Brookhaven
National Laboratory
U.S. DEPARTMENT OF
ENERGY
Quantum Utility on Near-Term Quantum Computers
Kwangmin Yu
May 28th, 2026
@BrookhavenLab

Introduction

Supremacy, Advantage, and Utility

- Quantum Supremacy:
  - Demonstrating that a programmable quantum computer can solve a problem that no classical computer can solve in any feasible amount of time, **irrespective** of the **usefulness** of the problem (Preskill, 2012).
  - Neill, Charles, et al., Science 360.6385 (2018): 195-199.
  - Arute, Frank, et al., Nature 574.7779 (2019): 505-510.
- Quantum Advantage
  - The experimental demonstration of a quantum algorithm solving a real-world problem on a quantum computer **outperforms ANY classical algorithm** running on any classical computer.
  - The experimental demonstration of a quantum algorithm solving a **useful real-world problem** beyond the capabilities of the best known classical algorithms and supercomputers.
- Quantum Utility
  - Emphasizes the **practical usefulness** of quantum computers in scientific and industrial applications.
  - https://www.ibm.com/quantum/blog/what-is-quantum-utlity
  - Kim, Youngseok, et al., Nature 618.7965 (2023): 500-505.
  - Timothy Proctor, et al., ArXiv 2407.08828 (2024)

Challenge of Quantum Utility on NTQC

1. Noise-dominant execution regime: Not a small noise perturbation regime
   - Gate errors accumulate under realistic circuit depth
   - Decoherence limits coherent evolution
   - Error mitigation reduces bias, but does not restore signal
2. Hardware-limited circuit expressivity: Implemented circuit ≠ Designed circuit
   - Finite coherence time restricts circuit depth
   - Connectivity constraints induce SWAP overhead / Limited number of qubits
   - Native gate set biases circuit structure
3. Compilation-induced distortion: Compiler MUST be part of the physics problem
   - Logical circuit → transpiled hardware circuit
   - Optimization target ≠ physical fidelity
   - Effective noise increases after mapping

➢ **These challenges imply that achieving quantum utility is not an algorithmic problem alone, but a co-design problem across the full stack.**

Co-Design Paradigm for Quantum Utility

- Quantum utility emerges from full-stack co-design across hardware, compilation, algorithm, and noise-aware circuit design.
- Quantum utility requires hardware–software–algorithm co-design.
- **Hardware & Noise-aware** circuit design is structural, not optional.
- Compilation must be integrated into noise-aware circuit design.
- Observables should guide algorithm construction under realistic noise.
- Design implication
  - No single-layer optimization is sufficient
  - Execution strategy is part of circuit design
  - Utility emerges from integrated stack behavior
- **Hardware & Noise-aware** design for Quantum Utility on NTQC

What Does Quantum Utility Look Like in Practice?

Time Evolution of Hamiltonian

- Time evolution of Hamiltonian, H, is given by

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

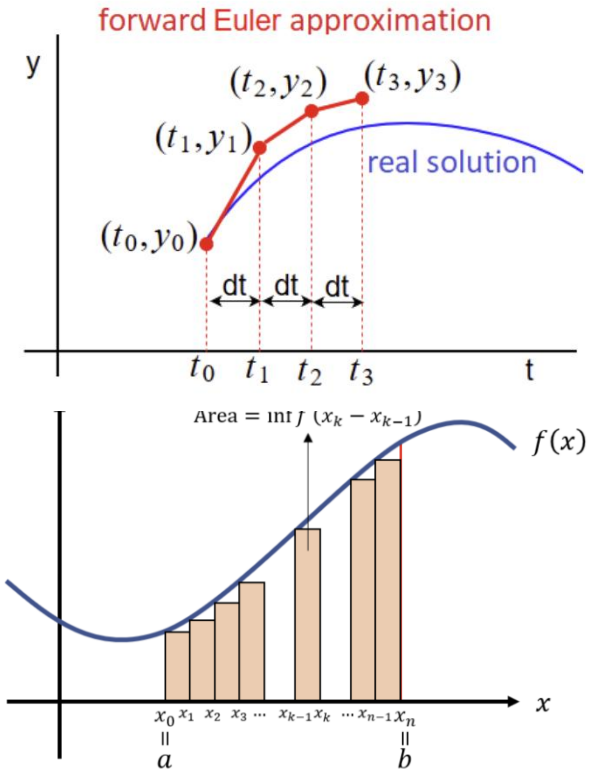
- We determine the time evolution of an observable $\hat{O}$, with respect to the time-evolved state $|\psi(t)\rangle$ as $\langle\psi(t)|\hat{O}|\psi(t)\rangle$.
- We consider Nèel's initial state $|\psi_{Neel}(t)\rangle = |\uparrow\downarrow\uparrow\downarrow\cdots\uparrow\downarrow\rangle = |0101\ldots01\rangle$.
- For the observable, our choice is the staggered magnetization that capture the antiferromagnetic ordering as follows:

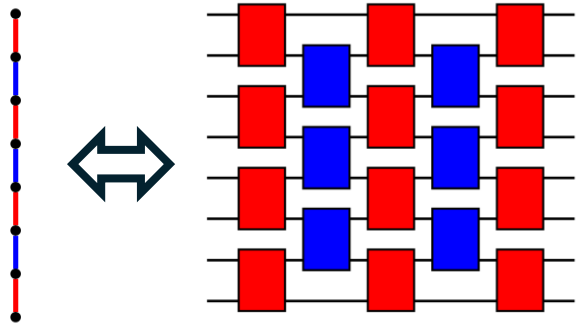
$$\hat{O}_{M_{st}} = \frac{1}{N}\sum_{i=1}^{N}(-1)^i S_i^z$$

- We determine the variation $\langle\psi_{Neel}(t)|\hat{O}_{M_{st}}|\psi_{Neel}(t)\rangle$ with time in the quantum computer.

Trotterization (Numerical Method)

- Also known as the product formulas or the splitting method.
- Let the Hamiltonian of the system of N qubits be given by $H = \sum_{i=1}^{L} H_i$.
- $U(t) = e^{-itH} = e^{-it\sum_{i=1}^{L} H_i} \approx (\prod_{i=1}^{L} e^{-\frac{it}{r}H_i})^r$
- Let $U_1(t) = (\prod_{i=1}^{L} e^{-\frac{it}{r}H_i})^r$. Then $U(t) = U_1(t) + O(\frac{t^2}{r})$
- When $U_2(t) = (\prod_{i=1}^{L} e^{-\frac{it}{2r}H_i} \prod_{i=L}^{1} e^{-\frac{it}{2r}H_i})^r$, then $U(t) = U_2(t) + O(\frac{t^3}{r^2})$
- In general, $U(t) = U_p(t) + O(\frac{t^{p+1}}{r^p})$
- p: Trotterization order, r: Trotter number
- Trotterization can preserve locality of the system. which can reduce the simulation cost.





Brick structure

Brookhaven National Laboratory

Trotterization of H_{XXZ}

- For the anisotropic Heisenberg Hamiltonian

$$H = \frac{J_1}{4} \sum_{j=0}^{N-1} \left(\sigma_j^x \sigma_{j+1}^x + \sigma_j^y \sigma_{j+1}^y + \Delta \sigma_j^z \sigma_{j+1}^z \right)$$

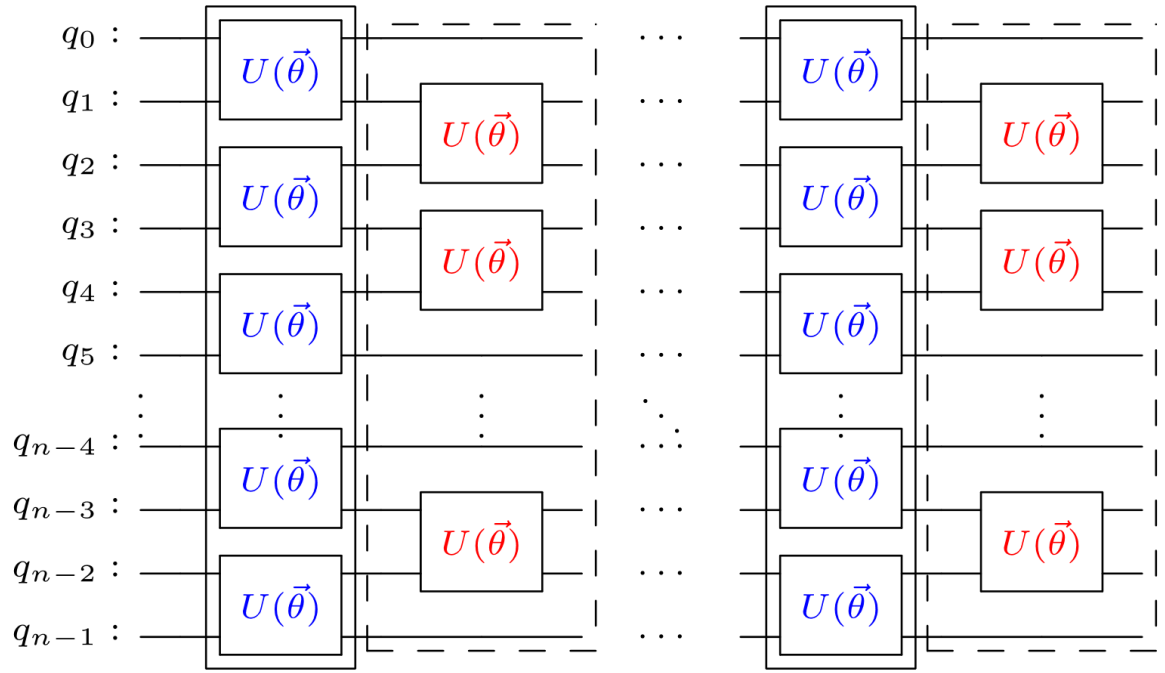
our basic building block is the following unitary operator:

$$U_j(\vec{\theta}) = \exp\left(-i \left(\frac{\theta_x}{2} \sigma_j^x \sigma_{j+1}^x + \frac{\theta_y}{2} \sigma_j^y \sigma_{j+1}^y + \frac{\theta_z}{2} \sigma_j^z \sigma_{j+1}^z \right) \right) \quad \vec{\theta} = (\theta_x, \theta_y, \theta_z) = (\tfrac{J_1}{2}\delta t,\ \tfrac{J_1}{2}\delta t,\ \tfrac{J_1}{2}\Delta\delta t)$$

- One Trotter step is defined in terms of the even layer, $U_e\left(\vec{\theta}\right)$, and the odd layer, $U_o\left(\vec{\theta}\right)$, as follows:

$$U_e(\vec{\theta}) = \left(\prod_{j=0}^{N/2-1} U_{2j}(\vec{\theta}) \right), \quad U_o(\vec{\theta}) = \left(\prod_{j=0}^{N/2-1} U_{2j+1}(\vec{\theta}) \right), \quad U(\vec{\theta}) = U_e(\vec{\theta}).U_o(\vec{\theta})$$

Trotterization of H_{XXZ} (first-order)

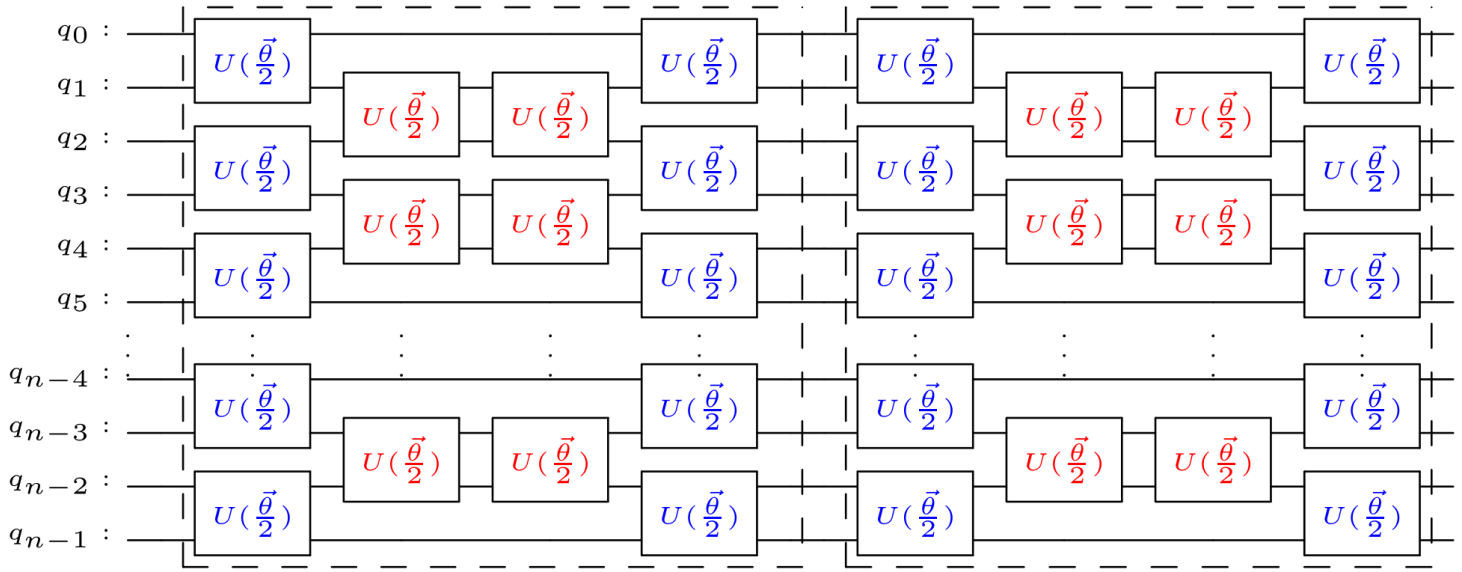


Trotterization of H_{XXZ} (first-order)

- The first-order Trotterization of the model with open boundary condition.
- One Trotter step is composed of the even layer and the odd layer.
- For a PBC, the odd layers have the two-qubit gates, $U_0\left(\vec{\theta}\right)$, between q_{n-1} and q_0.

