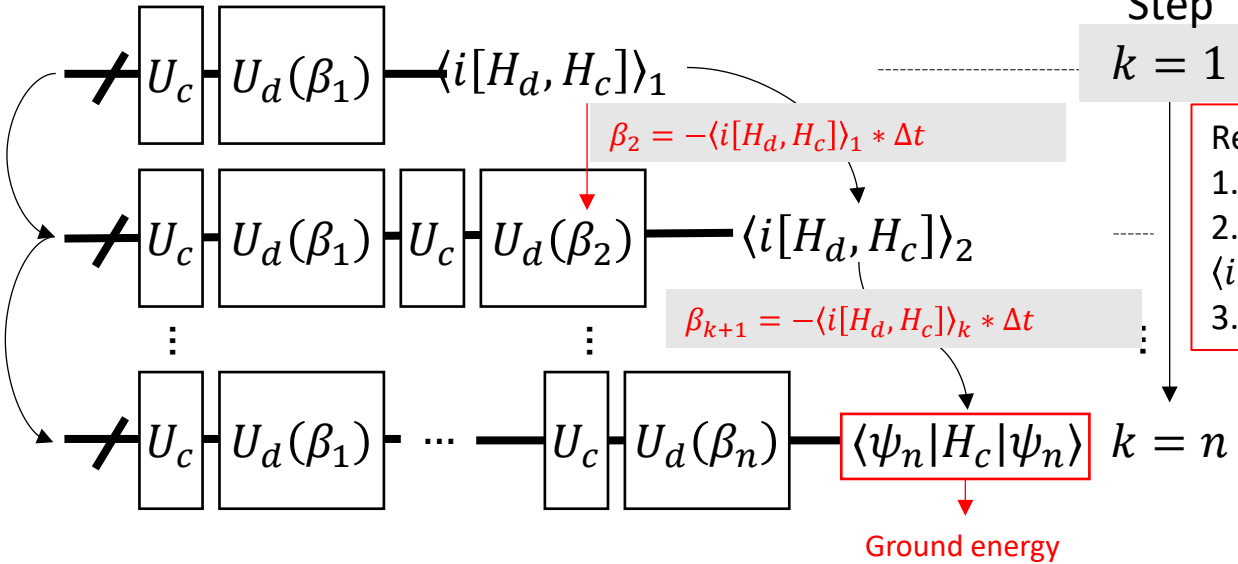


Vehicle routing

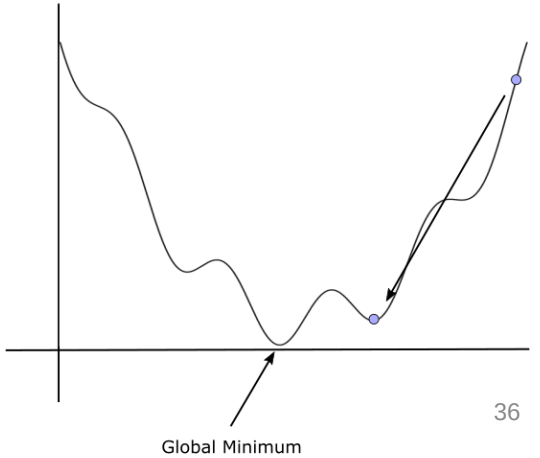
[1.33] https://pennylane.ai/qml/demos/tutorial_qaoa_intro

1.6.2. Quantum optimization

QAOA → FALQON $\langle i[H_d, H_c]\rangle_k \equiv \langle \psi_1 | i[H_d, H_c] | \psi_1 \rangle$



Repeat $k = 1$ to n:
1. Prepare $|\psi_k\rangle = U_d(\beta_k)U_c \dots U_d(\beta_1)U_c|\psi_0\rangle$
2. Measure the expectation value $\langle i[H_d, H_c]\rangle_k$
3. Set $\beta_{k+1} = -\langle i[H_d, H_c]\rangle_k * \Delta t$

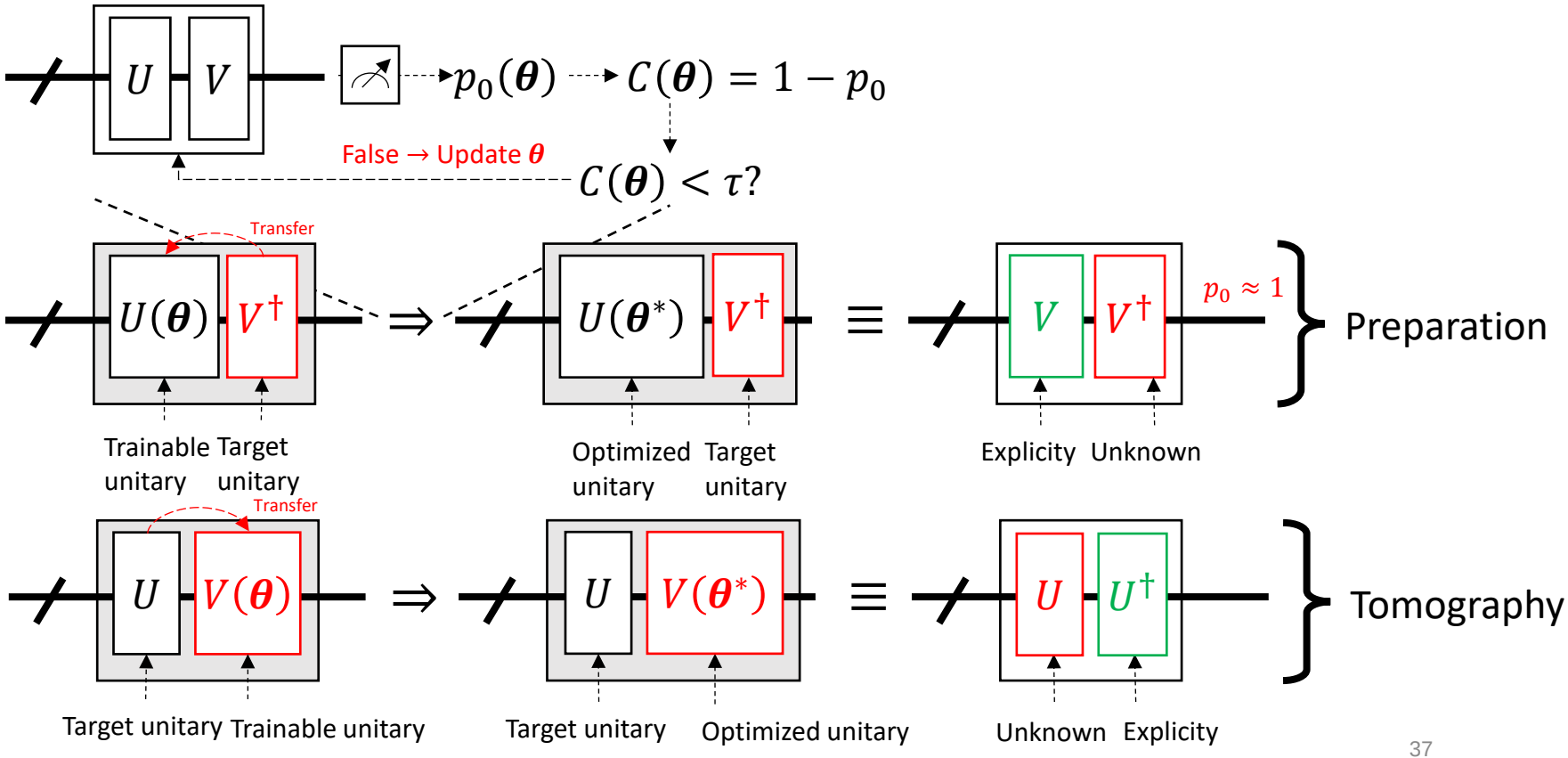


1. Problem Definition Objective Avoid local minimals
2. Hamiltonian $H = H_c + \beta(t) H_d$
3. Find ground energy corresponding to H_c

[1.34] https://pennylane.ai/qml/demos/tutorial_falqon

1.6.3. Quantum Optimization

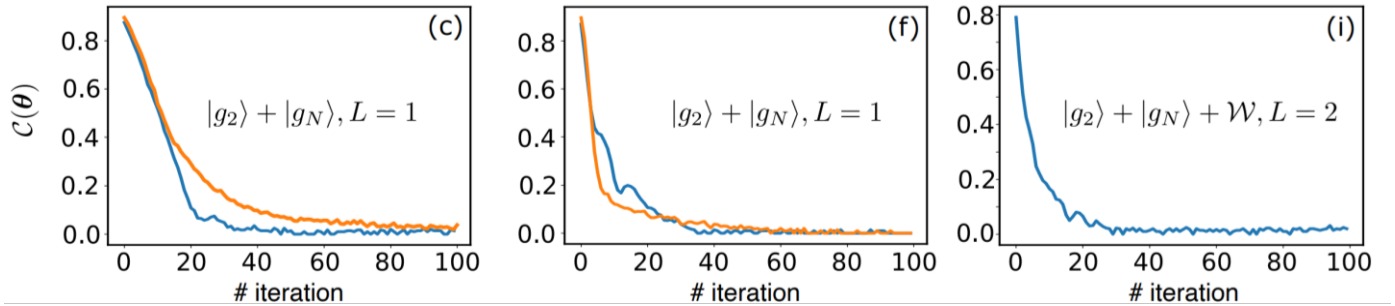
Quantum compiling



[1.35] Hai, V. T., Viet, N. T., & Ho, L. B. (2024). *SoftwareX*, *26*, 101726. https://doi.org/10.1016/j.softx.2024.101726

1.6.3. Quantum Optimization

Quantum compiling



Some entangled states are hard for preparation:

$$|\mathrm{GHZ_n}\rangle = \frac{1}{\sqrt{2}}\left(|0\rangle^{\otimes n} + |1\rangle^{\otimes n}\right)$$

$$|W_n\rangle = \frac{1}{\sqrt{n}}\left(|0\rangle + |1\rangle + \cdots + |n\rangle\right)$$

Almost unitaries acts as "black-box"

U only being extracted via measurement

If $U(\boldsymbol{\theta}^*) \equiv GHZ^\dagger$ then preparing $|GHZ\rangle$ through $U(\boldsymbol{\theta}^*)$ is better

If $V(\boldsymbol{\theta}^*) \equiv U$ we know structure of U via $V(\boldsymbol{\theta}^*)$

Preparation Tomography

[1.36] Hai, V. T., & Urbaneja, J. (2025, July). Entangled State Preparation via Cluster States on Quantum Computers with $\langle qo|op\rangle$ Software. In *2025 IEEE International Conference on Quantum Software (QSW)* (pp. 116-122). IEEE.

[1.37] Hai, V. T., & Ho, L. B. (2023). Universal compilation for quantum state tomography. *Scientific Reports*, *13*(1), 3750.

1.7. Additional informations

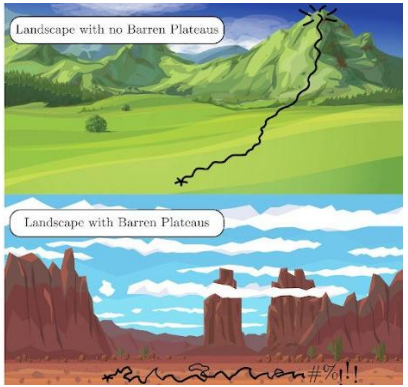
Barren platueas

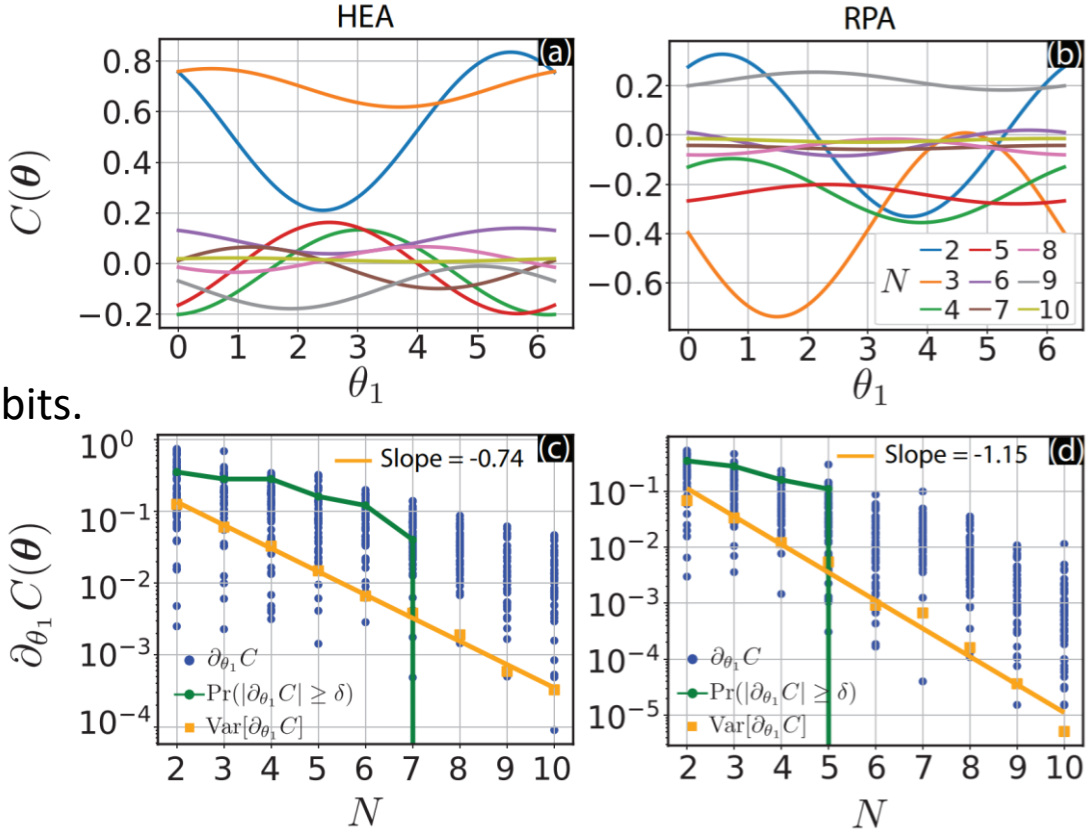
Analyze a PQC, these term:

$$\mathbb{E}\left[\frac{\partial\langle H\rangle}{\partial\theta_i}\right] = \langle\partial C\rangle,$$

$$\mathbf{Var}\left[\frac{\partial\langle H\rangle}{\partial\theta_i}\right] = \langle\partial C^2\rangle - \langle\partial C\rangle^2$$

vanish exponentially based on #qubits.