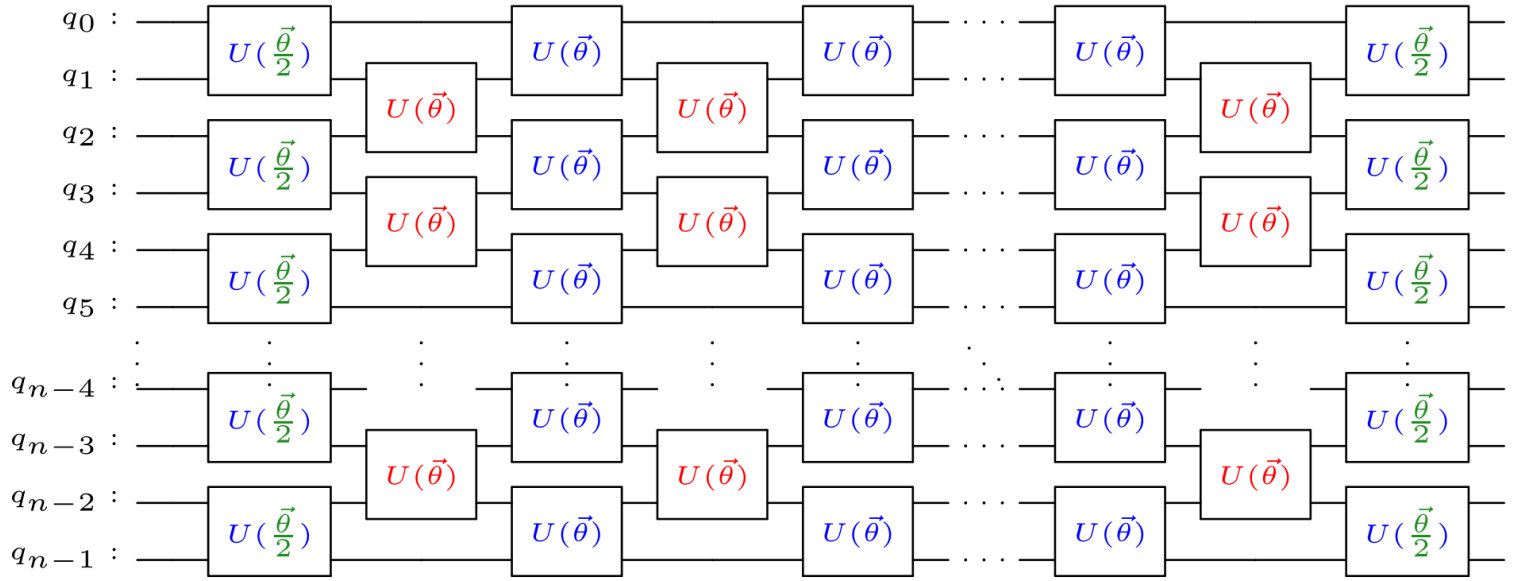
Trotterization of H_{XXZ} (second-order)

- $U_2(t) = (\prod_{i=1}^{L} e^{-\frac{it}{2r}H_i} \prod_{i=L}^{1} e^{-\frac{it}{2r}H_i})^r$



- The second-order Trotterization of the model with OBC.
- One Trotter step is compared of the even layer and the odd layer with parameter $\frac{\vec{\theta}}{2}$.
- For a PBC, the odd layers have the two-qubit gates, $U_0\left(\frac{\vec{\theta}}{2}\right)$, between q_{n-1} and q_0.

Trotterization of H_{XXZ} (second-order) Opt.



- The optimized second-order Trotterization of H_{iso} is obtained by the following identities:

$$U_e(\vec{\theta_1})U_e(\vec{\theta_2}) = U_e(\vec{\theta_1} + \vec{\theta_2}) \quad \text{and} \quad U_o(\vec{\theta_1})U_o(\vec{\theta_2}) = U_o(\vec{\theta_1} + \vec{\theta_2})$$

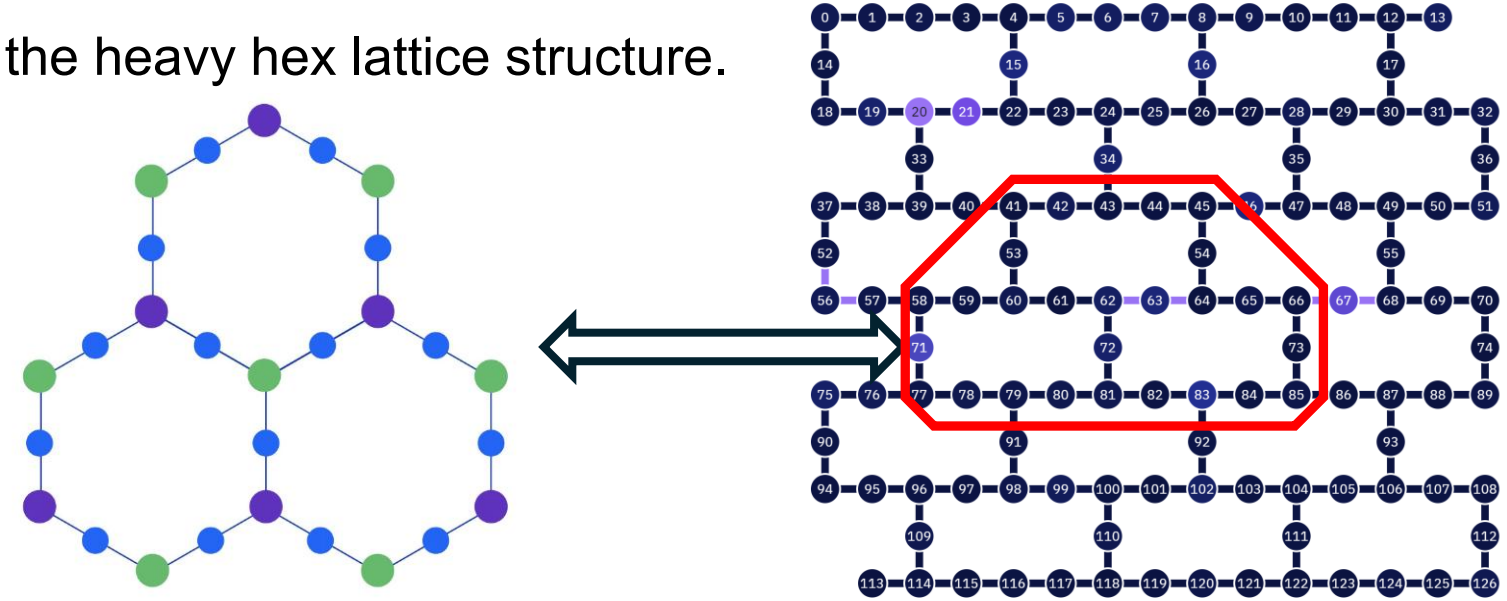
- We merge the adjacent odd layers, and the last even layer can be merged with the first even layer of the next Trotter step.
- 2L (1st order) => 4L (2nd order) => 2L+1 (2nd order optimized) when we have L Trotter steps.

Trotterization of J_1-J_2 Chain

- The J_1-J_2 chain Hamiltonian (frustrated spin 1/2 antiferromagnetic Heisenberg spin chain), $\boldsymbol{H}$, is

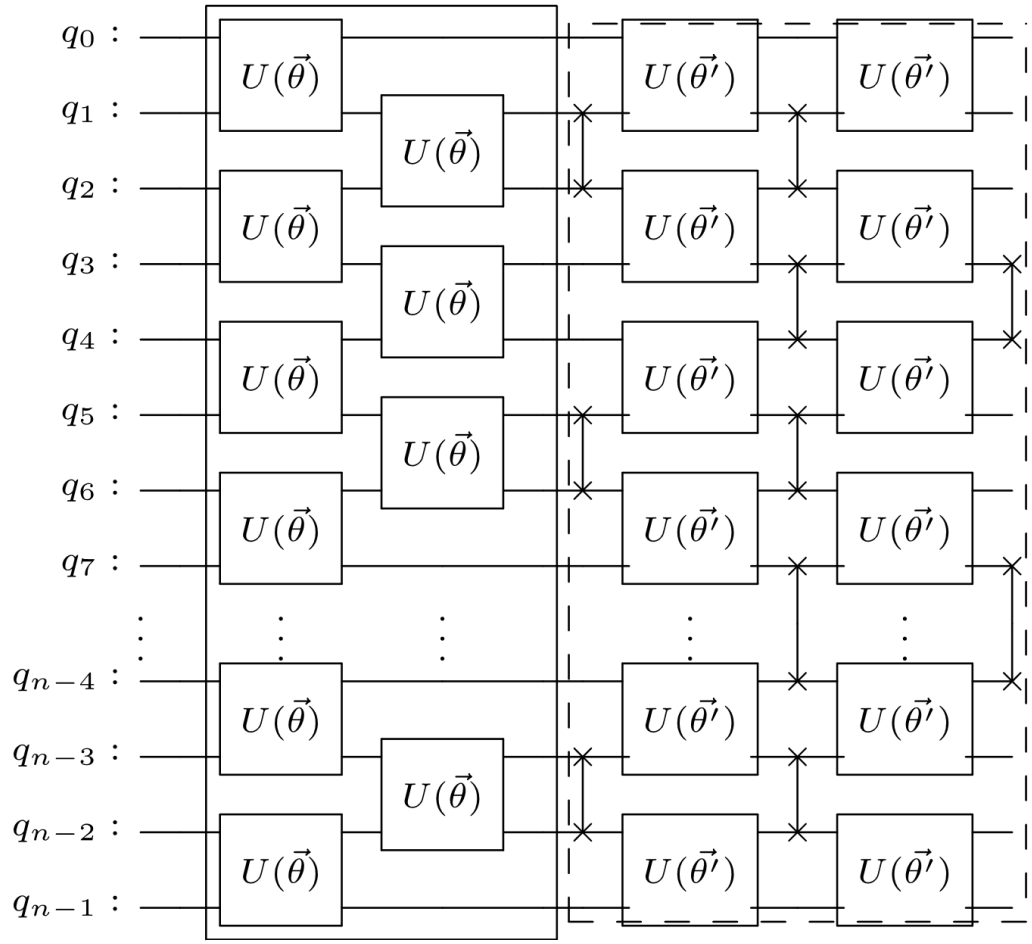
$$H = J_1 \sum_{i=1}^{N} \left(S_i^x S_{i+1}^x + S_i^y S_{i+1}^y + S_i^z S_{i+1}^z \right) + J_2 \sum_{i=1}^{N} \left(S_i^x S_{i+2}^x + S_i^y S_{i+2}^y + S_i^z S_{i+2}^z \right), \quad J_2 = J_1/2$$

- IBM's quantum computers have the heavy hex lattice structure.



- We implemented the next-nearest neighbor interaction with the nearest neighbor interaction in IBM's heavy hex lattice topology by interweaving SWAP gates.

Trotterization of $H_{J_1-J_2}$



Notation

n is the number of qubits and a multiple of 4.

The numbers represent qubit index from 0 to $n-1$.

$\bar{k}$ represent k modulo 4.

1: Place SWAP gates between $\bar{1}$ and $\bar{2}$.
2: Place the even layer of $\vec{\theta}'$ on the whole qubits.
3: Place SWAP gates between $\bar{1}$ and $\bar{2}$, and between $\bar{3}$ and $\bar{0}$.
   if PBC, place SWAP gate between $n-1$ and 0.
4: Place the even layer of $\vec{\theta}'$ on the whole qubits.
5: Place SWAP gates between $\bar{3}$ and $\bar{0}$.
   if PBC, place SWAP gate between $n-1$ and 0.

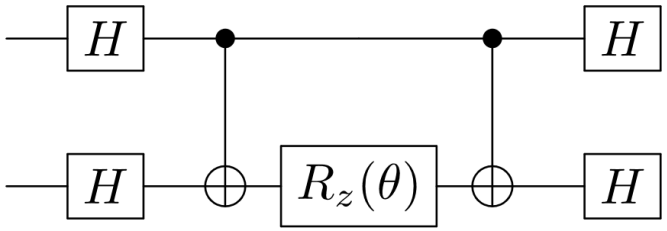
$$H = J_1 \sum_{i=1}^{N} \left(S_i^x S_{i+1}^x + S_i^y S_{i+1}^y + S_i^z S_{i+1}^z \right) + J_2 \sum_{i=1}^{N} \left(S_i^x S_{i+2}^x + S_i^y S_{i+2}^y + S_i^z S_{i+2}^z \right), \quad J_2 = J_1/2$$

Brookhaven National Laboratory

Circuit Implementation of U (one brick)

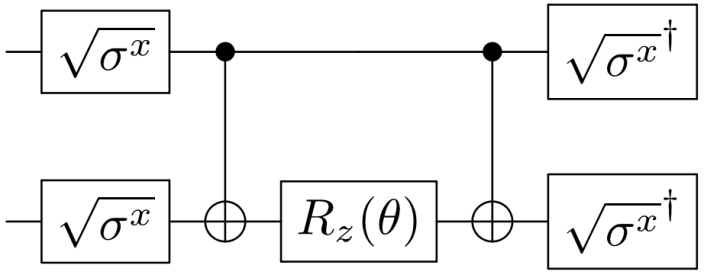
- $U_i(\vec{\theta}) = \exp\left(-i\frac{\theta}{2}\left(\sigma_i^x\sigma_{i+1}^x + \sigma_i^y\sigma_{i+1}^y + \sigma_i^z\sigma_{i+1}^z\right)\right), \theta = 2J_1\delta t$
- An efficient implementation of $U\left(\vec{\theta}\right)$ is crucial for the simulation.
- We build up our Trotterization circuit using two-qubit rotation gates, $R_{XX}(\theta)$, $R_{YY}(\theta)$, and $R_{ZZ}(\theta)$ as follows:

$R_{XX}(\theta) =$



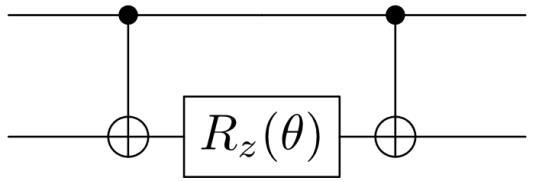
$= \exp\left(-i\frac{\theta}{2}X\otimes X\right) = \begin{pmatrix} \cos\left(\frac{\theta}{2}\right) & 0 & 0 & -i\sin\left(\frac{\theta}{2}\right) \\ 0 & \cos\left(\frac{\theta}{2}\right) & -i\sin\left(\frac{\theta}{2}\right) & 0 \\ 0 & -i\sin\left(\frac{\theta}{2}\right) & \cos\left(\frac{\theta}{2}\right) & 0 \\ -i\sin\left(\frac{\theta}{2}\right) & 0 & 0 & \cos\left(\frac{\theta}{2}\right) \end{pmatrix}$,

$R_{YY}(\theta) =$



$= \exp\left(-i\frac{\theta}{2}Y\otimes Y\right) = \begin{pmatrix} \cos\left(\frac{\theta}{2}\right) & 0 & 0 & i\sin\left(\frac{\theta}{2}\right) \\ 0 & \cos\left(\frac{\theta}{2}\right) & -i\sin\left(\frac{\theta}{2}\right) & 0 \\ 0 & -i\sin\left(\frac{\theta}{2}\right) & \cos\left(\frac{\theta}{2}\right) & 0 \\ i\sin\left(\frac{\theta}{2}\right) & 0 & 0 & \cos\left(\frac{\theta}{2}\right) \end{pmatrix}$,

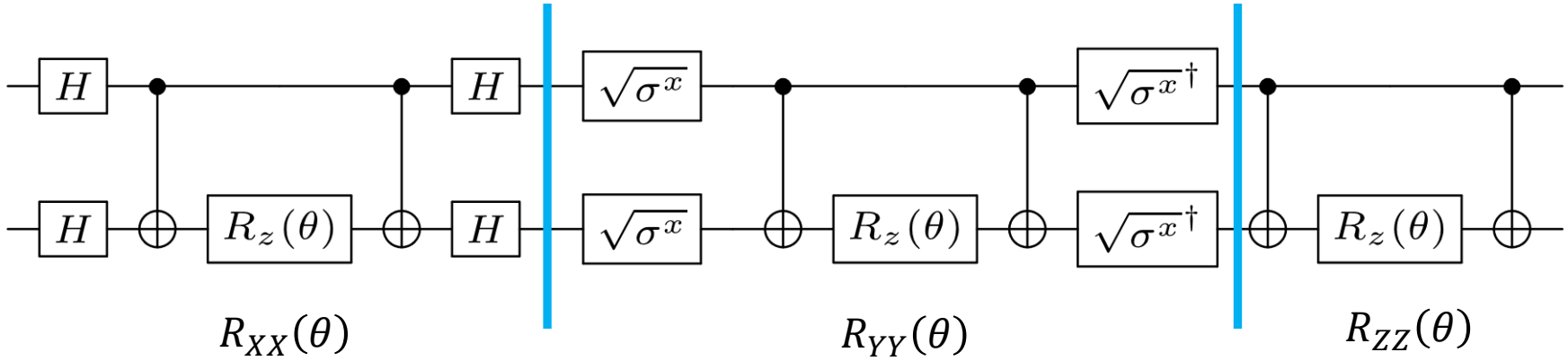
$R_{ZZ}(\theta) =$



$= \exp\left(-i\frac{\theta}{2}Z\otimes Z\right) = \begin{pmatrix} e^{-i\frac{\theta}{2}} & 0 & 0 & 0 \\ 0 & e^{i\frac{\theta}{2}} & 0 & 0 \\ 0 & 0 & e^{i\frac{\theta}{2}} & 0 \\ 0 & 0 & 0 & e^{-i\frac{\theta}{2}} \end{pmatrix}$

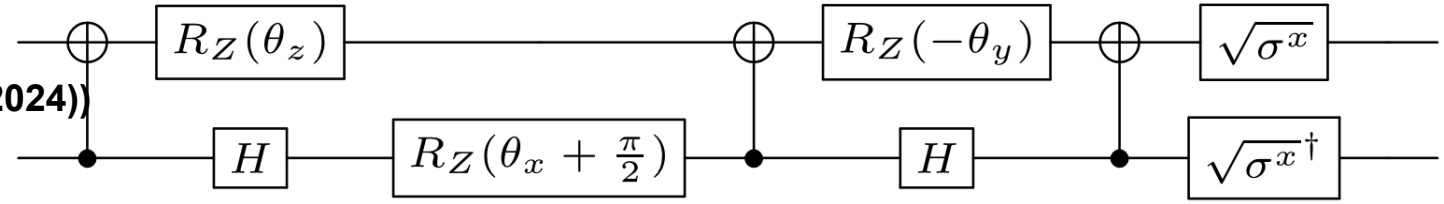
Circuit Implementation (Optimization)

- The unitary operator $U\left(\vec{\theta}\right)$ is implemented as:



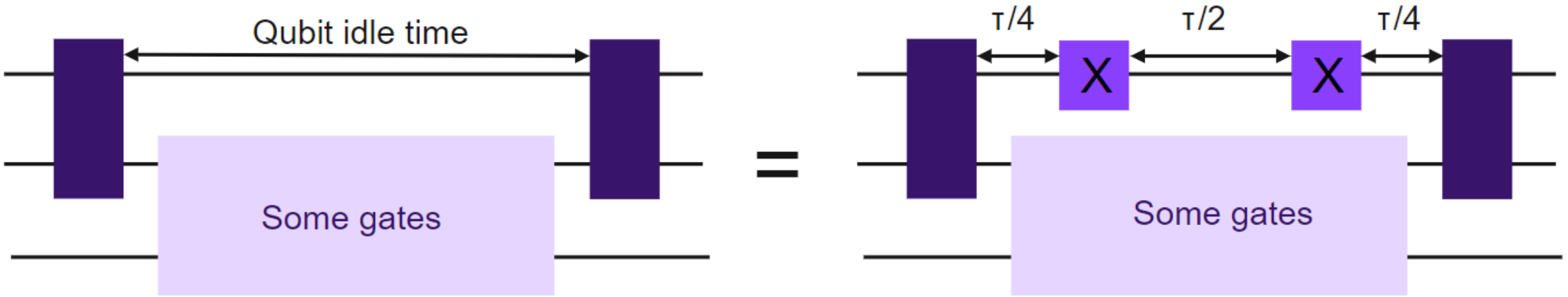
- This implementation has **6 CNOT** gates and **13 circuit depth**.
- This is optimized as:

(Refer to Zhang et al., Adv. Quantum Technol. 7.4 (2024))

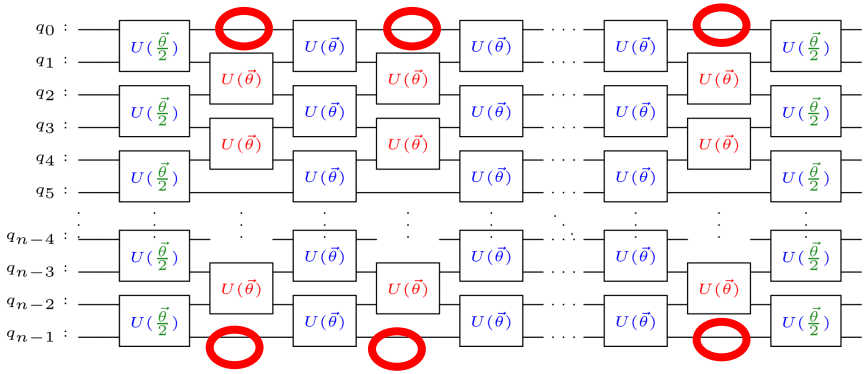


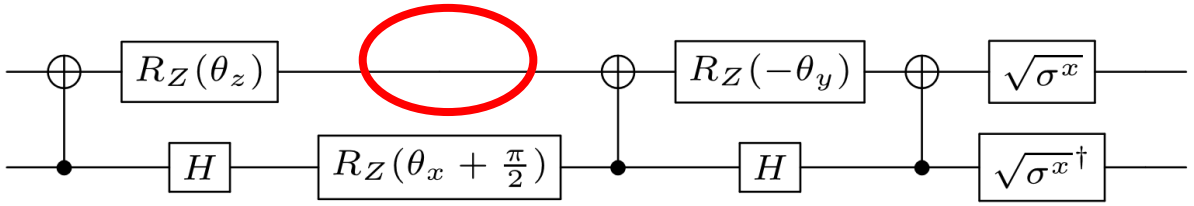
- The optimized circuit has **3 CNOT** gates and **7 circuit depth**.
- Reducing the number of CNOT gates as well as the circuit depth is essential to reduce the overall noise of the simulation on noisy quantum computers.

Noise-Aware Circuit Design (DD)

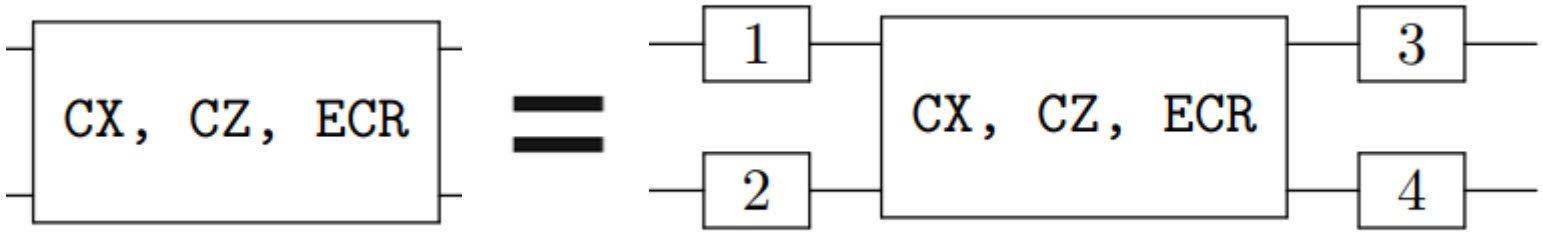


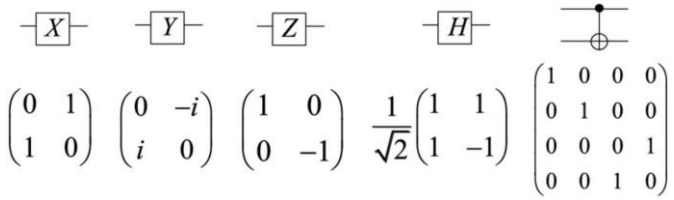
- Idle qubits may undergo damping and dephasing error due to T1-relaxation and T2-dephasing.
- Dynamical Decoupling (DD) can suppress those errors.
- Applying DD involves dressing the idle times between gates on a qubit with some choice of DD sequence.