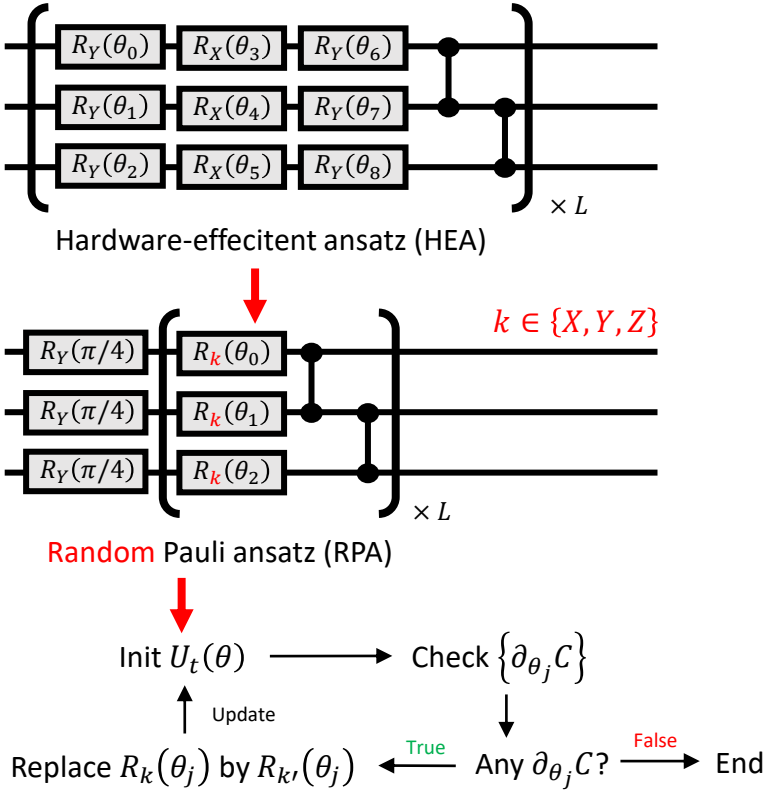


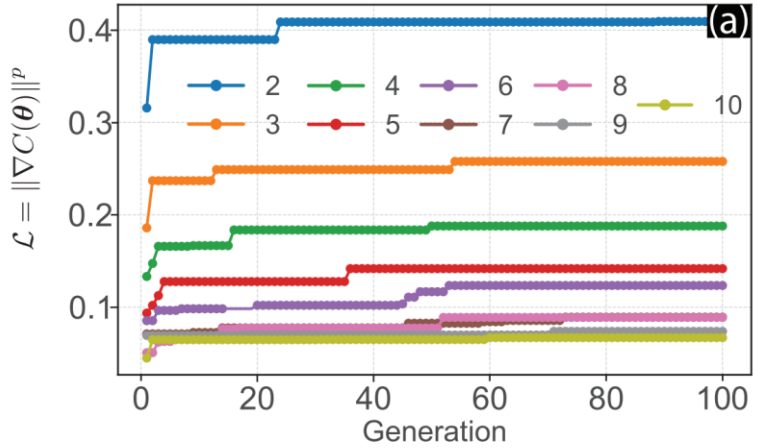
[1.38] McClean, Jarrod R., et al. "Barren plateaus in quantum neural network training landscapes." Nature communications 9.1 (2018): 1-6.

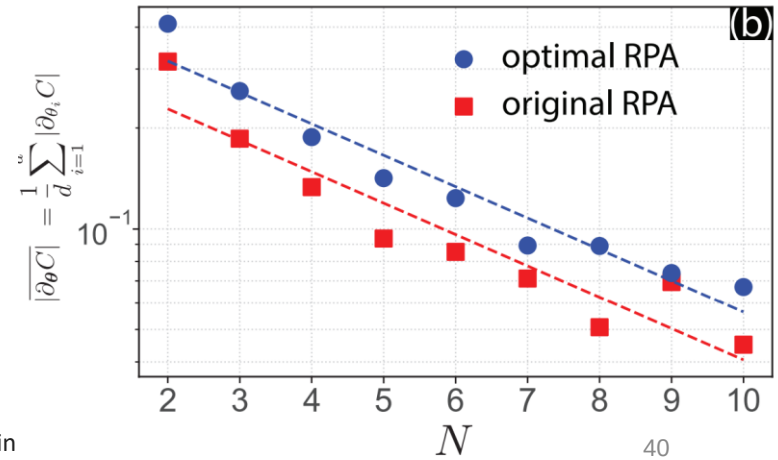
1.7. Additional informations

Barren platueas (BP) → A method for mitigate BP

Different ansatz affect strongly to the performance







[1.39] Ho, L. B., Urbaneja, J., & Ashhab, S. (2025). Statistical analysis of barren plateaus in variational quantum algorithms. *arXiv preprint arXiv:2508.08915*.

1.7. Additional informations

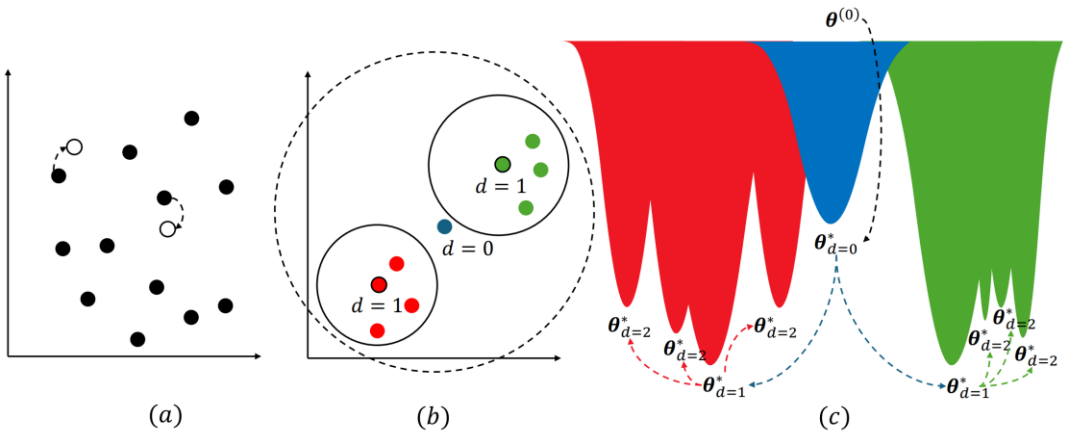
Can we apply classical techniques into quantum optimization?

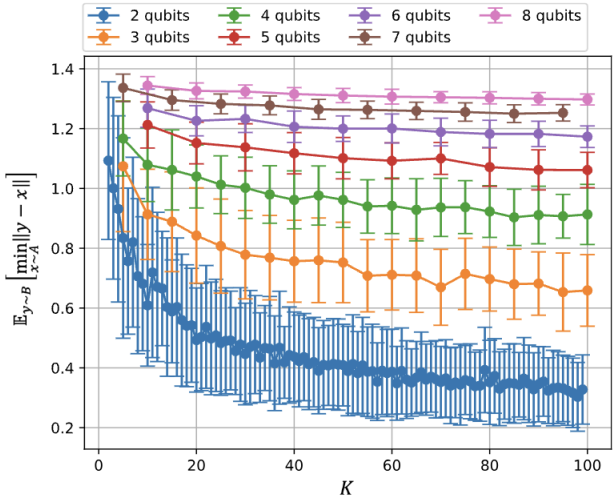
Consider a set of K optimization targets denoted as $T_1, T_2, \dots, T_K$, all defined over the same search space S. Each target T_k is associated with its own cost function:

$$C\left(\theta_{(k)}\right): R^m \rightarrow [0,1]$$

The goal of multi-target optimization is to find a set of solutions:

$$\{\theta^*_{(1)}, \theta^*_{(2)}, \dots \theta^*_{(K)}\}$$





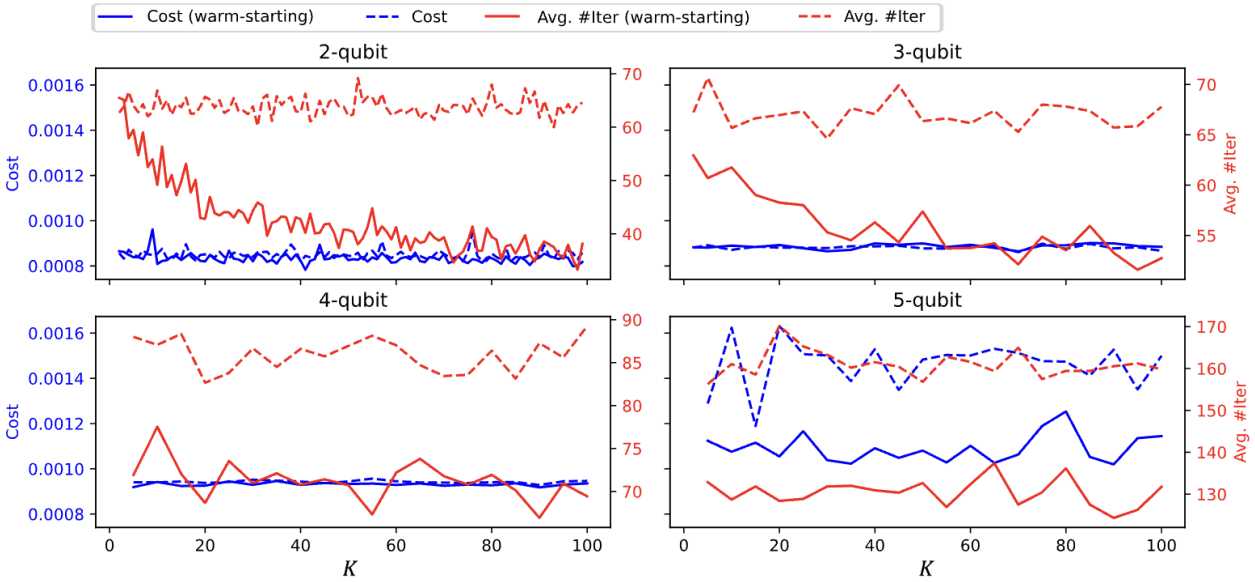
[1.41] Hai, V. T., Doanh, B. C., Duong, L. V. T., Luan, P. H., & Nakashima, Y. (2025). Transfer-Based Strategies for Multi-Target Quantum Optimization. *CANDAR 2025.*

1.7. Additional informations

Can we apply classical techniques into quantum optimization?

Warm-starting method: simply set $\theta^*_{x_{(k)}}$ as initial parameter $\theta^{(0)}_{y_{(k')}} = \theta^*_{x_{(k)}}$.