Noise-Aware Circuit Design (PT)





- Pauli twirling is a method averaging out the off-diagonal coherent errors of the circuits in the Pauli basis.
- Pauli-twirling is a quantum error mitigation technique that uses randomization to noise-shape coherent error into stochastic errors, by combining the results from many random, but logically equivalent circuits, together.
- Search all possible Pauli gate sets satisfying the above equality and duplicate a given circuit with the logically equivalent circuits added by the Pauli gates. After that averaging all the duplicated circuit results.
- Kim, Y., Wood, C.J., Yoder, T.J. *et al.* Scalable error mitigation for noisy quantum circuits produces competitive expectation values. *Nat. Phys.* **19**, 752–759 (2023). https://doi.org/10.1038/s41567-022-01914-3
- **In our experiments, we searched all the combinations of Pauli gates that are mathematically identical up the global phase with only the Clifford gate (ECR or CZ gate in our experiments) out of 256(=4^4) cases. Finally, we obtained 16 combinations of {I, X, Y, Z}.**
- **Duplicated 10 copies of the base quantum circuit.**
- **In the 10 copies, we applied the 16 combination randomly to all ECR (or CZ) gates.**

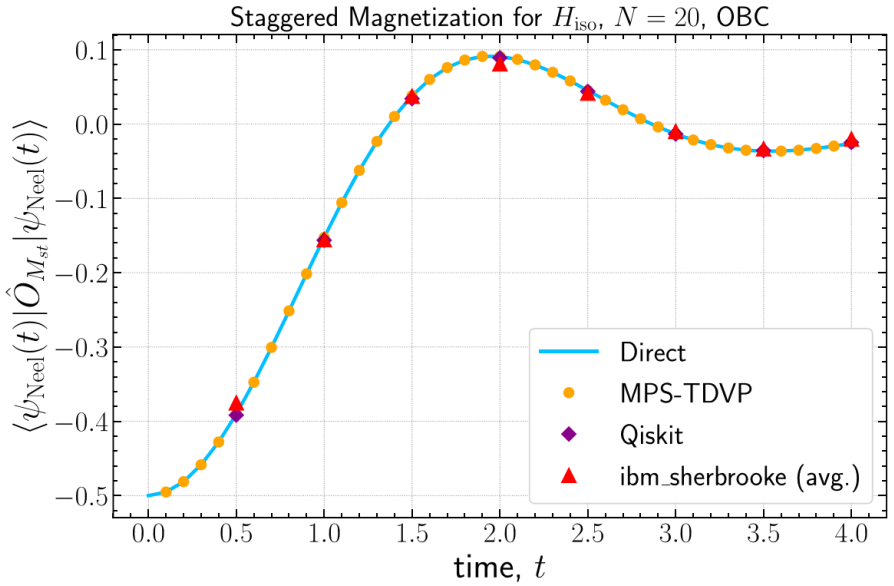
Results

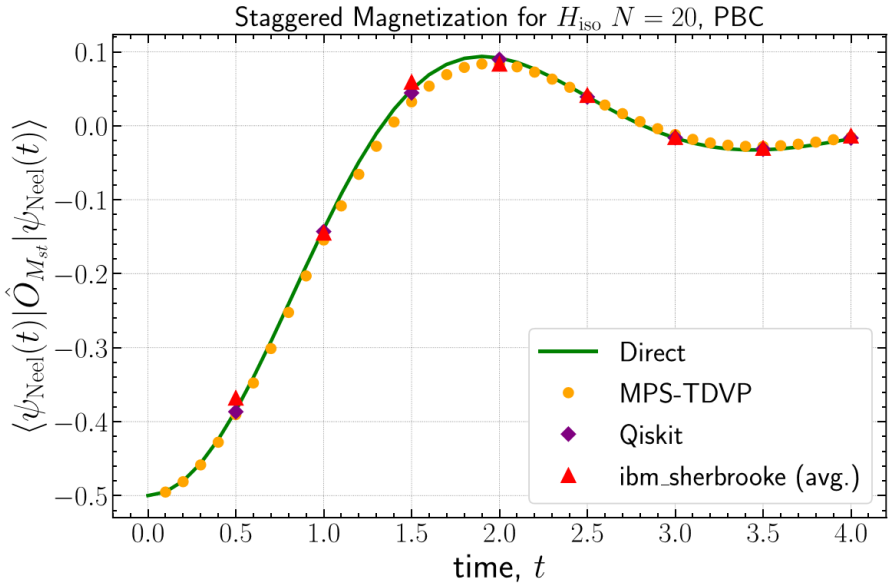
Results: N=20 sites (qubits)

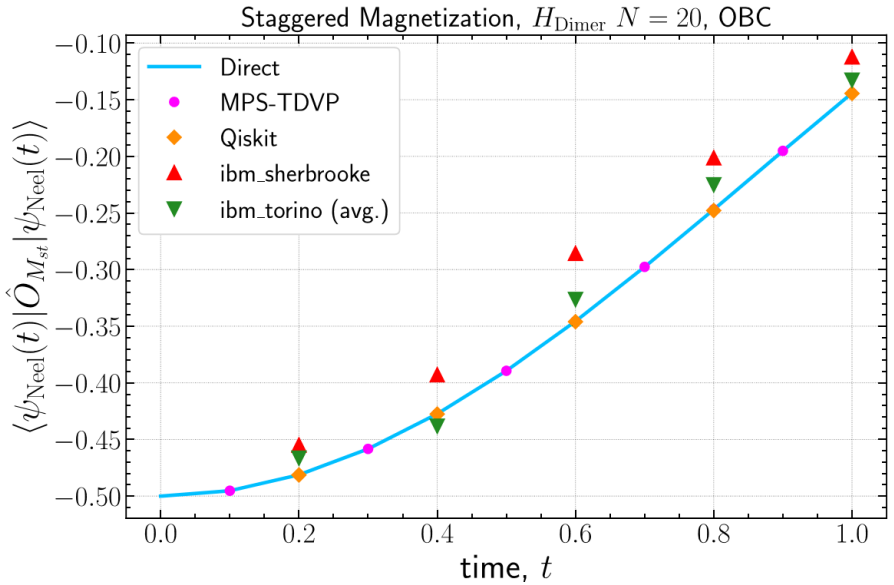
$\boldsymbol{H_{XXZ}}$

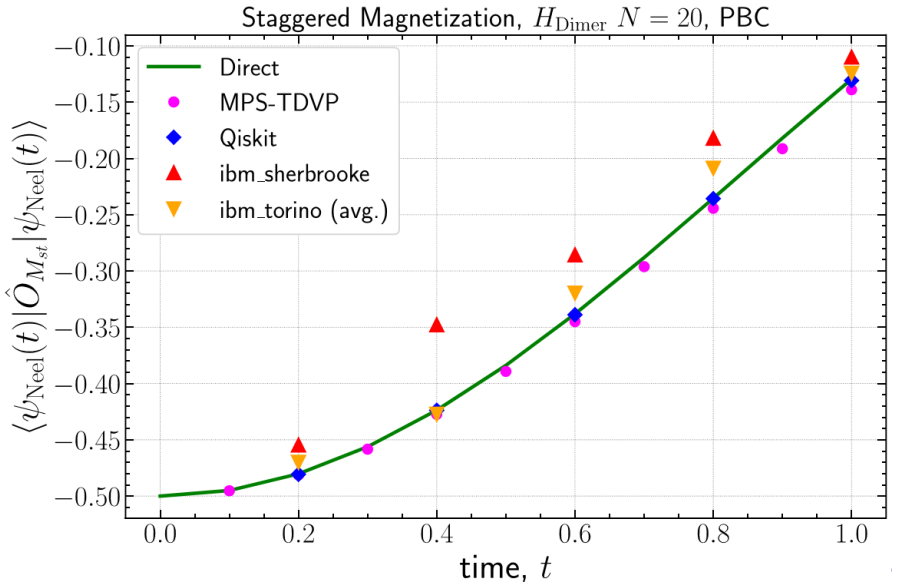
$$\hat{O}_{M_{st}} = \frac{1}{N}\sum_{i=1}^{N}(-1)^i S_i^z$$

$\boldsymbol{H_{J_1-J_2}}$





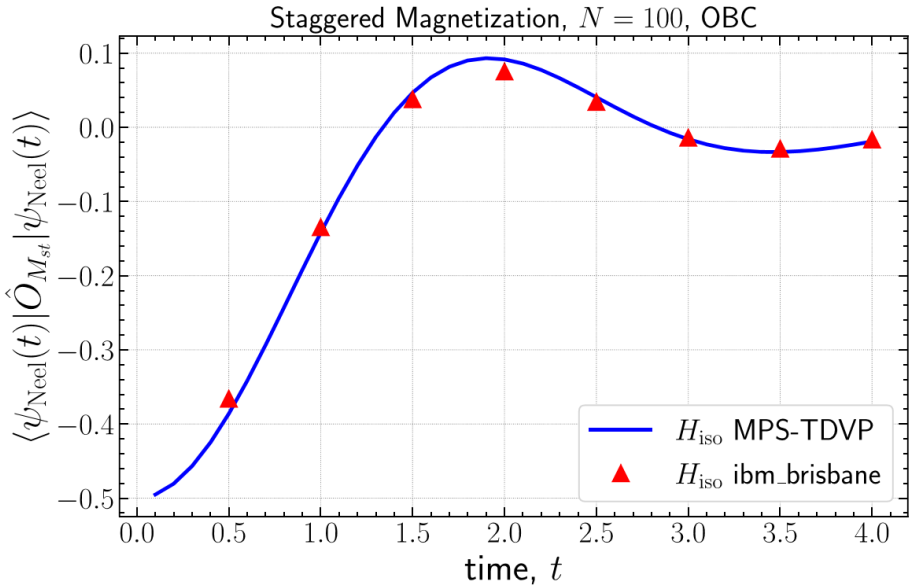


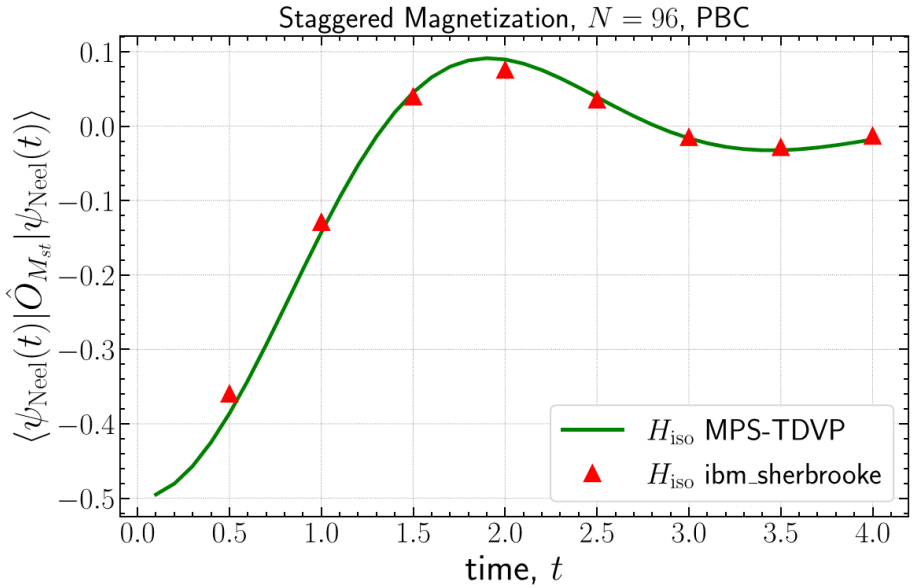


Brookhaven National Laboratory

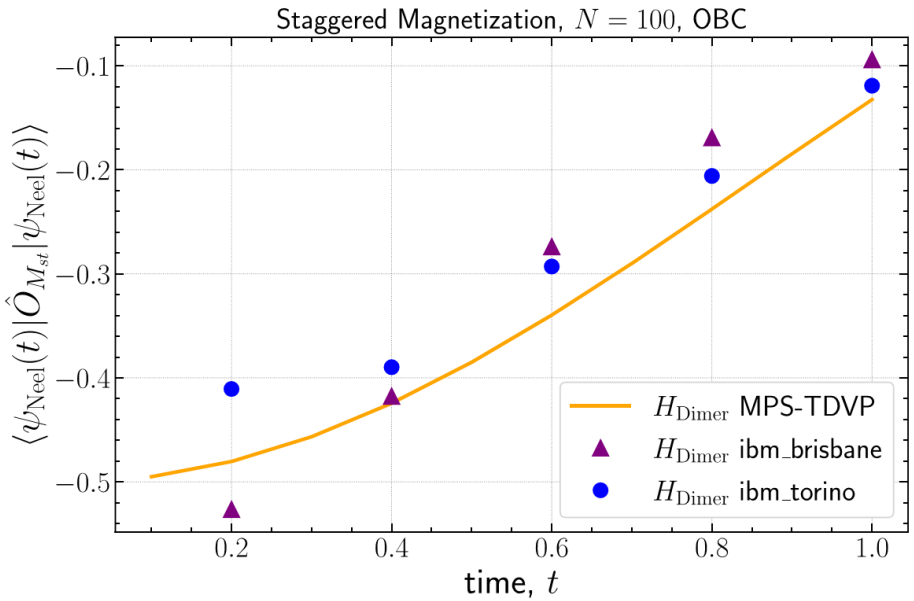
Results: N=96, 100 sites (qubits)