If $y \approx x + \epsilon$, x and y are targets, considering the first-order Taylor expansion of cost function C with respect to $\boldsymbol{\theta}$ at $\boldsymbol{\theta}^*_{\boldsymbol{x}}$ and fixed y, we have:



$$\widetilde{\boldsymbol{\theta}}^*_y \approx \boldsymbol{\theta}^*_x - \frac{\nabla_\theta C_y(\theta^*_x)}{\left\|\nabla_\theta C_y(\theta^*_x)\right\|^2} C_y(\theta^*_x)$$

Parameter estimator

2. Quantum machine learning (QML)

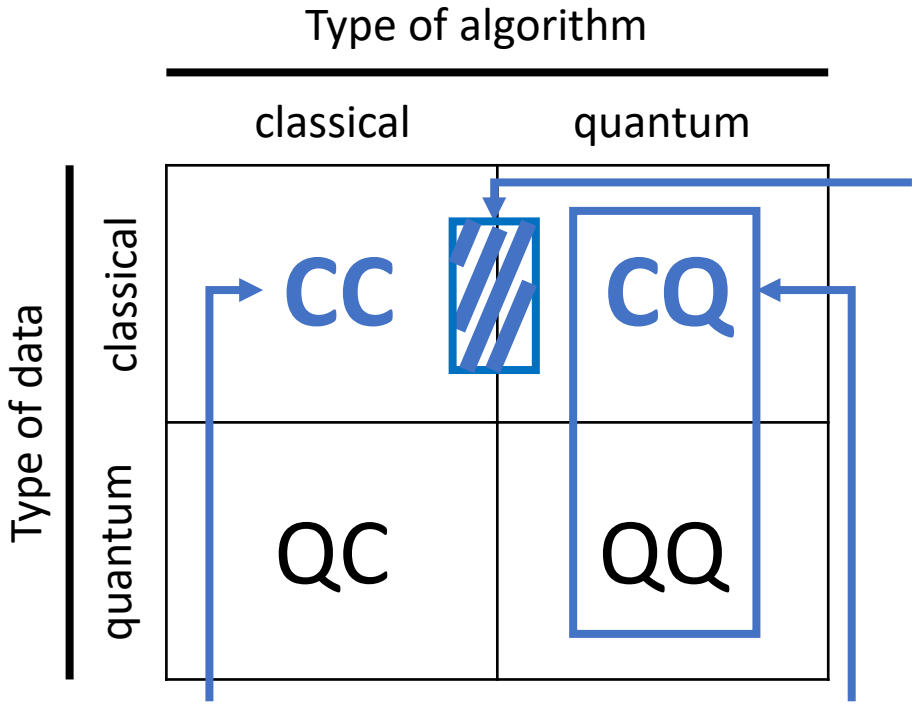
2.1. Overview

2.2. Quantum machine learning algorithms

2.3. Quantum deep learning

2.4. Additional information

2.1. Overview



Hybrid quantum – classical learning: algorithm running on both quantum and classical computers

QML model should resolve below classical machine learning problems:

(1) Require huge dataset

(2) High computation resources

(3) Lack of generality

Classical model: KNN, SVM, Deep learning, …

Quantum Machine learning (QML): algorithm running on only quantum computer or both quantum / classical computers

2.1. Overview

What QML models can do?
Based on quantum advantage [2.2]

(1) Require huge dataset

⇒ Require less items

(2) High computation resources

⇒ Lower time complexity

(3) Lack of generality

⇒ Example: Require less #parameter than classical ML model or achieve higher accuracy on the same dataset.

Be careful with hype QML papers!

[2.1] Simões, R. D. M., Huber, P., Meier, N., Smailov, N., Füchslin, R. M., & Stockinger, K. (2023). Experimental evaluation of quantum machine learning algorithms. *IEEE Access*, *11*, 6197-6208.
[2.2] Schuld, M., & Killoran, N. (2022). Is quantum advantage the right goal for quantum machine learning?. *Prx Quantum*, *3*(3), 030101.

2.1. Overview

Type of QML model

a) Speed up version of classical ML algorithms:

- QKNN
- QKmean
- QSVM

b) Quantum analogue version of a deep learning model

- QuNN, QCNN
- QGNN
- QGAN

c) Quantum learning

→ I don’t know this type

2.1. Overview

Pioneers in QML

Maria Schuld

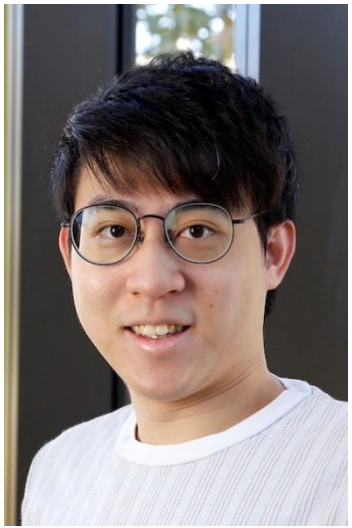
Xanadu.ai (Pennylane)

Ryan Babbush

Google Quantum

John Preskill

Caltech

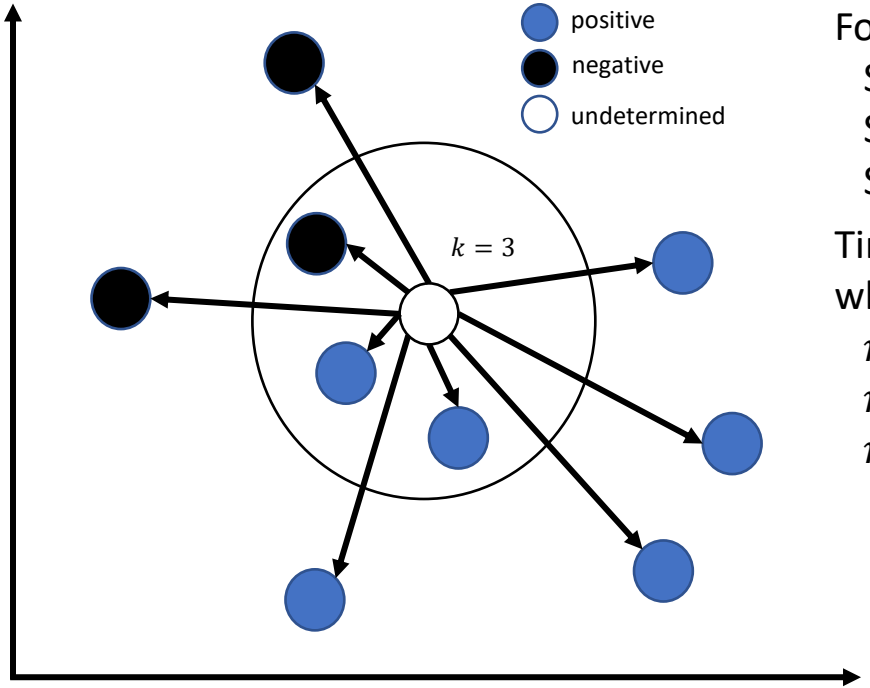


Hsin-Yuan Huang

Google Quantum AI

2.2. QML algorithms

QkNN



For each test data point a:

Step 1: Calculate $\mathrm{Ds_a} = dist(a, x_i)\ \forall\ i = 1..n$

Step 2: Sort and get k smallest dist in $\mathrm{Ds_a}$.

Step 3: Get k corresponding label and do major vote

Time complexity: $\mathcal{O}(mnn_{\mathrm{test}})$

where:

m: dimensional of x_i

n: # train data points

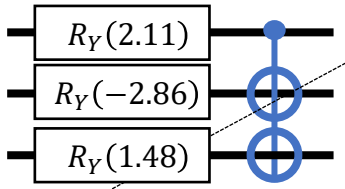
n_{test}: # test data points

2.2. QML algorithms

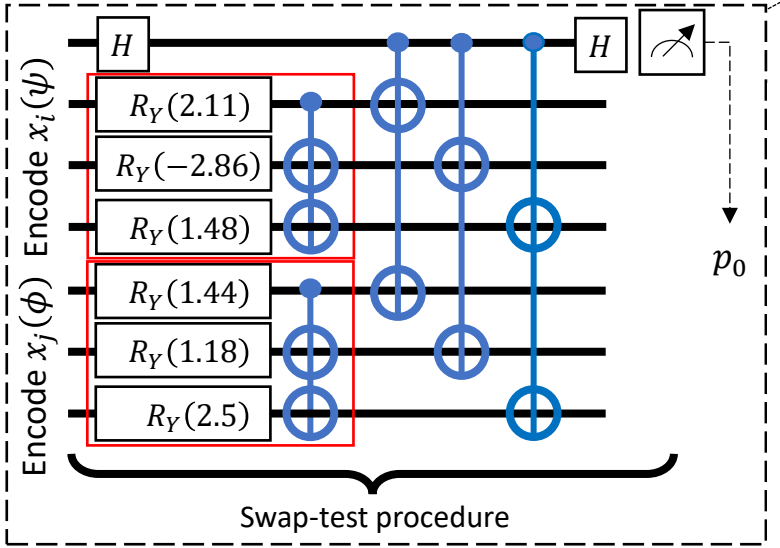
QkNN

$$x = \begin{bmatrix}1\\0\\2\\3\end{bmatrix} \xrightarrow{\text{Normalized}} \hat{x} = \begin{bmatrix}1/\sqrt{14}\\0\\2/\sqrt{14}\\3/\sqrt{14}\end{bmatrix}$$

Encoded



Fredkin-gate ≡ SWAP



For each test data point a:

Step 1: Calculate $Ds_a = dist(a, x_i)\ \forall\ i = 1..n$

Step 2: Sort and get k smallest dist in $\mathrm{Ds_a}$.

Step 3: Get k corresponding label and major vote

$$|0\rangle|\psi\rangle|\phi\rangle \xrightarrow{H_0} \frac{1}{\sqrt{2}}(|0\rangle|\psi\rangle|\phi\rangle + |1\rangle|\psi\rangle|\phi\rangle)$$

$$\xrightarrow{\{SWAP_{i,j}\}} \frac{1}{\sqrt{2}}(|0\rangle|\psi\rangle|\phi\rangle + |1\rangle|\phi\rangle|\psi\rangle)$$

$$\xrightarrow{H_0} \frac{1}{2}(|0\rangle|\psi\rangle|\phi\rangle + |1\rangle|\phi\rangle|\psi\rangle) + \frac{1}{2}(|0\rangle|\phi\rangle|\psi\rangle - |1\rangle|\psi\rangle|\phi\rangle)$$

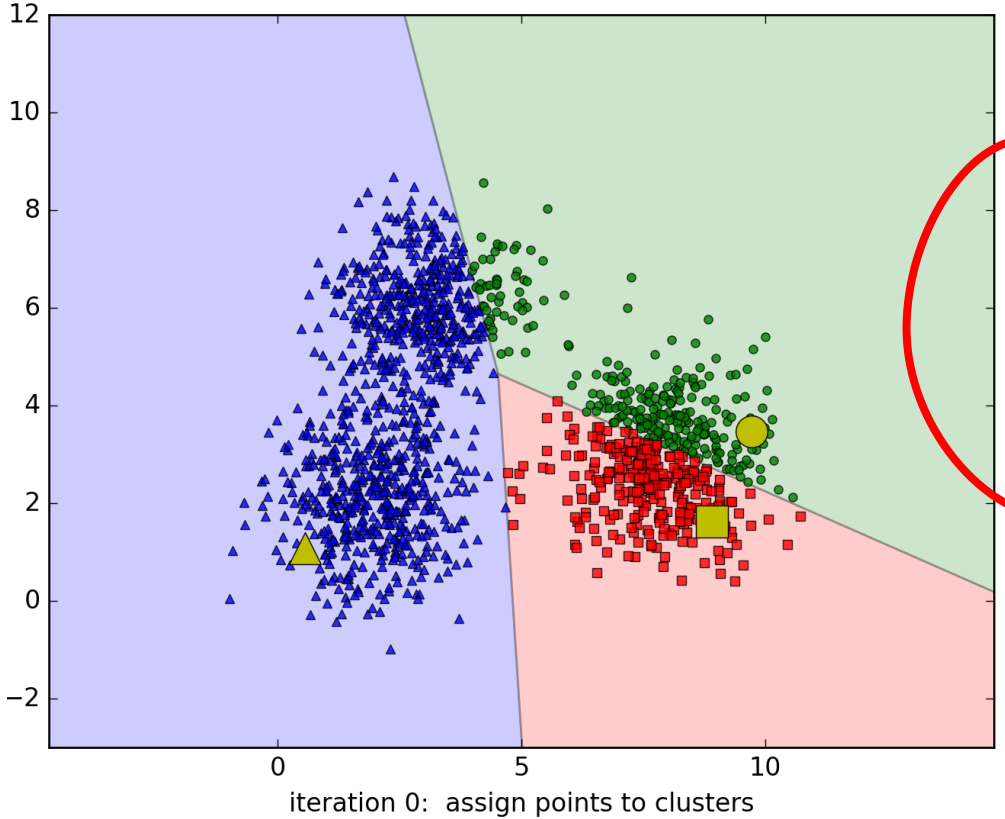
So $p_0 = \frac{1}{2}(\langle\phi|\langle\psi| + \langle\psi|\langle\phi|)\frac{1}{2}(|\phi\rangle|\psi\rangle + |\psi\rangle|\phi\rangle)$

$$\Rightarrow \boxed{|\langle\psi|\phi\rangle| = 2p_0 - 1}$$

[2.3] Vu Tuan Hai, Phan Hoang Chuong and Pham The Bao (2022). New approach of KNN Algorithm in quantum computing based on new design of quantum circuits. *Informatica*, *46*(5).

2.2. QML algorithms

k-means



Step 1. Select k random centers.

Step 2. Assign each data point to the closest center.

If the assigned labels do not change, then stop.

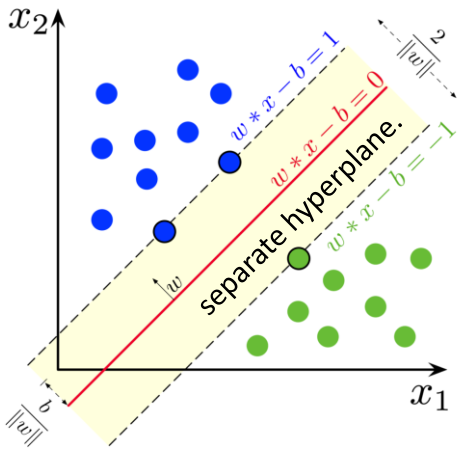
Step 3. New center = average of all assigned points

Step 4. Return to step 2.

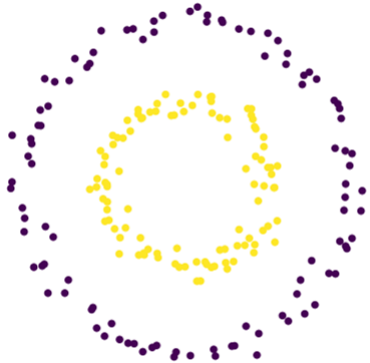
Swap-test procedure can be applied again!