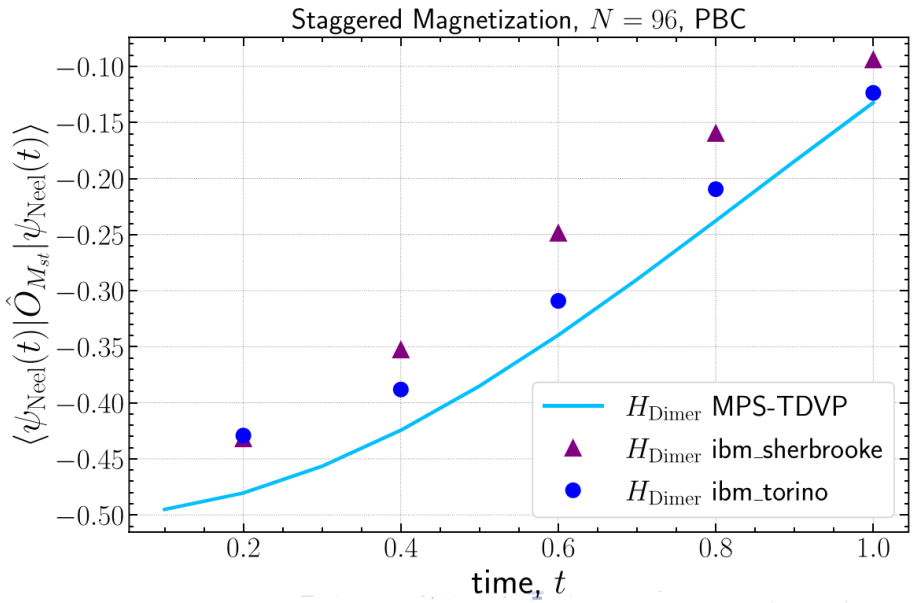
H_{XXZ}





$H_{J_1-J_2}$

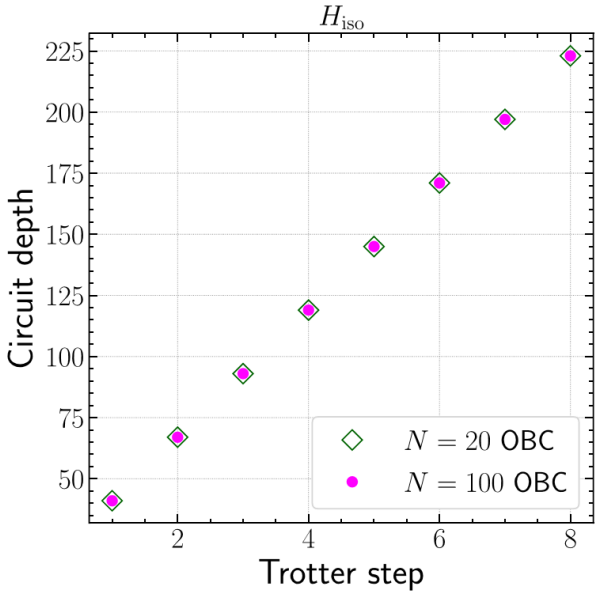


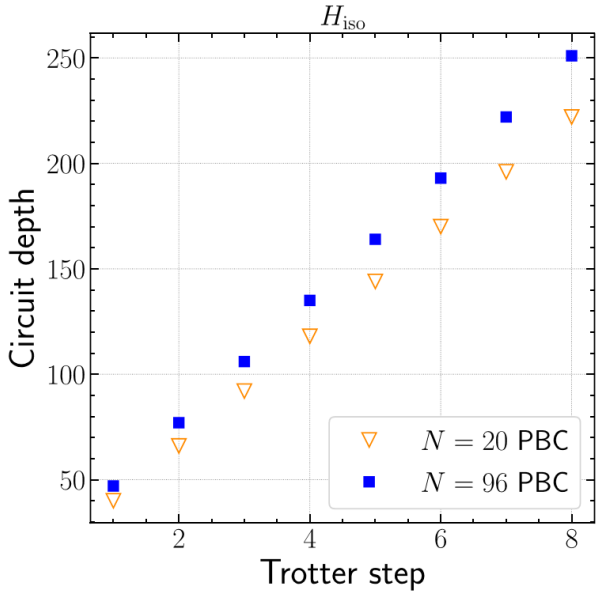


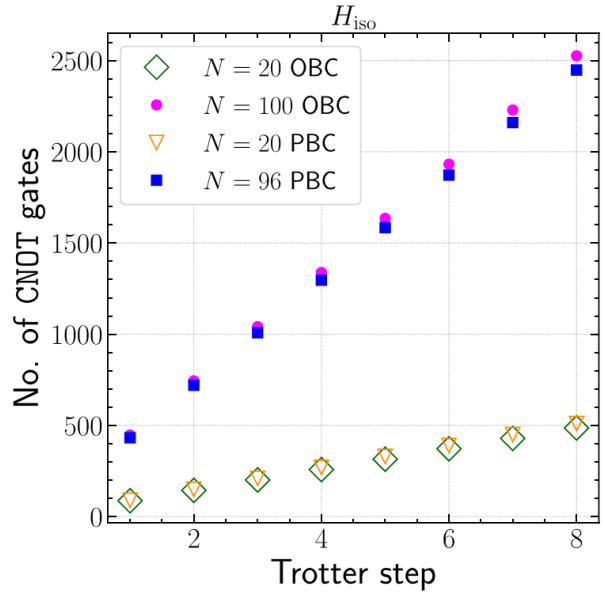
Brookhaven National Laboratory

Results: Circuit Depths & CX Gate Counts

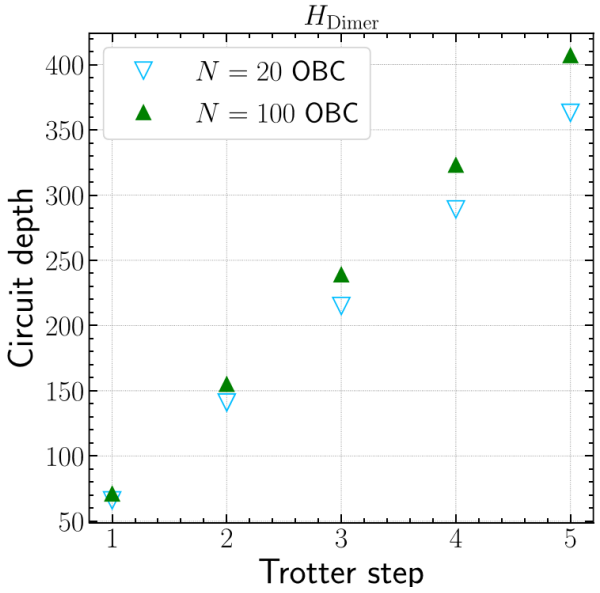
H_{XXZ}

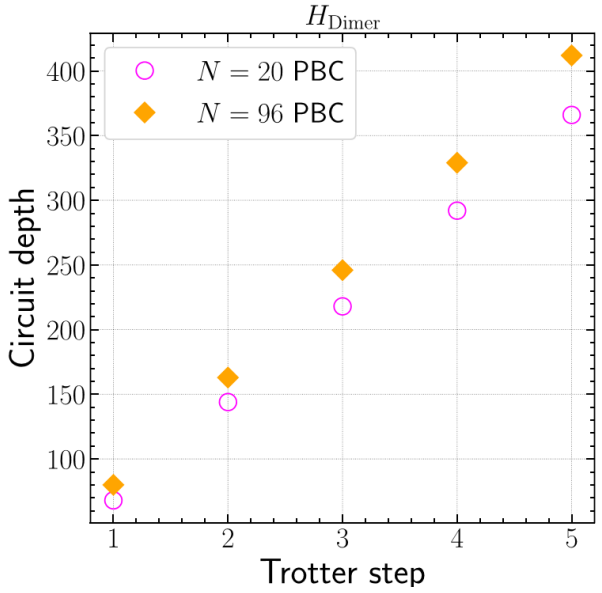


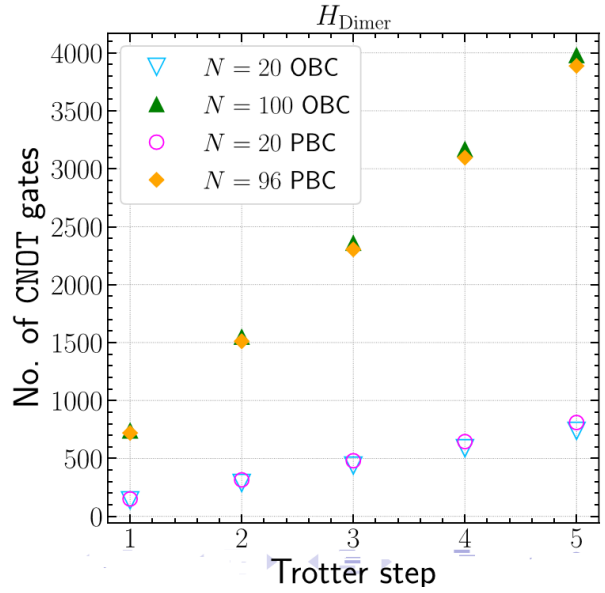




$H_{J_1-J_2}$







Brookhaven National Laboratory

Toward Complex Hamiltonians
(Extension of Noise & Hardware-aware Design)