2.2. QML algorithms

Quantum SVM



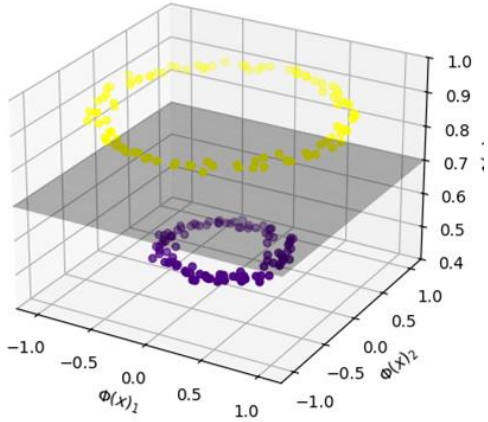
$\Rightarrow$



Inseparable data

$x = [x_1 \quad x_2]^\intercal$

Kernel method $\Rightarrow$



Separable data

$\phi(x) = [x_1 \quad x_2 \quad x_1 x_2]^\intercal$

We have: parameters

$$f^*(x) = \mathbf{sgn}(w \cdot x \pm b),$$

with target (w^*, b^*):

$$\min_{w,b,c} \frac{1}{2}\|w\|_2^2 + C\sum_{i=1}^{n} \epsilon_i$$

$$\max_{\beta} L(\beta) = \sum_{i=1}^{n} \beta_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \beta_i \beta_j y_i y_j \, \phi(x_i)^\intercal \phi(x_j)$$

Inner product

$\mathcal{O}(M * ?)$ complex

$$K(x_i, x_j) \equiv \phi(x_i)^\intercal \phi(x_j)$$

Kernel function is faster!

2.2. QML algorithms

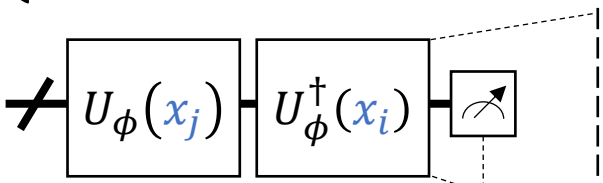
Quantum SVM

Classical kernel:

$$K_{\text{RBF}}(x_i, x_j) = \exp\left(-\gamma \|x_i - x_j\|_2^2, \gamma > 0\right)$$

$$K_{\text{Linear}}(x_i, x_j) = x_i^{\intercal}\, x_j$$

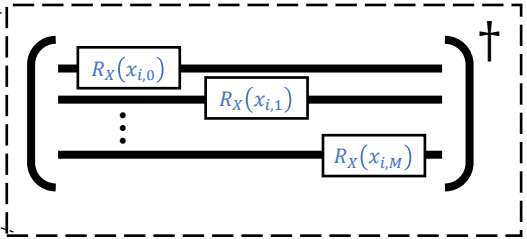
Quantum kernel:



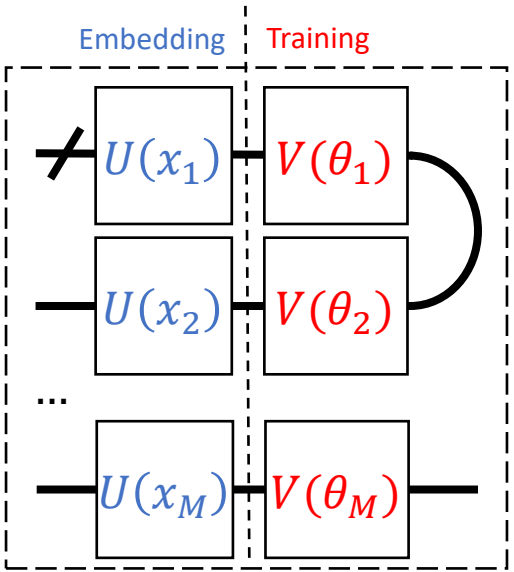
Unitary operator as kernel

$$p_0 = \left|\langle U_\phi(x_j) | U_\phi(x_i)\rangle\right|^2$$

"Nature" inner product

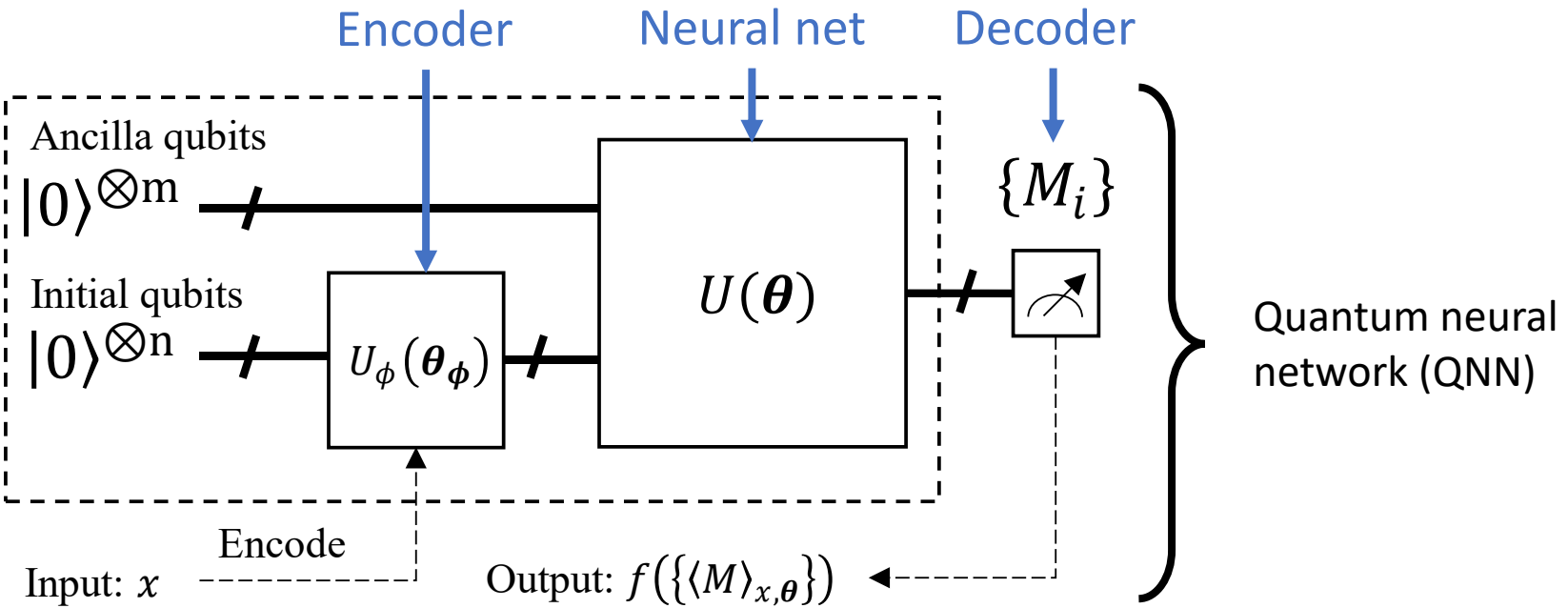


Feature encoding (embedding)
(Fixed kernel)



Data re-uploading
(Trainable kernel)

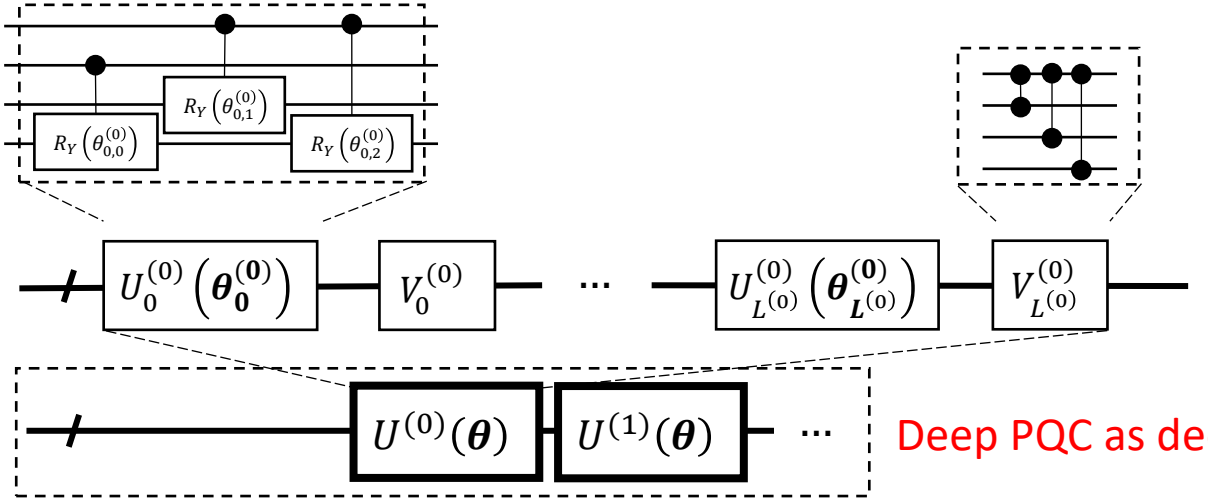
2.3. Quantum deep learning



[2.4] Benedetti, M et al, Parameterized quantum circuits as machine learning models. Quantum Science and Technology, 4(4), 043001 (2019)

2.3. Quantum deep learning

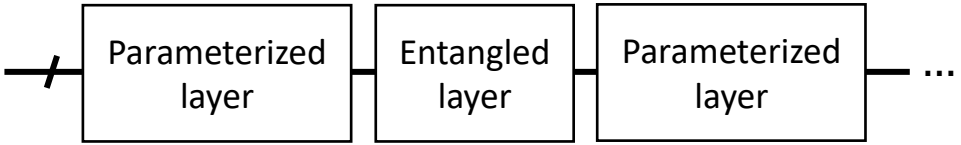
QNN



Deep PQC as deep QNN

Note that $U(\theta, \lambda, \phi) \equiv \left[R_X(\theta)\ R_Y(\theta)\ R_Z(\theta)\right] \times K$

Then



Successive parameterized gates are meaningless

Entangled gates (CX. MCX, Control-U, ...) should be appended in the middle

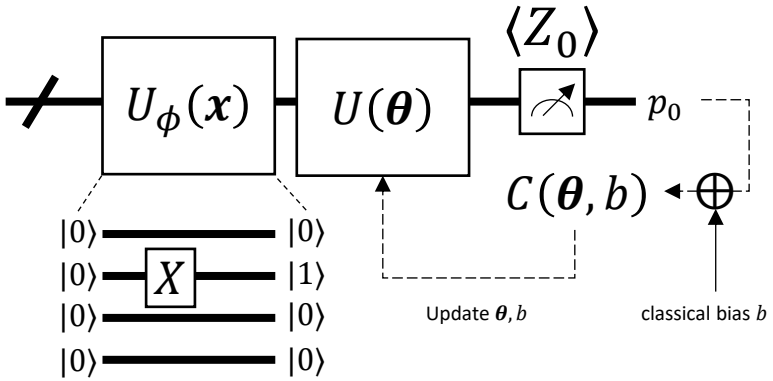
2.3. Quantum deep learning

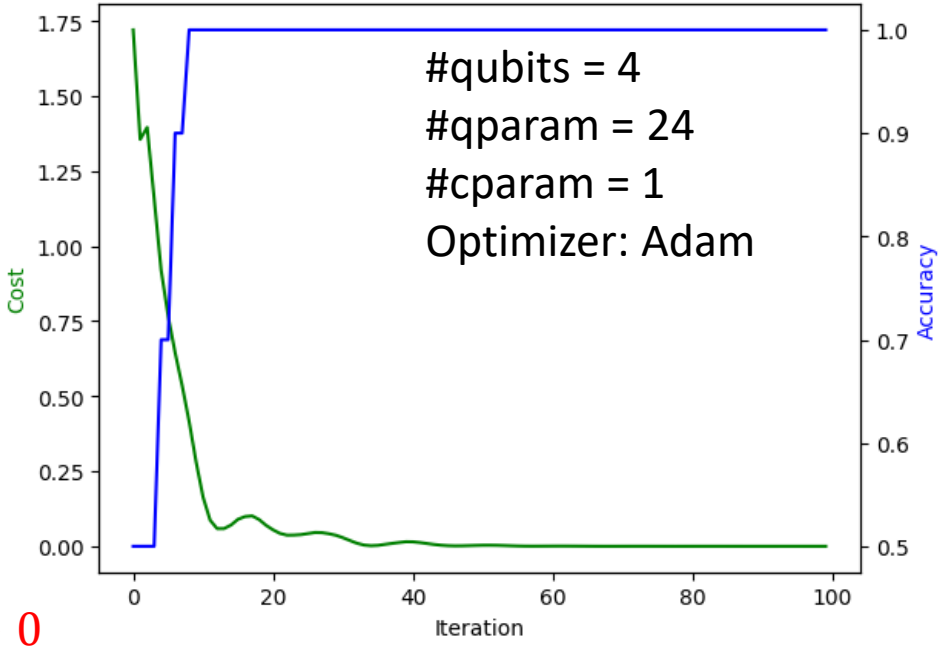
QNN

Attribute	**Classical deep learning**	**Quantum deep learning**
Main concept	Neural network	Parameterized quantum circuit (PQC)
Parameter	Weight $\{w_i\}$ in nodes	Angle value $\{\theta_i\}$ in rotation gates $\{R_X, R_Y, R_Z\}$
Gradient	Backpropagation	General parameter-shift rule and gradient tensor metric
Optimizer	Classical optimizer	Classical optimizer + gradient-free optimizer
Loss	Various type	Associated with expectation values

2.3. Quantum deep learning

Variational classifier for simple classification problem





Problem:

Simulate $f: x \in \{0,1\}^{\otimes n} \rightarrow y = \oplus_i x_i$

Example: $f(|0100\rangle) = 1$

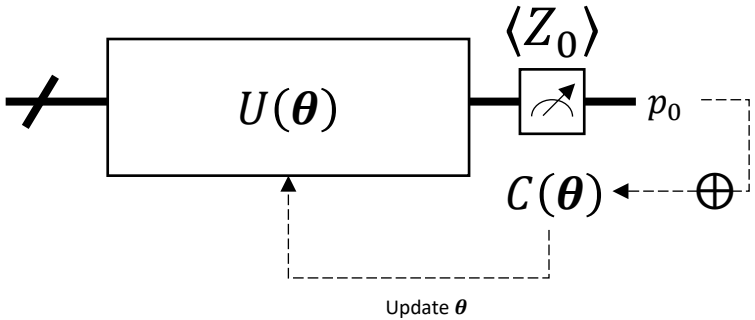
Cost function $C(\boldsymbol{\theta}, b) = \frac{1}{N}\sum(\hat{y}_i - y_i)^2 \rightarrow 0$

Accuracy metric $A = \frac{1}{N}\sum \mathbb{I}(|y_i - \hat{y}_i| < \tau) \rightarrow 1$

[2.5] https://pennylane.ai/qml/demos/tutorial_variational_classifier

2.3. Quantum deep learning

Variational classifier for ordinary differential equation (ODE)



Solve ODE: $y'(x) = -0.5\sin(x) + 0.5$.

Then find $y(x)$ $\left(y(x) = \frac{1}{2}(\cos(x) + x)\right)$.

Assume that $U(\boldsymbol{\theta}, x) \equiv y(x)$, then:

$$y'(x) = \frac{dU}{dx} = \frac{\left(U(\boldsymbol{\theta}, x+\epsilon) - U(\boldsymbol{\theta}, x-\epsilon)\right)}{2\epsilon}$$

We **simulate** $y(x)$ via $U(\boldsymbol{\theta}, x)$

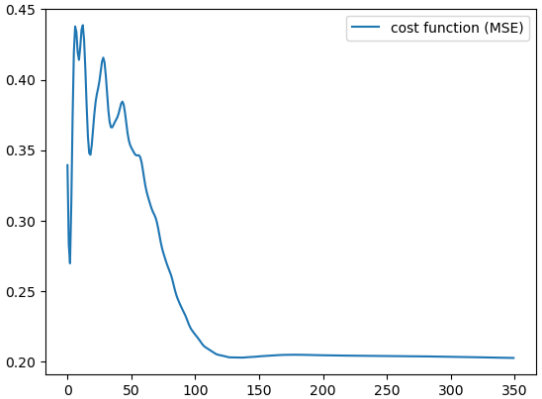
Cost function:

$$C(\boldsymbol{\theta}) = [U(\boldsymbol{\theta}, x_0) - y_0]^2 + \sum_j \left(\frac{dU(\boldsymbol{\theta}, x_j)}{dx} - y'(x_j)\right)^2$$

Sampling from y'

$$\{\partial C(\boldsymbol{\theta})/\partial\theta_j\}$$

$$\min_{\boldsymbol{\theta}} C(\boldsymbol{\theta})$$



[*] Results from Nguyen Quoc Chuong with Python package Qsun

2.3. Quantum deep learning

Variational classifier for quantum battery optimization

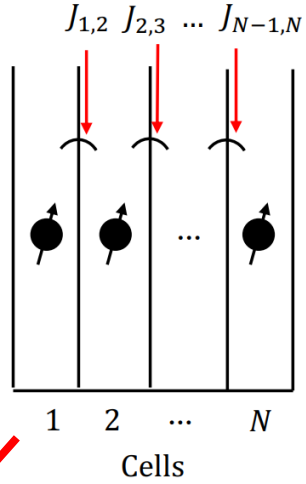
A quantum battery is given in the ground state of a Hamiltonian $H_0(h) = -h\sum_{j=1}^{N} Z_j$. The **charging Hamiltonian** is given by:

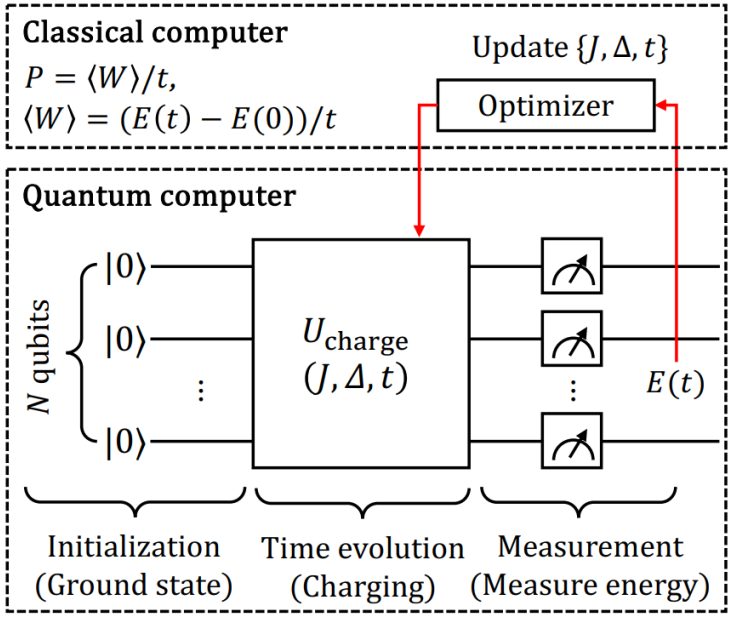
$$H_{\text{charge}}(J,\Delta,\gamma) = \sum_{j=1}^{N-1} J_j h(1+\gamma)X_jX_{j+1} + (1-\gamma)Y_jY_{j+1} \quad (5.1)$$

$$-\sum_{j=1}^{N-1} \Delta_j Z_jZ_{j+1} + H_0(h)$$

Optimizing $\{J, \Delta\}$ mean we have a better battery

How to optimize $\{J, \Delta\}$?





(a) A model of N-cell quantum battery.
(b) A quantum machine learning algorithm for optimizing battery's power.

2.3. Quantum deep learning

Variational classifier for quantum battery optimization

Classical Quantum

t: charging time

$|\psi(0)\rangle$: initial ground state

After charging, we have power ($\boldsymbol{P}$) (performance of battery) :

$$\langle \boldsymbol{W} \rangle = \langle \boldsymbol{\psi(t)} | \boldsymbol{H_0} | \boldsymbol{\psi(t)} \rangle - \langle \boldsymbol{\psi(0)} | \boldsymbol{H_0} | \boldsymbol{\psi(0)} \rangle$$

$$\boldsymbol{\psi(t)} = \boldsymbol{e^{-iHt}} |\boldsymbol{\psi(0)}\rangle \quad (5.2)$$

Calculate directly

$\boldsymbol{P = \langle W \rangle / t}$

Can be found after survey $t \in [0, T]$ with $t^{k+1} \leftarrow t^k + \Delta_t$

$\boldsymbol{P_{max} > P\ \forall t?}$

We want to find "max" of $P_{\max}$

$\boldsymbol{P^*_{max} > P_{max}\ \forall J, \Delta}$ → $\boldsymbol{\mathcal{L}(J, \Delta, \gamma) = -|P_{max}|}$

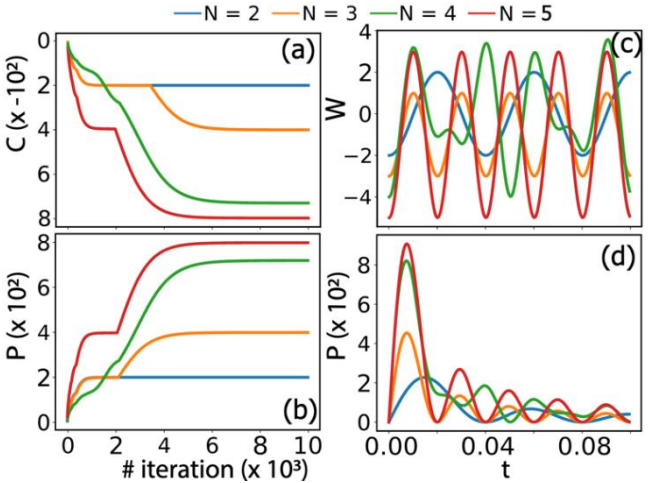
Calculate gradient

$\boldsymbol{\nabla \mathcal{L}}$

Update new $\{J, \Delta\}$ by classical optimizer

$\boldsymbol{(J|\Delta)^{t+1} = (J|\Delta)^t - \alpha \nabla \mathcal{L}}$

$$H_{\text{charge}} = \sum_{j=1}^{n-1} J_j \left(X_j X_{j+1} + Y_j Y_{j+1} \right)$$



(a) Plot of cost value

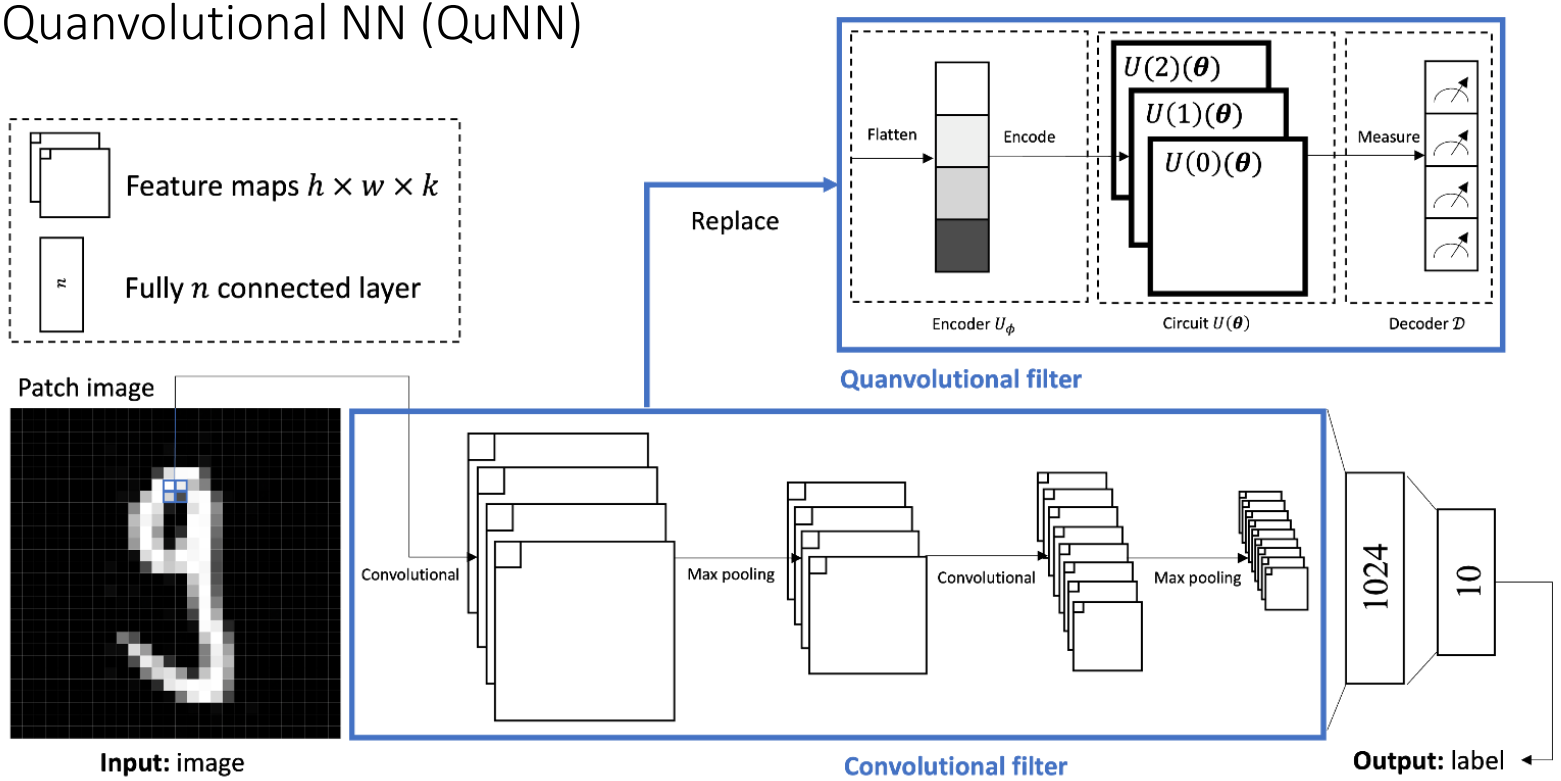
(b) Plot of power versus the iteration.

(c) Energy as a function of time.

(d) Power as a function of time.