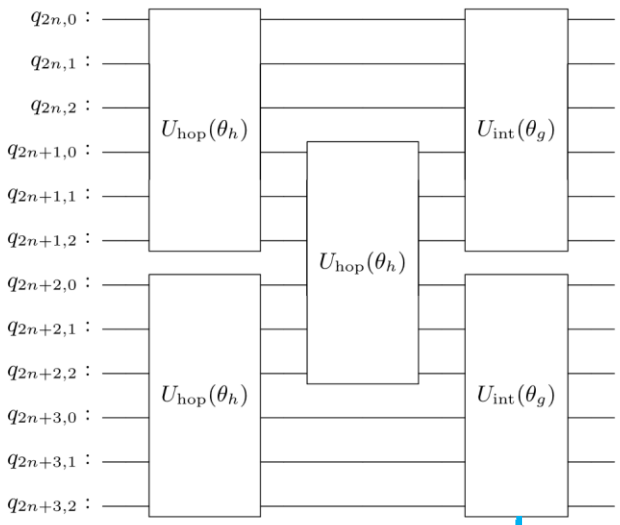
Gross-Neveu model

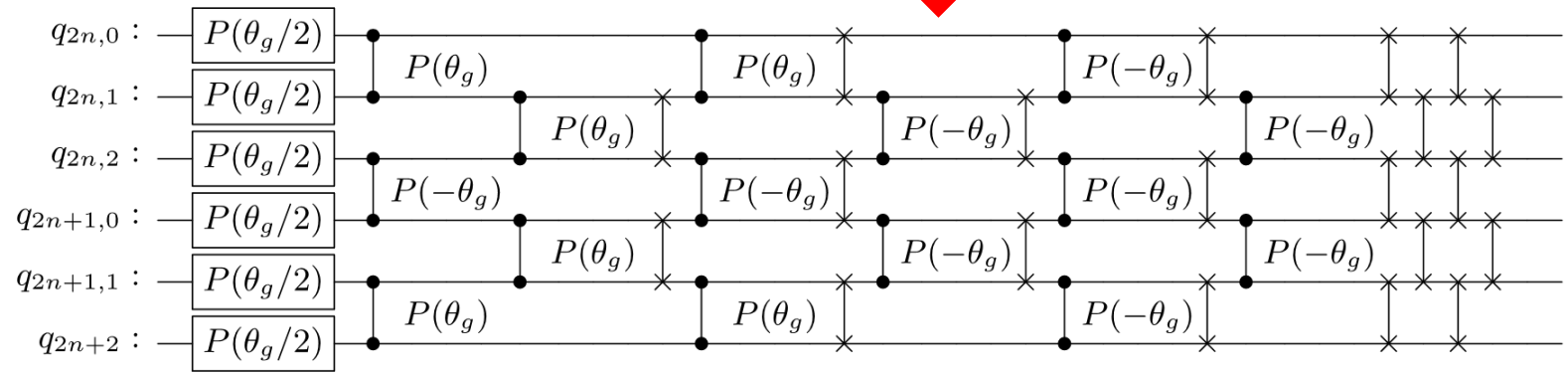
- The Gross-Neveu model is a toy quantum field theory (QFT) model of Dirac fermions designed to study the core features of strong interactions (QCD) in a simplified setting.
- 1+1 Dimensions: It is restricted to one spatial and one time dimension for mathematical simplicity.
- **Four-Fermion Interaction**: It describes **N flavors** of Dirac fermions interacting via a quartic self-coupling term. Four-fermion interactions map to **diagonal multi-qubit terms, leading to costly diagonal unitary synthesis** in Trotterized quantum simulation.
- Asymptotic Freedom: The coupling constant becomes weaker at higher energy scales, mimicking real-world QCD.
- Dynamical Mass Generation: Fermions acquire mass through quantum corrections, even if their bare mass is zero.
- Spontaneous Chiral Symmetry Breaking: The discrete chiral symmetry of the model breaks down spontaneously at lower energies.
- Toy Model Utility: It serves as a laboratory to analyze complex non-perturbative phenomena with exact solvability at large N.

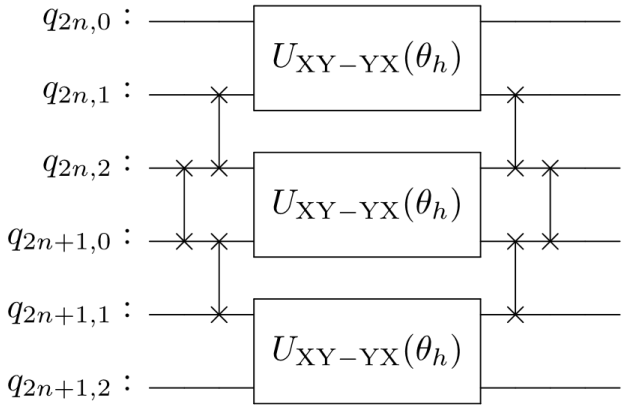
Trotterization of the 3-flavor GN model

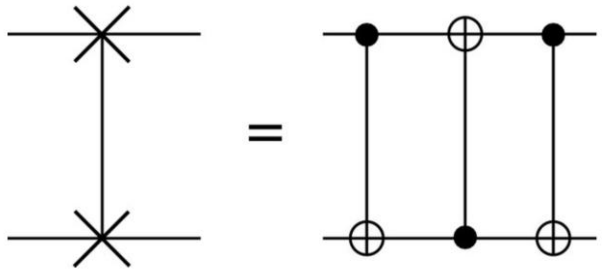
$$\tilde{H} = \tilde{H}_{\text{quadratic}} + \tilde{H}_{\text{quartic}}$$

$$e^{-i\tilde{H}_{\text{quartic}}\delta t/\hbar}$$





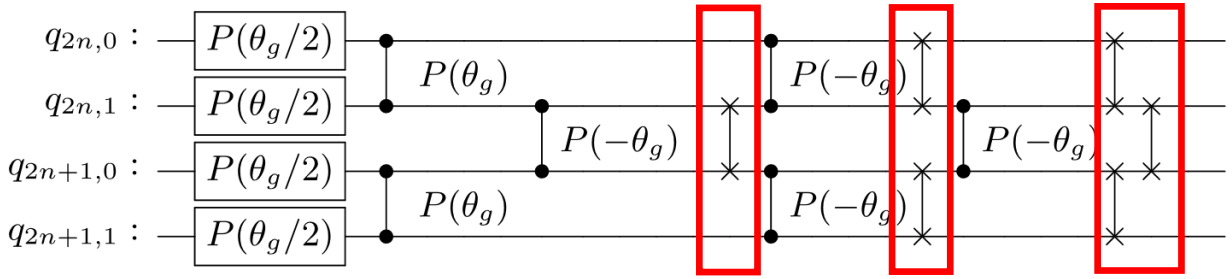




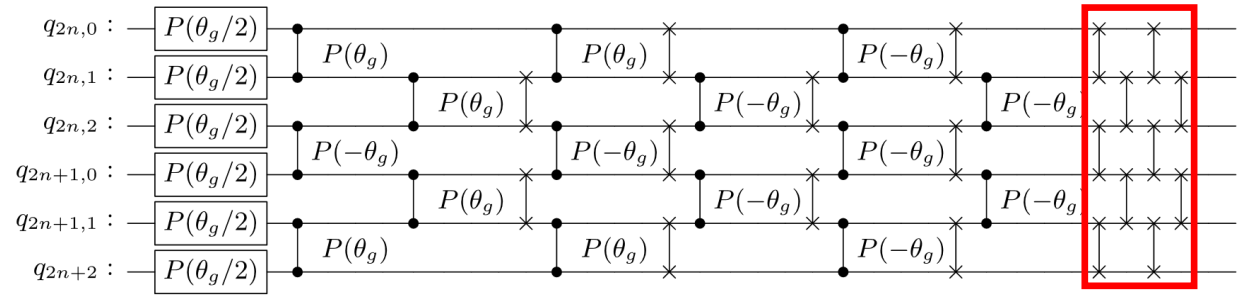
Brookhaven National Laboratory

N-flavor Gross-Neveu Model

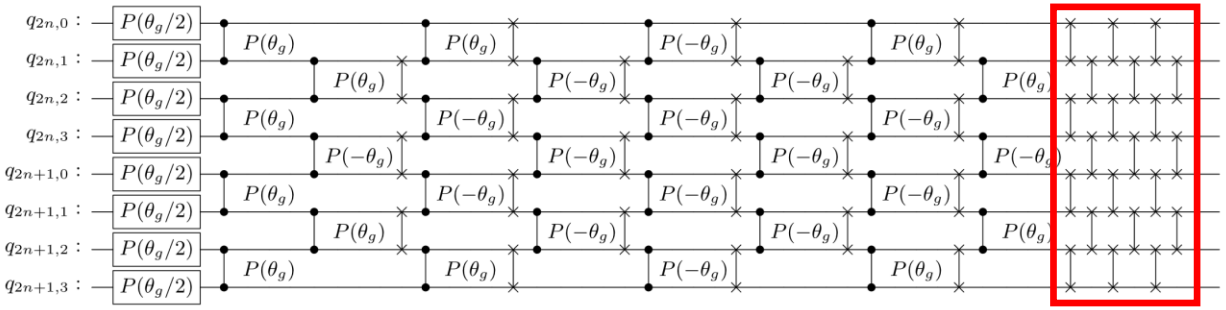
2-flavor

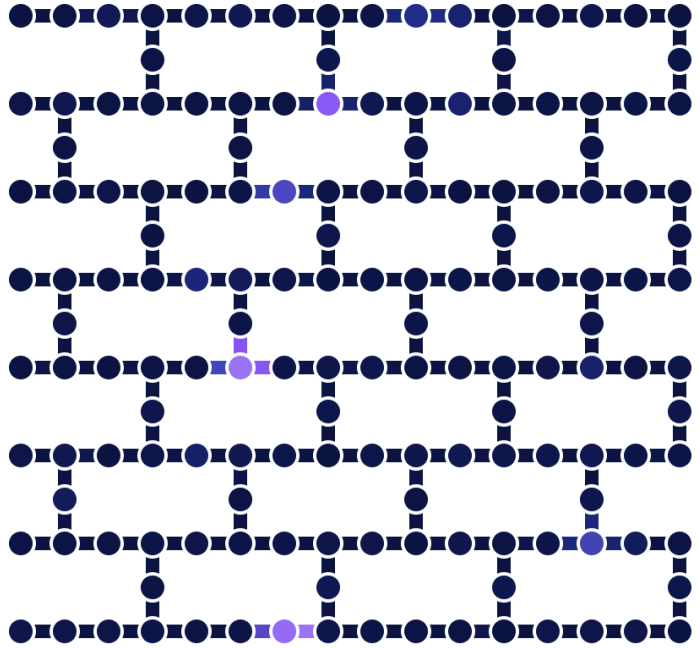


3-flavor



4-flavor





ibm_boston

Brookhaven National Laboratory

Localized Diagonal Operator App. (LDOA)

Algorithm 1: Localized Diagonal Operator Approximation (LDOA)

Require: Target diagonal unitary $U_{\text{target}}(\vec{\phi})$
Require: Hardware-efficient Ansatz $U_{\text{Ansatz}}(\vec{x})$
Ensure: Optimal parameter vector $\vec{x}_{\text{opt}}$

1: Extract target phase vector $\vec{\phi} \in \mathbb{R}^{2^n}$

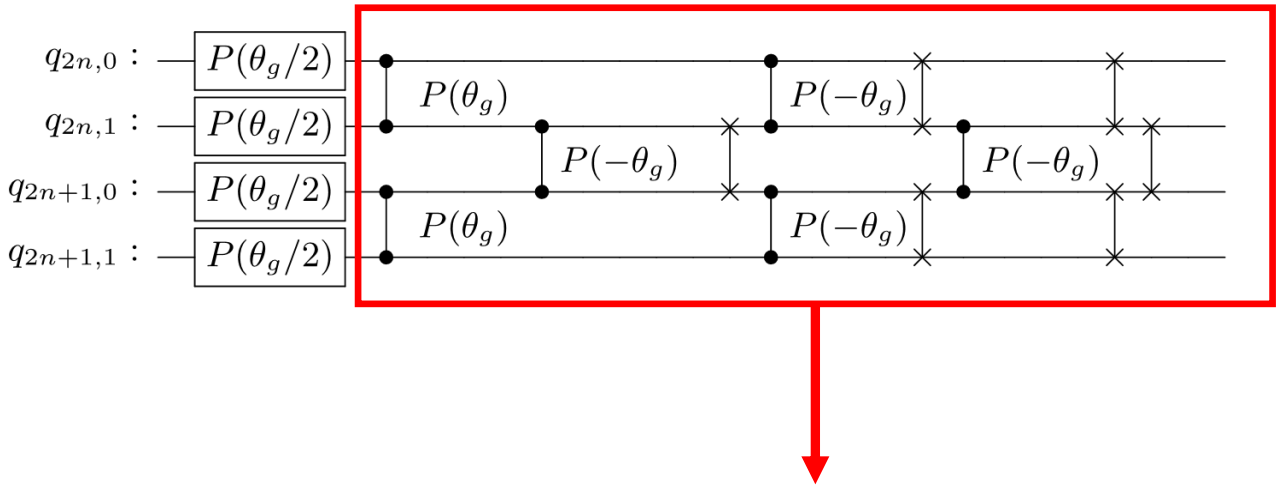
2: Construct the Ansatz phase map $\vec{\theta}(\vec{x})$

3: Formulate the linear least-squares system $A\vec{x} \approx \vec{b}$

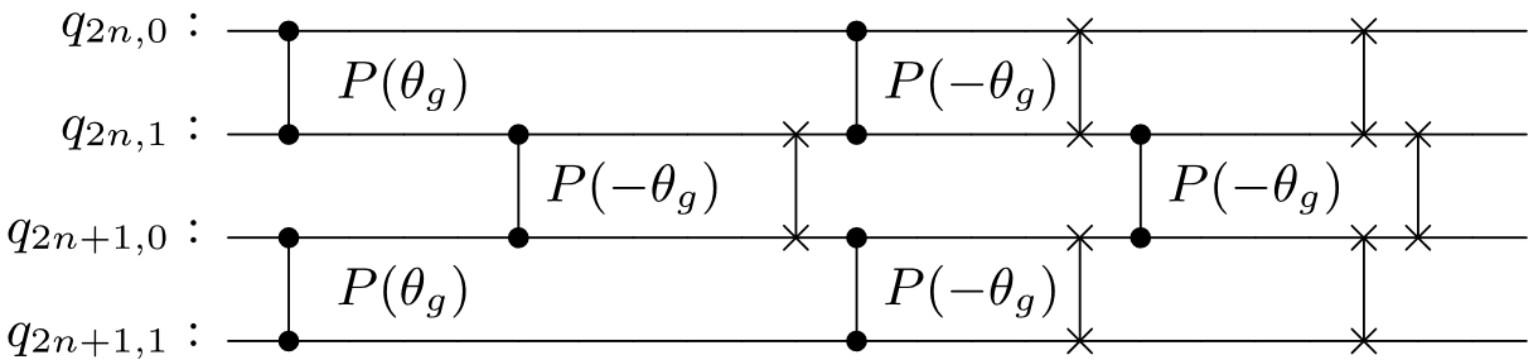
4: Compute the pseudoinverse solution $\vec{x}_{\text{opt}} = A^{+}\vec{b}$

5: Construct the approximated diagonal operator $U_{\text{Ansatz}}(\vec{x}_{\text{opt}})$

LDOA (2-flavor Ex.)