2.3.1. Quantum deep learning

Quanvolutional NN (QuNN)

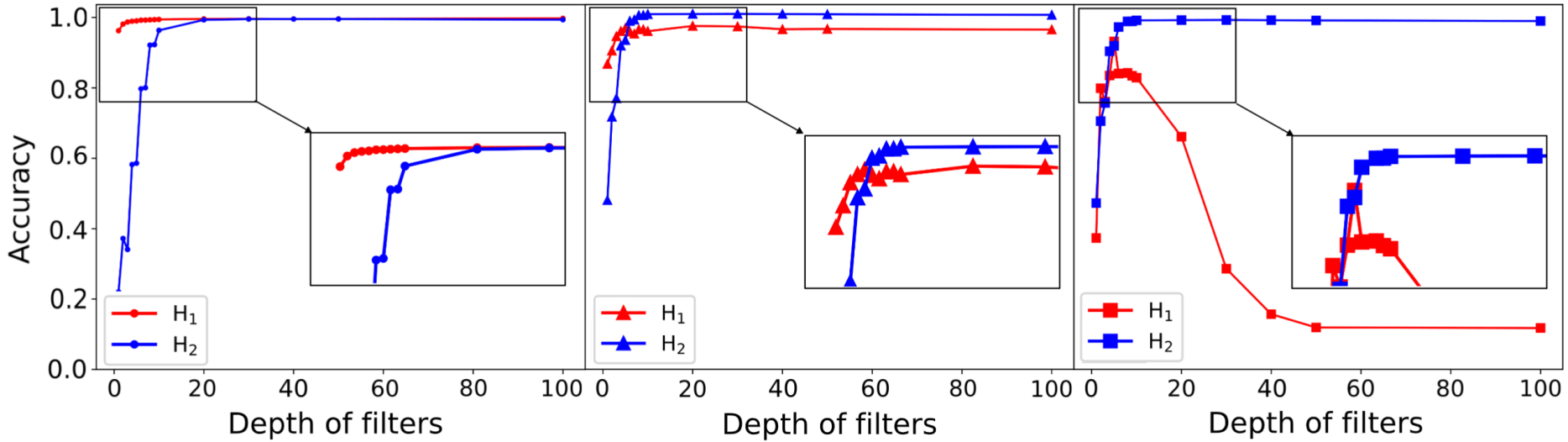


[2.6] Y. Lecun et al, Gradient-based learning applied to document recognition, Proceedings of the IEEE 86(11) (1998) 2278–2324

[2.7] T. Hur et al, Quantum convolutional neural network for classical data classification, Quantum Machine Intelligence 4 (Feb 2022) p. 3.

2.3.1. Quantum deep learning

Quanvolutional NN (QuNN)

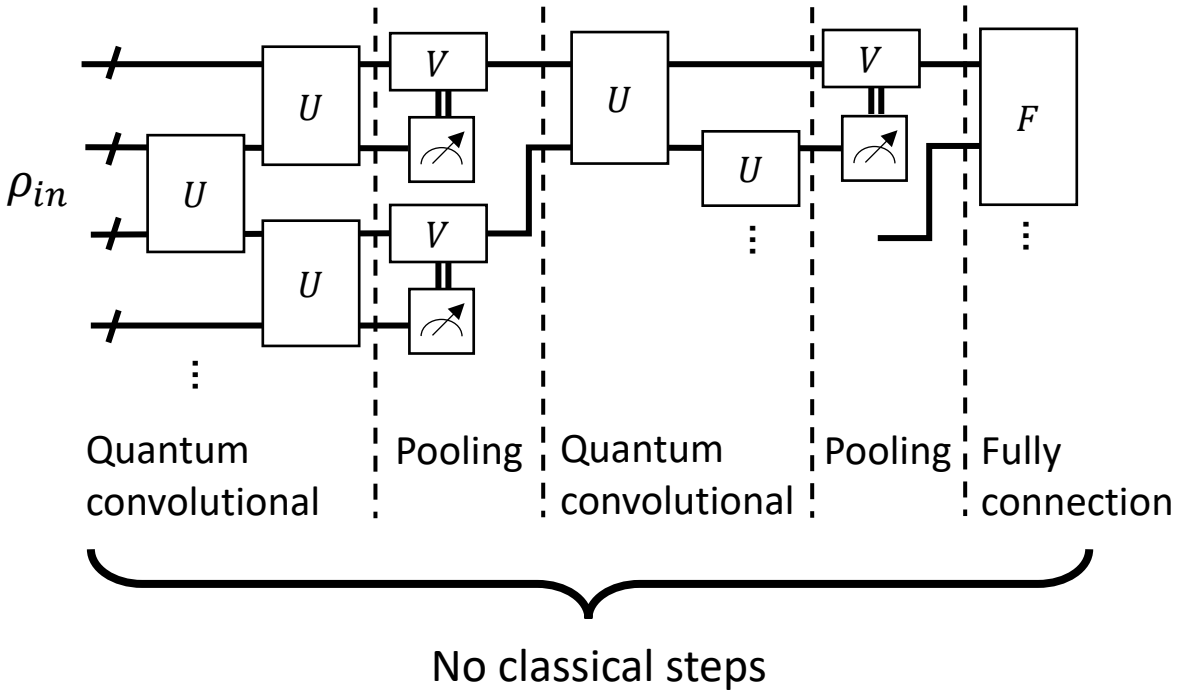


Accuracies of the CNN (H_1) and QNN (H_2) using MNIST (left), F-MNIST (center), and CIFAR-10 (right) datasets with depth of filters.

[2.8] Vu, T. H., Le, L. H., & Pham, T. B. (2024). Exploring the features of quanvolutional neural networks for improved image classification. *Quantum Machine Intelligence*, *6*(1), 29.

2.3.2. Quantum deep learning

Quantum Convolution NN (QCNN)



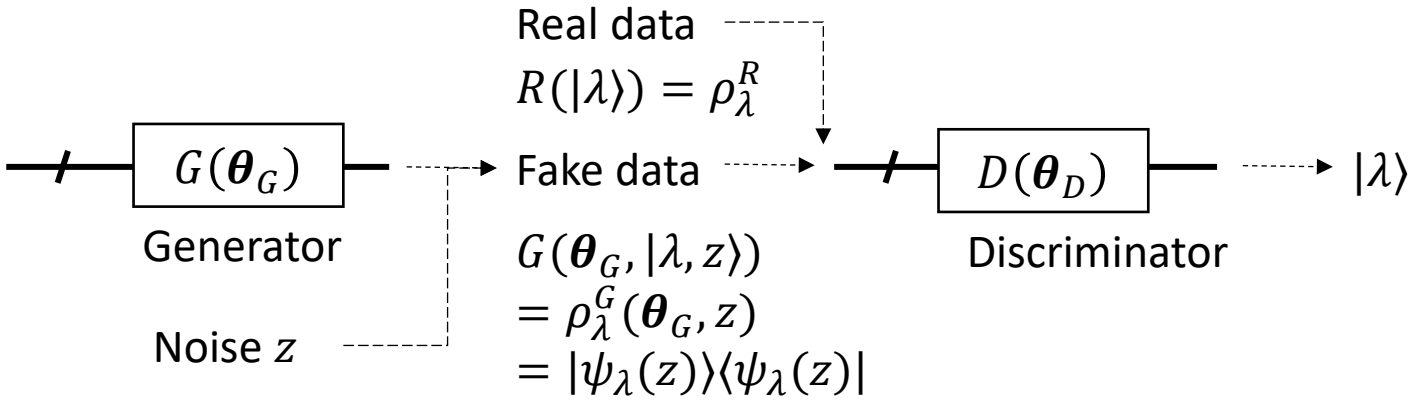
[2.9] Cong, I., Choi, S. & Lukin, M.D. Quantum convolutional neural networks. Nat. Phys. 15, 1273–1278 (2019). https://doi.org/10.1038/s41567-019-0648-8

2.3.3. Quantum deep learning

Quantum Generative adversarial net (QGAN)

Assume real data comes from the fixed distribution $p_R(x)$ generated by process R. Adversarial task:

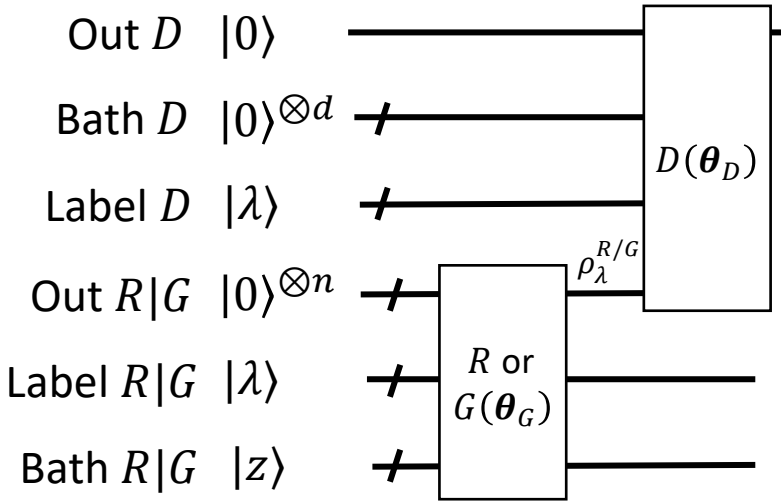
$$\min_{\boldsymbol{\theta}_G} \max_{\boldsymbol{\theta}_D} \sum_{\lambda=1} \Pr\Big(\big(D\big(\boldsymbol{\theta}_D, |\lambda\rangle, R(|\lambda\rangle)\big) = |\text{real}\rangle\big) \cap \big(D\big(\boldsymbol{\theta}_D, |\lambda\rangle, G(\boldsymbol{\theta}_G, |\lambda, z\rangle)\big) = |\text{fake}\rangle\big)\Big)$$



[2.10] https://pennylane.ai/qml/demos/tutorial_QGAN.html

2.3.3. Quantum deep learning

Quantum Generative adversarial net (QGAN)



$\langle Z\rangle \equiv |real\rangle\langle real| - |fake\rangle\langle fake|$

Initial state:

$$\rho_\lambda^0(z) = (|0\rangle\langle 0|)^{\otimes d+1} \otimes |\lambda\rangle\langle\lambda| \otimes (|0\rangle\langle 0|)^{\otimes n} \otimes |\lambda\rangle\langle\lambda| \otimes |z\rangle\langle z|$$

Choose R or G randomly to fed into D, before fed:

$$U_R = I^{\otimes(1+d+s)} \otimes R,\ U_D(\boldsymbol{\theta}_D) = D(\boldsymbol{\theta}_D) \otimes I^m$$

$$U_G(\boldsymbol{\theta}_G) = I^{\otimes(1+d+s)} \otimes G(\boldsymbol{\theta}_G)$$

After fed:

$$\rho_\lambda^R = U_R \rho_\lambda^0(z) U_R^\dagger$$

$$\rho_\lambda^G(\boldsymbol{\theta}_G, z) = U_G(\boldsymbol{\theta}_G)\rho_\lambda^0(z)U_G^\dagger(\boldsymbol{\theta}_G)$$

After classify:

$$\rho_\lambda^{DR}(\boldsymbol{\theta}_D) = U_D(\boldsymbol{\theta}_D)\rho_\lambda^R U_D^\dagger(\boldsymbol{\theta}_D)$$

$$\rho_\lambda^{DG}(\boldsymbol{\theta}_D, \boldsymbol{\theta}_G) = U_D(\boldsymbol{\theta}_D)\rho_\lambda^G(\boldsymbol{\theta}_G, z)U_D^\dagger(\boldsymbol{\theta}_D)$$

[2.11] Dallaire-Demers, P. L., & Killoran, N. (2018). Quantum generative adversarial networks. *Physical Review A*, *98*(1), 012324.