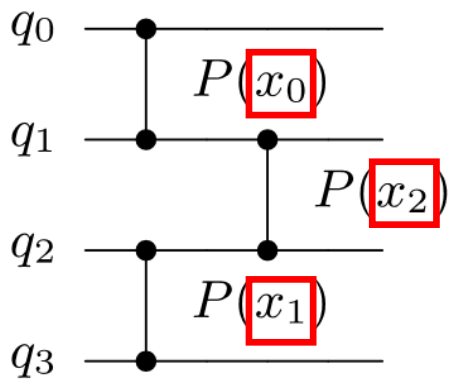


Goal: To approximate the **target** circuit using the **Ansatz** (hardware-efficient) circuit.
In other words, finding the parameters, **x0**, **x1**, and **x2**, minimizing the difference of the diagonal operators between the **target** circuit and the **Ansatz** circuit.



Target Circuit

$\approx$



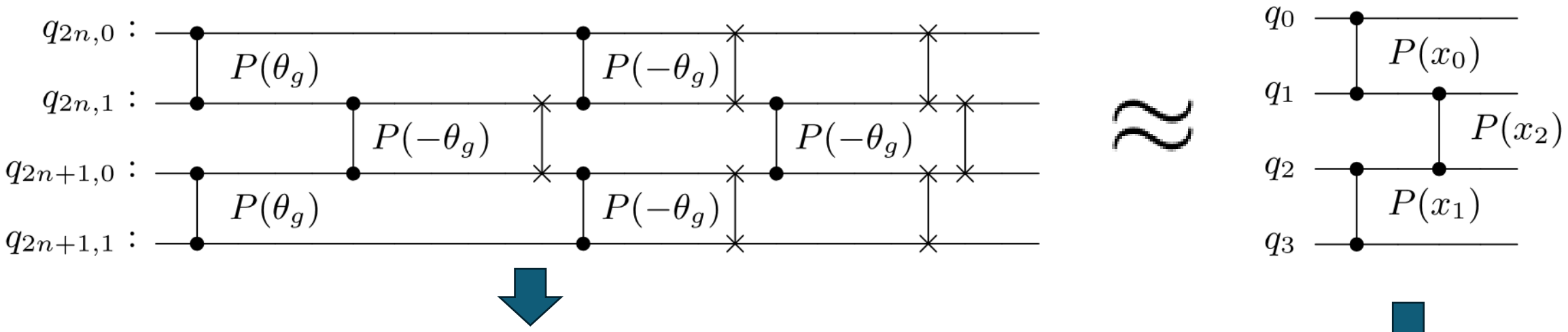
Ansatz Circuit

Brookhaven National Laboratory

LDOA (2-flavor Ex.): Phase Space Rep.

Note that any diagonal unitary operator on a quantum computer can be represented as a phase vector whose entries take the **form $e^{i\theta}$**.

$$\text{CP}(\theta) = I \otimes |0\rangle\langle 0| + P \otimes |1\rangle\langle 1| = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & e^{i\theta} \end{pmatrix}$$



$$U_{\text{target}}(\vec{\phi}(\theta_g)) = \text{diag}\left(1, 1, 1, e^{i\theta_g}, 1, e^{-i\theta_g}, e^{-i\theta_g}, e^{-i\theta_g}, 1, e^{-i\theta_g}, e^{-i\theta_g}, e^{-i\theta_g}, e^{i\theta_g}, e^{-i\theta_g}, e^{-i\theta_g}, e^{-2i\theta_g}\right)$$

$$U_{\text{Ansatz}}(\vec{\theta}(x_0, x_1, x_2)) = \text{diag}\left(1, 1, 1, e^{ix_0}, 1, 1, e^{ix_2}, e^{i(x_0+x_2)}, 1, 1, 1, e^{ix_0}, e^{ix_1}, e^{ix_1}, e^{i(x_1+x_2)}, e^{i(x_0+x_1+x_2)}\right)$$

LDOA (2-flavor Ex.): Phase Space Opt.

$$U_{\text{target}}(\vec{\phi}(\theta_g)) = \text{diag}\left(1,1,1,e^{i\theta_g},1,e^{-i\theta_g},e^{-i\theta_g},e^{-i\theta_g},1,e^{-i\theta_g},e^{-i\theta_g},e^{-i\theta_g},e^{i\theta_g},e^{-i\theta_g},e^{-i\theta_g},e^{-2i\theta_g}\right)$$

$$U_{\text{Ansatz}}(\vec{\theta}(x_0,x_1,x_2)) = \text{diag}\left(1,1,1,e^{ix_0},1,1,e^{ix_2},e^{i(x_0+x_2)},1,1,1,e^{ix_0},e^{ix_1},e^{ix_1},e^{i(x_1+x_2)},e^{i(x_0+x_1+x_2)}\right)$$

$$\Rightarrow \begin{cases} x_0 = \theta_g \\ 0 = -\theta_g \\ x_2 = -\theta_g \\ x_0 + x_2 = -\theta_g \\ 0 = -\theta_g \\ 0 = -\theta_g \\ x_0 = -\theta_g \\ x_1 = \theta_g \\ x_1 = -\theta_g \\ x_1 + x_2 = -\theta_g \\ x_0 + x_1 + x_2 = -2\theta_g \end{cases}$$

Our goal is to find $\vec{x}$ satisfying

$$\min_{\vec{x}\in\mathbb{R}^m} \left\| U_{\text{target}}(\vec{\phi}(\theta_g)) - U_{\text{Ansatz}}(\vec{\theta}(\vec{x})) \right\|$$

for the given θ_g and the L2 norm, $\|\cdot\|$.

Since we have the first-order Taylor expansion $e^{ix} \approx 1 + ix$,

We have $\left\| U_{\text{target}}(\vec{\phi}) - U_{\text{Ansatz}}(\vec{\theta}(\vec{x})) \right\| \approx \left\| \vec{\phi}(\theta_g) - \vec{\theta}(\vec{x}) \right\|$

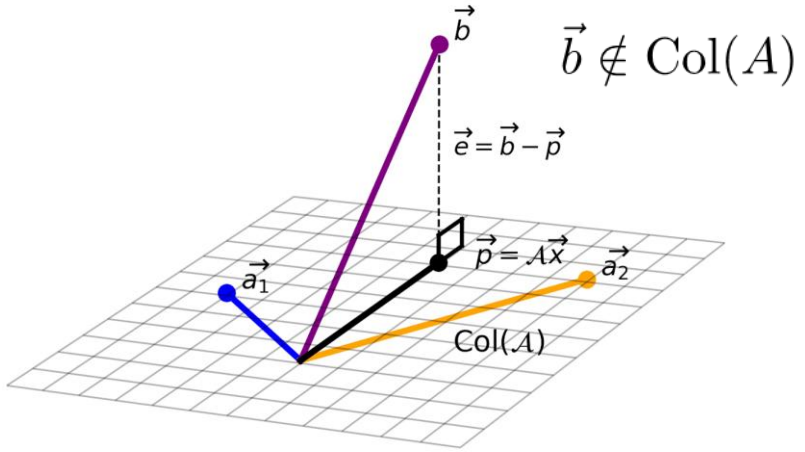
when we have $\vec{\phi}$ and $\vec{\theta}$ $(\vec{x})$ which are close to $\vec{0}$.

This is justified because $\theta_g = \Delta t g^2/a$ and Δt is a Trotter step size.

Finding $\vec{x}$ minimizing $\left\| \vec{\phi}(\theta_g) - \vec{\theta}(\vec{x}) \right\|$ for given θ_g is a linear least-square problem.

LDOA (2-flavor Ex.): Least-Square Prob.

$$\begin{cases} x_0 = \theta_g \\ 0 = -\theta_g \\ x_2 = -\theta_g \\ x_0 + x_2 = -\theta_g \\ 0 = -\theta_g \\ 0 = -\theta_g \\ x_0 = -\theta_g \\ x_1 = \theta_g \\ x_1 = -\theta_g \\ x_1 + x_2 = -\theta_g \\ x_0 + x_1 + x_2 = -2\theta_g \end{cases} \Rightarrow A = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}, \quad \vec{x} = \begin{pmatrix} x_0 \\ x_1 \\ x_2 \end{pmatrix}, \quad \vec{b} = \begin{pmatrix} \theta_g \\ -\theta_g \\ -\theta_g \\ -\theta_g \\ -\theta_g \\ -\theta_g \\ -\theta_g \\ -\theta_g \\ \theta_g \\ -\theta_g \\ -\theta_g \\ -2\theta_g \end{pmatrix}$$



Geometric interpretation
of a least-square problem

$$\vec{x}_{\mathrm{CP}} = A^{+}\vec{b} \quad \text{where} \quad A^{+} = (A^T A)^{-1} A^T$$

$$A^{+} = \begin{pmatrix} \frac{1}{3} & 0 & -\frac{1}{6} & \frac{1}{6} & 0 & 0 & \frac{1}{3} & 0 & 0 & -\frac{1}{6} & \frac{1}{6} \\ 0 & 0 & -\frac{1}{6} & -\frac{1}{6} & 0 & 0 & 0 & \frac{1}{3} & \frac{1}{3} & \frac{1}{6} & \frac{1}{6} \\ -\frac{1}{6} & 0 & \frac{5}{12} & \frac{1}{4} & 0 & 0 & -\frac{1}{6} & -\frac{1}{6} & -\frac{1}{6} & \frac{1}{4} & \frac{1}{12} \end{pmatrix}$$

Moore-Penrose pseudoinverse

$$\vec{x}_{\mathrm{CP}} = \begin{pmatrix} -\frac{\theta_g}{6} & -\frac{\theta_g}{6} & -\frac{13\theta_g}{12} \end{pmatrix}^T$$

minizize

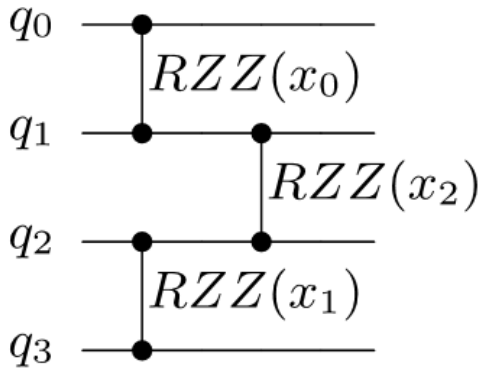
$$\left\| U_{\mathrm{target}}(\vec{\phi}) - U_{\mathrm{Ansatz}}(\vec{\theta}(\vec{x})) \right\| \approx \left\| \vec{\phi}(\theta_g) - \vec{\theta}(\vec{x}) \right\|$$

Brookhaven National Laboratory

LDOA (2-flavor Ex.): Summary

Since the Rzz gate is also a diagonal operator,
One can adopt the Rzz Ansatz.