2.3.3. Quantum deep learning

Quantum Generative adversarial net (QGAN)

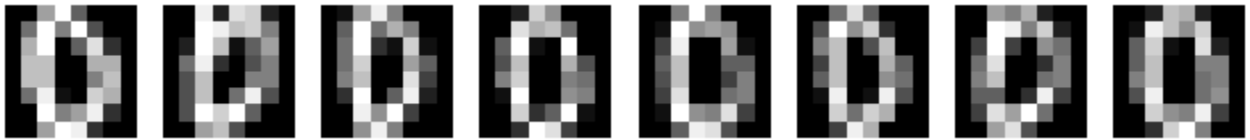
Quantum cost function

$$V(\boldsymbol{\theta}_D, \boldsymbol{\theta}_G) = \frac{1}{2} + \frac{1}{2\Lambda}\sum_{\lambda}\left(\cos^2(\phi)\mathrm{Tr}\left(Z\rho_{\lambda}^{DR}(\boldsymbol{\theta}_R)\right) - \sin^2(\phi)\mathrm{Tr}\left(Z\rho_{\lambda}^{DR}(\boldsymbol{\theta}_R, \boldsymbol{\theta}_D)\right)\right)$$

Here the angle ϕ parametrizes the probability that R or G is used as a source

Assume that $\phi = \frac{\pi}{4}$, mean p(R as source) $= p$(G as source)

$$V(\boldsymbol{\theta}_D, \boldsymbol{\theta}_G) = \frac{1}{2} + \frac{1}{4\Lambda}\sum_{\lambda}\mathrm{Tr}\left(\rho_{\lambda}^{DR}(\boldsymbol{\theta}_D) - \rho_{\lambda}^{DG}(\boldsymbol{\theta}_D, \boldsymbol{\theta}_G)\right)Z$$



2.3.3. Quantum deep learning

Quantum Generative adversarial net (QGAN)

In case G perfectly generates data source, $G(\boldsymbol{\theta}_G^*) = R$

$$\Rightarrow \Pr(Success\ D(\boldsymbol{\theta}_D)|\boldsymbol{\theta}_D) = \frac{1}{2} \Rightarrow \boxed{\nabla_{\boldsymbol{\theta}_D} V = 0, \nabla_{\boldsymbol{\theta}_G} V = 0}$$

Stop training

D is bounded by the purity function $C(\boldsymbol{\theta}_G) = \mathrm{tr}\left(\rho_R \rho_G(\boldsymbol{\theta}_G)\right)$:

$$\frac{1}{2} C(\boldsymbol{\theta}_G) \leq \Pr(Success\ D(\boldsymbol{\theta}_D)|\boldsymbol{\theta}_G) \leq 1 - \frac{1}{2} C(\boldsymbol{\theta}_G)$$

$\Rightarrow$ Increase gradient value when $\Pr \approx 1/2$

2.3.4. Quantum deep learning

QNN as compressor

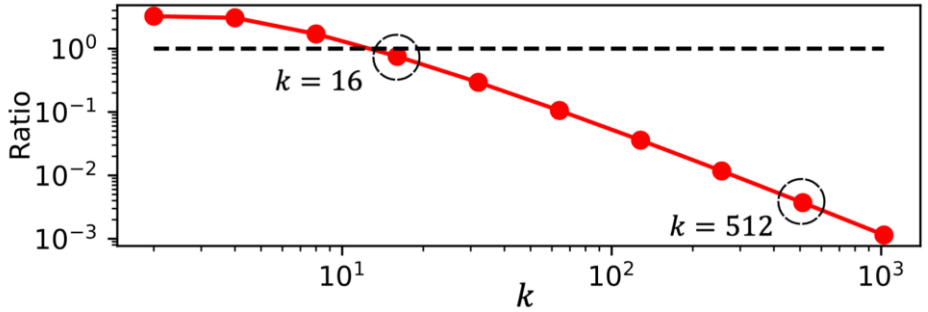
A digital image can be compressed by lossy/lossless compression algorithm

Classical image compression	**QIC**
Trade-off between loss and compression ratio	Trade-off between loss and compression time
Fixed compression ratio	Changeable compression ratio by block size k

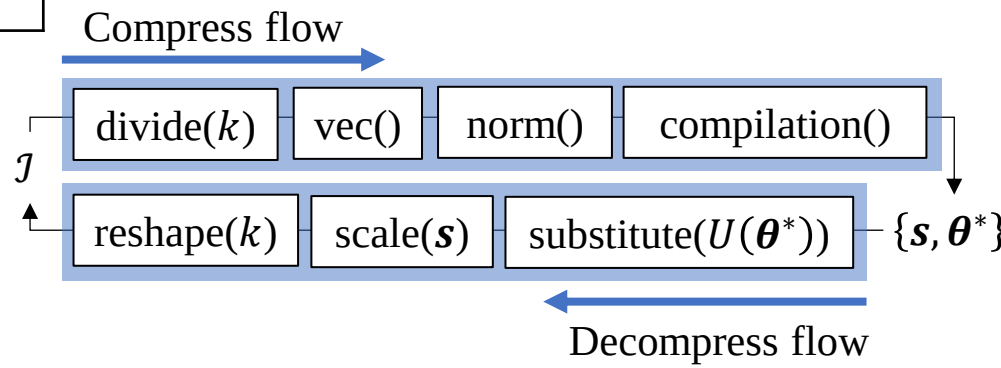
Compress

Decompress

Phases from parameterized gates

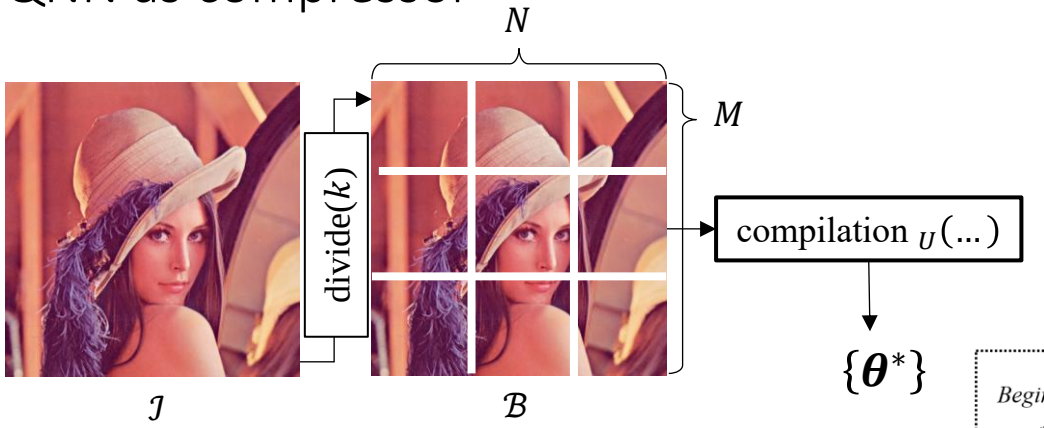


The ratio (data size) between compressed image and original image. The dotted line is ratio = 1.

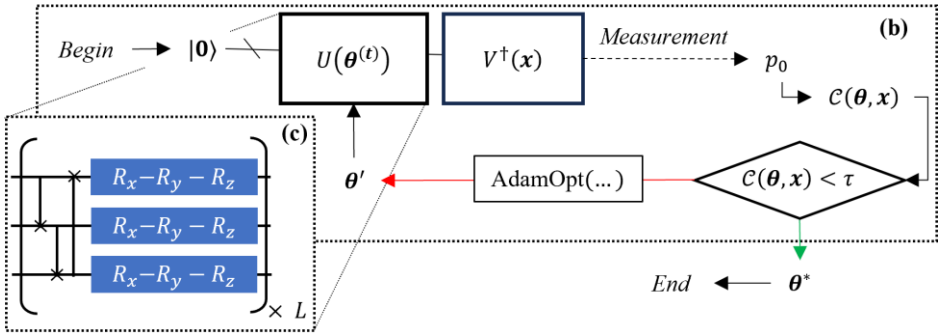


2.3.4. Quantum deep learning

QNN as compressor



The compilation algo's target is that finding $\boldsymbol{\theta}^*$ from an initial $\boldsymbol{\theta}^{(0)}$ to make sure that x is encoded in the parameterized circuit U.



Any 2^n- dim vector can be encoded as n – qubit state preparation circuit

1. An image is divided into $k \times k$ blocks $\{\mathfrak{B}_{i,j}\}$.
2. For block in blocks (looping $P_N \times P_M$ times):
   - 2.1. $x \leftarrow norm(vec(block))$ → Normalize and vectorize functions
   - 2.2. $\boldsymbol{\theta}^* \leftarrow compilation_U(\boldsymbol{\theta}^{(0)}, x)$ → Quantum compilation algorithms

2.4. Additional information

Power of QML

We will focus on PQC which is the main computation in QML

Encoder: use a very small qubit ($\log_2 N$) to loading / encoding N – dimensional normalized x.

Operator: like classical NN, the structure of $U(\boldsymbol{\theta})$ decides the model extract feature / process information good or bad.

Decoder: quantum information → classical information.

2.4. Additional information

Challenge in QML field

- Standard architechture?
- Trainability, Expressibility
- Lack of experiment resources (not enough simulated #qubit to transform toy dataset → real dataset)
- Hype in QML: some applied QML models are actually worse than classical models.
- Other problems

[2.12] Joseph Bowles, Shahnawaz Ahmed, & Maria Schuld. (2024). Better than classical? The subtle art of benchmarking quantum machine learning models.
[2.13] Schuld, M., & Killoran, N. (2022). Is Quantum Advantage the Right Goal for Quantum Machine Learning?. PRX Quantum, 3, 030101.
[2.14] Jens Eisert, & John Preskill. (2025). Mind the gaps: The fraught road to quantum advantage.

3. Quantum simulation

3.1. State vector

3.2. Stabilizer formalism

3.3. Tensor network (Matrix product state)

3.1. Quantum simulation

State vector

A n – qubit quantum state can be in the form:

$$|\psi^{(t)}\rangle = \begin{bmatrix} \alpha_0^{(t)} \\ \vdots \\ \alpha_{N-1}^{(t)} \end{bmatrix}$$

where $N = 2^n$ and $\alpha_j^{(t)} \in \mathbb{C}$ $\left\| \alpha_j^{(t)} \right\| \leq 1$

The number of amplitudes grows exponentially based on #Qubits

A quantum gate update amplitudes $\left\{\alpha_j^{(t)}\right\}$:

$$g(w,\theta): \left\{\alpha_j^{(t)}\right\} \rightarrow \left\{\alpha_j^{(t+1)}\right\}$$

Traversing on a long vector

where w and θ are wires and parameter (if applicable).

Our target is to compute the evolved (final) state:

$$|\psi^{(m)}\rangle = U(\boldsymbol{\theta})|\psi^{(0)}\rangle$$

Both $|\psi^{(t+1)}\rangle$ and $|\psi^{(t)}\rangle$ need to be saved at the same time

under unitary U. U is composed from gates or sub-operators via tensor product/matrix multiplication

3.1. Quantum simulation

State vector

Tab. Comparison between variants (†) $\hat{n}_j \in [1, 2]$

Matrix-vector multiplication

Op	Wave-function	Density-matrix
Mul/gate	$2^{n+\hat{n}_j+1}$ (†)	2^{2n}
Add/gate	$2^{n+\hat{n}_j+1}$ (†)	2^{2n-2}

The final state is calculated through series of $\mathcal{W}$:

$$|\psi^{(m)}\rangle = \mathcal{W}(g_m) \dots \mathcal{W}(g_2)\mathcal{W}(g_1)|\psi^{(0)}\rangle$$

The transition function $\mathcal{W}(g_j)$ updates amplitudes $\{\alpha_j\}$ on the $(w_0)^{th}$ qubit:

$$\mathcal{W}(g): \begin{bmatrix} \alpha_{s_i}^{(t)} \\ \alpha_{s_i+2^w}^{(t)} \end{bmatrix} \rightarrow g \begin{bmatrix} \alpha_{s_i}^{(t)} \\ \alpha_{s_i+2^w}^{(t)} \end{bmatrix} = \begin{bmatrix} \alpha_{s_i}^{(t+1)} \\ \alpha_{s_i+2^w}^{(t+1)} \end{bmatrix}$$

Algorithm 1 $\mathcal{W}(\{\alpha_j^{(t)}\}, g, \dots)$: Operation of gates from amplitude $head^{\text{th}}$ to amplitude $tail^{\text{th}}$ on n-qubit state.

Require: $\{\alpha_j^{(t)}\}$, target qubit $w_0 \in [0, n)$, $g = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \in \mathcal{G}$ or control-target qubit $w_0, w_1 \in [0, n), w_0 \neq w_1$, the head ($head$) and tail ($tail$) index of input amplitude.

Depend on type

Ensure: $\sum_j |\alpha_j^{(t)}|^2 = 1$

state $\leftarrow [\text{bin}(0, n), \text{bin}(1, n), \dots, \text{bin}(N-1, n)]$ ▷ $\text{bin}(i, k)$ is the function that represent i by k binary bit,

for $j \leftarrow [head, head+1, \dots, tail]$ **do**

[Single qubit gate]

cut $\leftarrow 2^{n-w_0-1}$

if state$[j][w_0] == 0$ **then**

$\alpha_j^{(t+1)} \leftarrow \alpha_j^{(t+1)} + a \times \alpha_j^{(t)}$

$\alpha_{j+\text{cut}}^{(t+1)} \leftarrow \alpha_{j+\text{cut}}^{(t+1)} + b \times \alpha_j^{(t)}$

else

$\alpha_j^{(t+1)} \leftarrow \alpha_j^{(t+1)} + d \times \alpha_j^{(t)}$

$\alpha_{j-\text{cut}}^{(t+1)} \leftarrow \alpha_{j-\text{cut}}^{(t+1)} + c \times \alpha_j^{(t)}$

[2-qubit gate]

cut $\leftarrow 2^{n-w_1-1}$

if state$[j][w_0] == 1$ **then**

if state$[j][w_1] == 1$ **then**

$\alpha_{j-\text{cut}}^{(t+1)} \leftarrow \alpha_{j-\text{cut}}^{(t+1)} + \alpha_j^{(t)}$

else

$\alpha_{j+\text{cut}}^{(t+1)} \leftarrow \alpha_{j+\text{cut}}^{(t+1)} + \alpha_j^{(t)}$

else

$\alpha_j^{(t+1)} \leftarrow \alpha_j^{(t)}$

return $\{\alpha_j^{(t+1)}\}$

Algorithm. The implementation of single- and multiple-qubit gates is unified into a common framework

3.2. Quantum simulation

Stabilizer formalism

A stabilizer state $|\psi\rangle$ can be represented as **stabilizer generators**:

$$G = \begin{pmatrix} \mathbb{P}_{n,0}^{(0)} \\ \vdots \\ \mathbb{P}_{n,n-1}^{(0)} \end{pmatrix} \xrightarrow{g^m} \begin{pmatrix} \mathbb{P}_{n,0}^{(m)} \\ \vdots \\ \mathbb{P}_{n,n-1}^{(m)} \end{pmatrix} \tag{3.1}$$

where $\mathbb{P}_{n,j}^{(0)} \equiv Z_{n,j} = \mathbb{I}^{\otimes j} \otimes Z \otimes \mathbb{I}^{\otimes(n-j-1)}$.

A generator may transform from a single Pauli string into a linear combination of Pauli strings:

$$\mathbb{P}_n = \sum \lambda_j P_{n,j} \text{ with } \lambda P_n = \lambda p_0 \otimes \cdots \otimes p_{n-1} \equiv \lambda p_0 \dots p_{n-1}; \tag{3.2}$$

$\lambda \in \mathbb{R}$ is the weight and P_n is the n-qubit Pauli string which is composed from n Pauli matrices $p_j \in \{I, X, Y, Z\}$.

Example. Consider a 3-qubit circuit $CX_{0,1}R_z\left(\frac{\pi}{3}\right)_0 SX_0|000\rangle$.

$$\begin{pmatrix} Z_{3,0} \\ Z_{3,1} \\ Z_{3,2} \end{pmatrix} \xrightarrow{SX_0} \begin{pmatrix} -YII \\ IZI \\ IIZ \end{pmatrix} \xrightarrow{R_z(\frac{\pi}{3})_0} \begin{pmatrix} \frac{1}{2}XII - \frac{\sqrt{3}}{2}YII \\ IZI \\ IIZ \end{pmatrix}$$

$$\xrightarrow{CX_{0,1}} \begin{pmatrix} \frac{1}{2}XXI - \frac{\sqrt{3}}{2}YXI \\ ZZI \\ IIZ \end{pmatrix}$$

Table. Quantum gate operations

g	$\lambda P^{(t)}$	$\lambda P^{(t+1)}$
SX	Y	Z
	Z	$-Y$
$R_z(\theta)$	X	$\cos(\frac{\theta}{2})X + \sin(\frac{\theta}{2})Y$
	Y	$\cos(\frac{\theta}{2})Y - \sin(\frac{\theta}{2})X$

$$\rho \equiv |\psi\rangle\langle\psi| = \frac{1}{2^n}\prod_{j=1}^{n}\left(I^{\otimes n} + \mathbb{P}_{n,j}\right) \Rightarrow p_{j,k} = \text{tr}\left(\left[\frac{1}{2}\left(I^{\otimes n} + Z_k\right)\right]\rho\right)$$

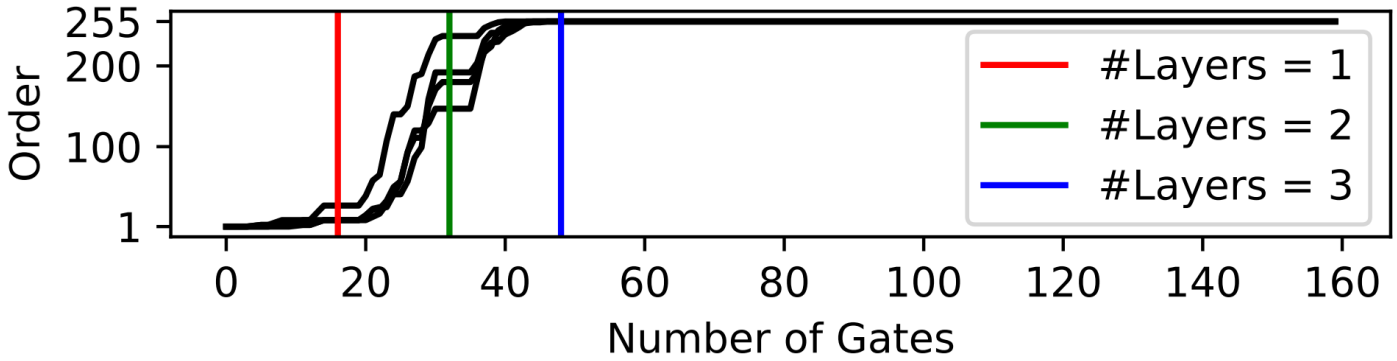
[3.1] C. Gidney, "Stim: a fast stabilizer circuit simulator," Quantum, vol. 5, p. 497, Jul. 2021

3.2. Quantum simulation

Challenges in stabilizer formalism

(a) Applying non-Clifford gates **increases the stabilizer rank** (non-stabilizerness) of a stabilizer state: higher stabilizer rank, harder for simulate.

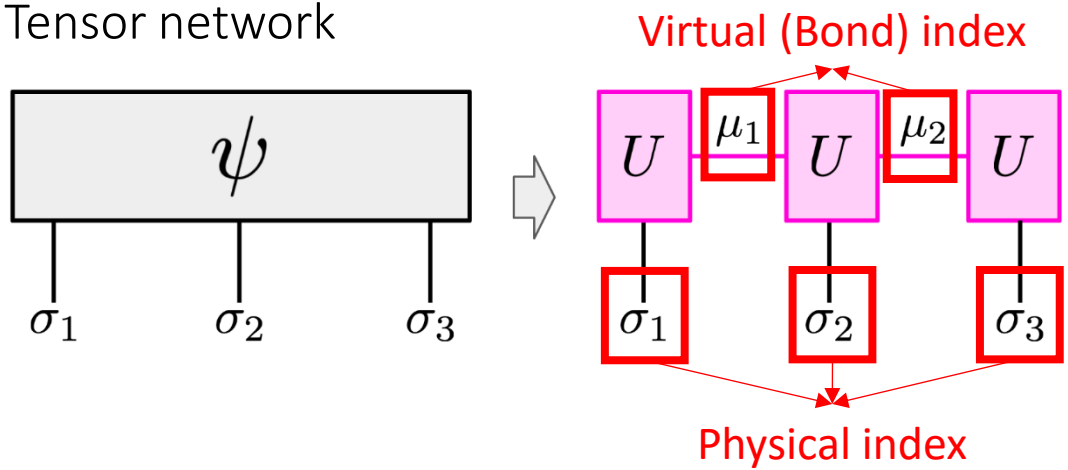
(b) The current stabilizer formalism are sequentially updated gate by gate: limited the power of GPU.



The stabilizer rank on four generators is shown as four black lines. We are considering the worst cases by using 4-qubit $|XYZ + chain\rangle$ ansatz.

3.3. Quantum simulation

Tensor network



We present 2^n statevector as $\underbrace{(2,2,\dots,2)}_{n}$ tensor

For qubit 0, we reshape $|\psi\rangle$ as row = (i_1), col = $(i_2,\dots,i_n)$, $\psi_{i_1,(i_2,\dots,i_n)} \xrightarrow{SVD} U \quad SV^\dagger$, $A_1^{i_1}$

For qubit 1, we reshape $SV^\dagger$ as row = $(bond \times i_2)$, col = $(i_3,\dots,i_n)$ $\xrightarrow{SVD}$ $A_2^{i_2} SV^\dagger$

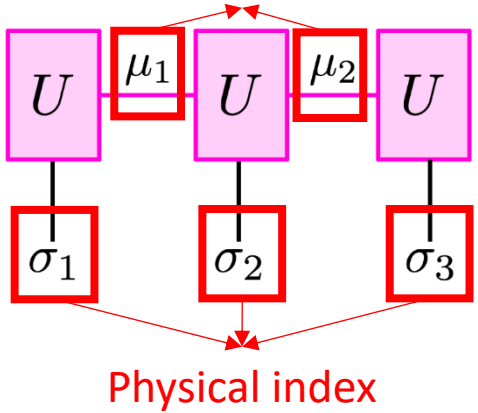
…

Finally, we collect $\left\{A_1^{i_1}, A_2^{i_2}, \dots, A_n^{i_n}\right\}$

3.3. Quantum simulation

Tensor network

$$|W_4\rangle = 1/\sqrt{4}(|0001\rangle + |0010\rangle + |0100\rangle + |1000\rangle)$$



$$|W_4\rangle \xrightarrow{(lvbond\times|\sigma_1|)\times\frac{2^n}{lvbond\times|\sigma_1|}} \psi_{i_1,(i_2 i_3 i_4)} \xrightarrow{SVD} U\Sigma V^\dagger$$

$$U \quad \Sigma \quad V^\dagger$$

$$2\times 2 \quad 2\times 2 \quad 2\times 8$$

$$\Sigma = \begin{bmatrix} \sqrt{3}/2 & 0 \\ 0 & 1/2 \end{bmatrix}$$

$$U(\mu_0, \sigma, \mu_1) \rightarrow \begin{cases} A_{0,\mu_0,\mu_1} = \begin{bmatrix} \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{6}} \end{bmatrix} \\ A_{1,\mu_0,\mu_1} = \begin{bmatrix} \frac{1}{\sqrt{3}} & \frac{-2}{\sqrt{6}} \end{bmatrix} \end{cases}$$

(l)vbond = 1

(r)vbond = {1,2}

In qubits system: $\sigma_j = \{0,1\}$

For next iteration $B = \Sigma V^\dagger$

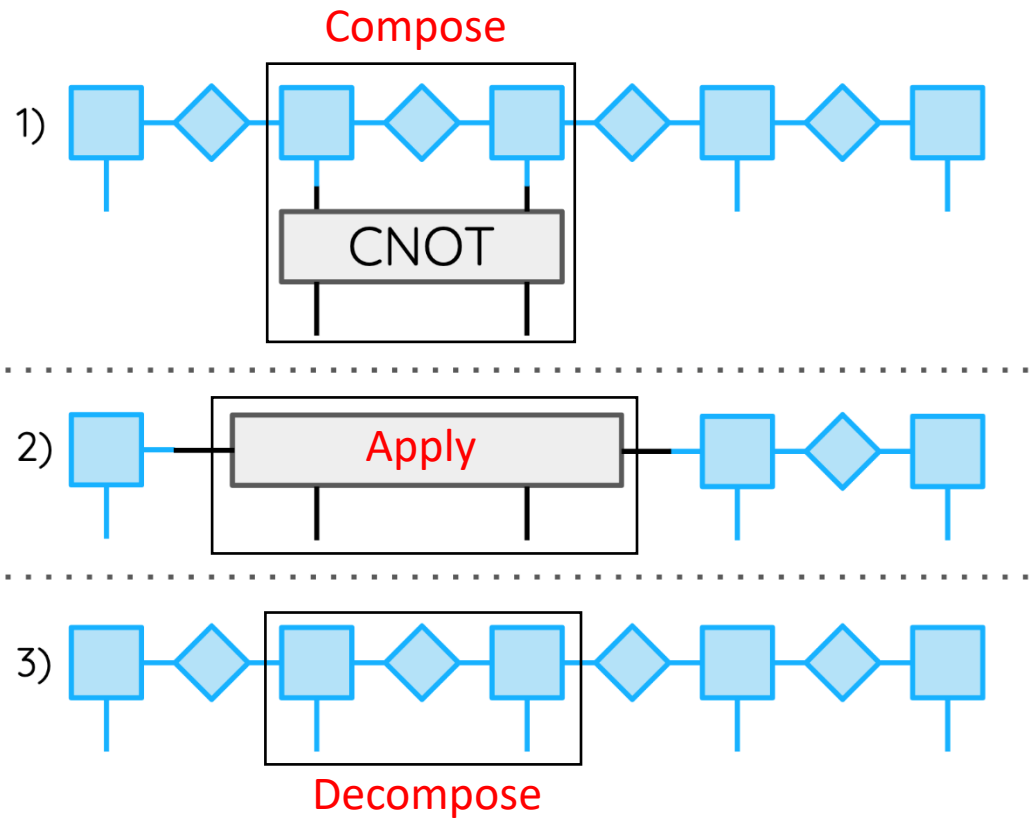
Collect A_{σ_0,μ_0,μ_1}

[3.2] Siddhartha Patra, Saeed S. Jahromi, Sukhbinder Singh, Roman Orus "Efficient tensor network simulation of IBM's largest quantum processors" arXiv:2309.15642, 2023.

[3.3] H. C. Jiang, Z. Y. Weng, T. Xiang "Accurate determination of tensor network state of quantum lattice models in two dimensions" arXiv:0806.3719, 2008.

3.3. Quantum simulation

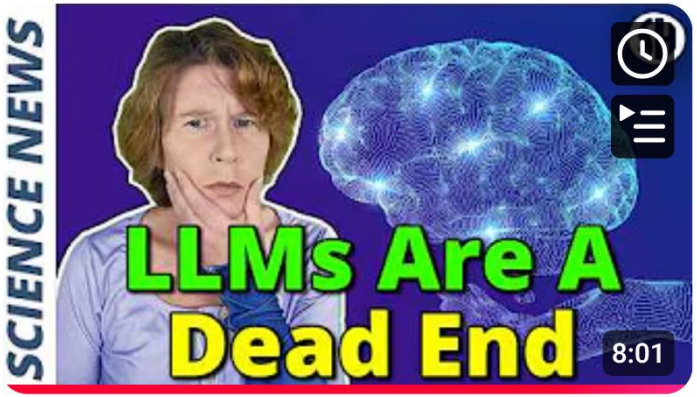
Tensor network: gate application



A local gate $U = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$ act on wire j and transform only A_j

$$A'_{0,j} = aA_{0,j} + bA_{1,j}$$
$$A'_{1,j} = cA_{0,j} + dA_{1,j}$$

A non-local gate CU act on wires $j, k, \ldots$ which is a 'bit' complicated



Keep skepticism in your research !

Thank you for everything!