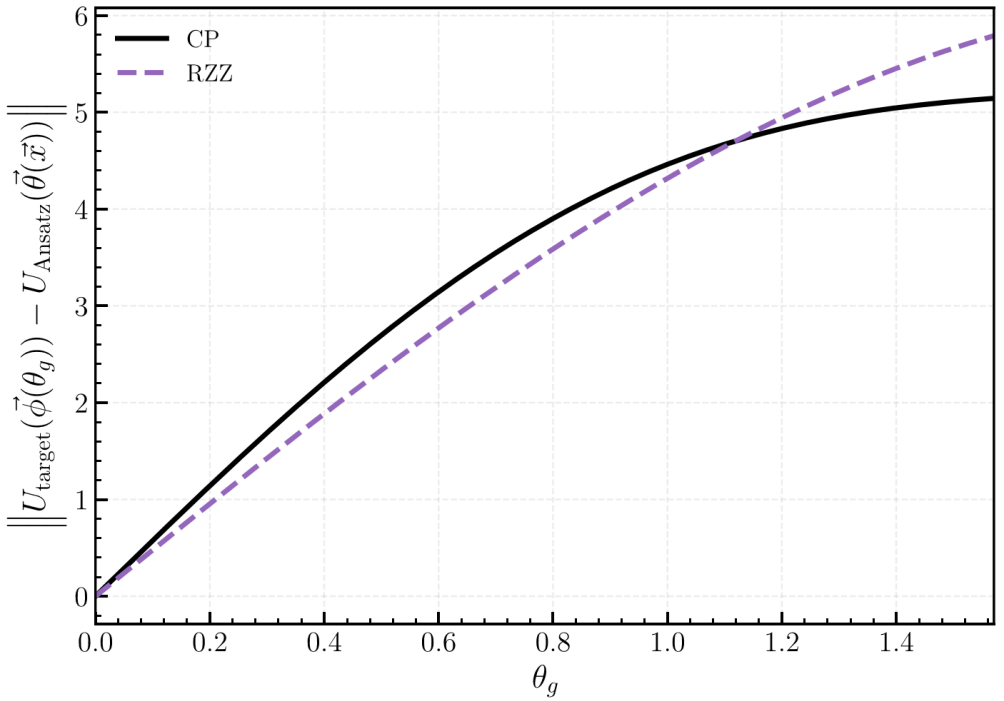
$$\text{Rzz}(\theta) = \exp\left(-i\frac{\theta}{2}Z\otimes Z\right) = \begin{pmatrix} e^{-i\frac{\theta}{2}} & 0 & 0 & 0 \\ 0 & e^{i\frac{\theta}{2}} & 0 & 0 \\ 0 & 0 & e^{i\frac{\theta}{2}} & 0 \\ 0 & 0 & 0 & e^{-i\frac{\theta}{2}} \end{pmatrix}$$



$$\vec{x}_{\text{Rzz}} = \begin{pmatrix} -\frac{\theta_g}{2} & -\frac{\theta_g}{2} & \frac{\theta_g}{2} \end{pmatrix}^T$$

$$\left\| U_{\text{target}}(\vec{\phi}(\theta_g)) - U_{\text{Ansatz}}(\vec{\theta}(\vec{x}_{\text{CP}})) \right\|^2 = 22 - 2\cos\left(\frac{\theta_g}{12}\right) - 4\cos\left(\frac{\theta_g}{4}\right) - 2\cos\left(\frac{7\theta_g}{12}\right) - 4\cos\left(\frac{5\theta_g}{6}\right) - 6\cos(\theta_g) - 4\cos\left(\frac{7\theta_g}{6}\right)$$

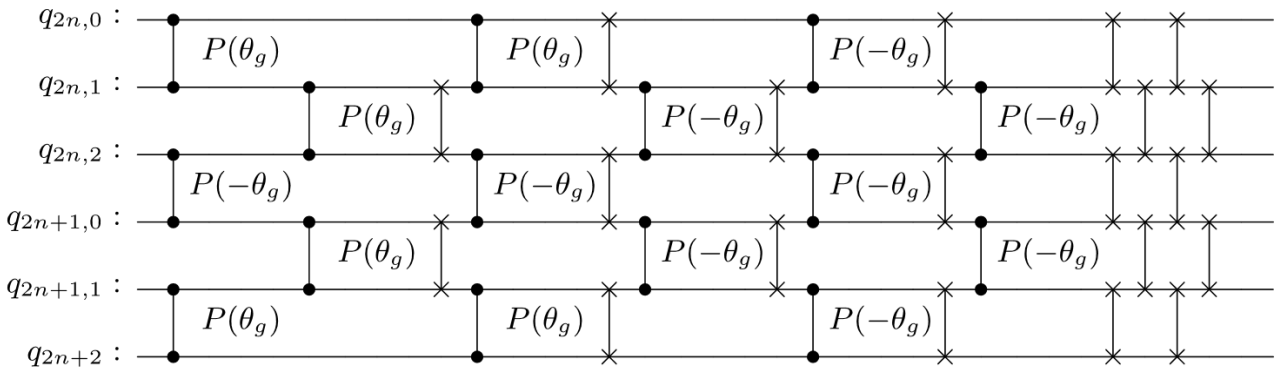
$$\left\| U_{\text{target}}(\vec{\phi}(\theta_g)) - U_{\text{Ansatz}}(\vec{\theta}(\vec{x}_{Rzz})) \right\|^2 = 32 - 4\cos(\frac{\theta_g}{4}) - 2\cos(\frac{3\theta_g}{8}) - 4\cos(\frac{5\theta_g}{8}) - 4\cos\theta_g - 4\cos(\frac{5\theta_g}{4}) - 4\cos(\frac{11\theta_g}{8}) - 4\cos(\frac{15\theta_g}{8}) - 4\cos(\frac{9\theta_g}{4}) - 2\cos(\frac{29\theta_g}{8}).$$



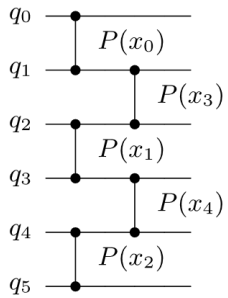
Brookhaven National Laboratory

LDOA (3-flavor Ex.)

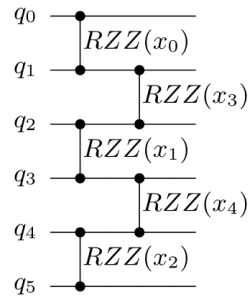
The target circuit of 3-flavor



$\approx$



(a) Ansatz circuit (CP gates)

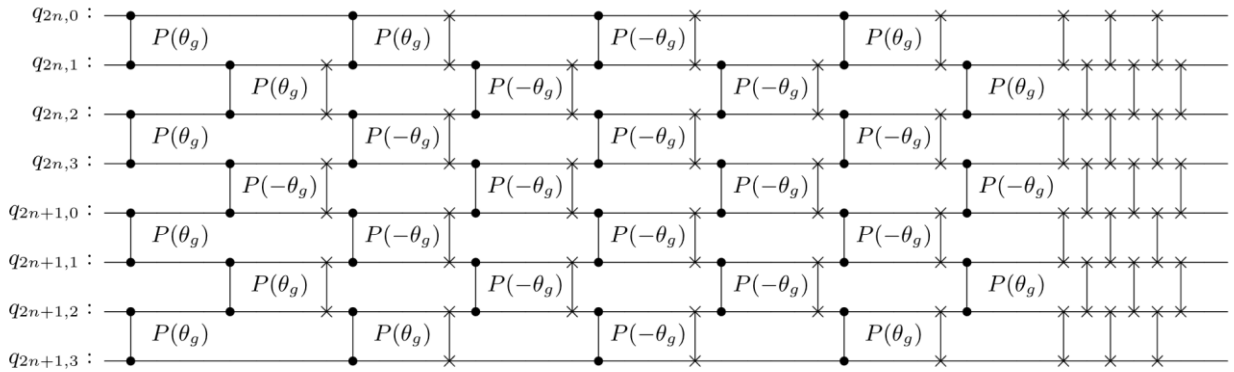


(b) Ansatz circuit (RZZ gates)

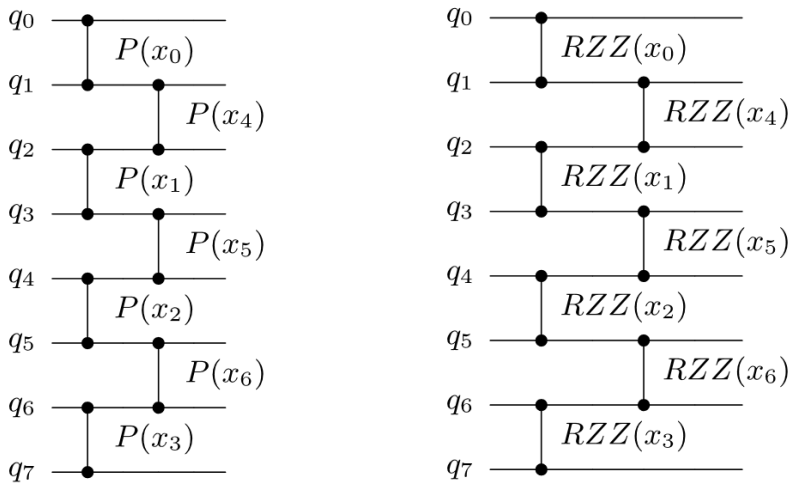
$$\vec{x}_{\mathrm{CP}} = \left(-\frac{11\theta_g}{19} \quad -\frac{24\theta_g}{19} \quad -\frac{11\theta_g}{19} \quad \frac{\theta_g}{19} \quad \frac{\theta_g}{19}\right)^T$$

$$\vec{x}_{\mathrm{Rzz}} = \left(-\frac{\theta_g}{2} \quad \frac{\theta_g}{2} \quad -\frac{\theta_g}{2} \quad -\frac{\theta_g}{2} \quad -\frac{\theta_g}{2}\right)^T$$

LDOA (4-flavor Ex.)



$\approx$



(a) Ansatz circuit (CP gates) (b) Ansatz circuit (RZZ gates)

$$\vec{x}_{\mathrm{CP}} = \left(-\frac{55\theta_g}{118} \quad -\frac{3\theta_g}{59} \quad -\frac{3\theta_g}{59} \quad -\frac{55\theta_g}{118} \quad -\frac{26\theta_g}{59} \quad -\frac{147\theta_g}{118} \quad -\frac{26\theta_g}{59}\right)^T$$

$$\vec{x}_{\mathrm{Rzz}} = \left(-\frac{\theta_g}{2} \quad -\frac{\theta_g}{2} \quad -\frac{\theta_g}{2} \quad -\frac{\theta_g}{2} \quad -\frac{\theta_g}{2} \quad \frac{\theta_g}{2} \quad -\frac{\theta_g}{2}\right)^T$$

Localized Diagonal Operator App. (LDOA)

Algorithm 1: Localized Diagonal Operator Approximation (LDOA)

Require: Target diagonal unitary $U_{\text{target}}(\vec{\phi})$
Require: Hardware-efficient Ansatz $U_{\text{Ansatz}}(\vec{x})$
Ensure: Optimal parameter vector $\vec{x}_{\text{opt}}$

1: Extract target phase vector $\vec{\phi} \in \mathbb{R}^{2^n}$

2: Construct the Ansatz phase map $\vec{\theta}(\vec{x})$

3: Formulate the linear least-squares system $A\vec{x} \approx \vec{b}$

4: Compute the pseudoinverse solution $\vec{x}_{\text{opt}} = A^{+}\vec{b}$

5: Construct the approximated diagonal operator $U_{\text{Ansatz}}(\vec{x}_{\text{opt}})$

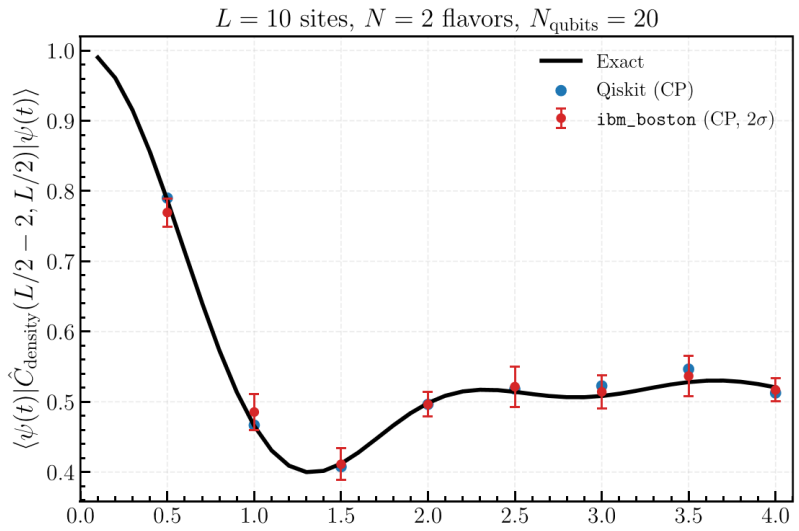
1. Phase-space representation
2. Ansatz-phase construction
3. Phase-space optimization
4. Linear least-squares formulation

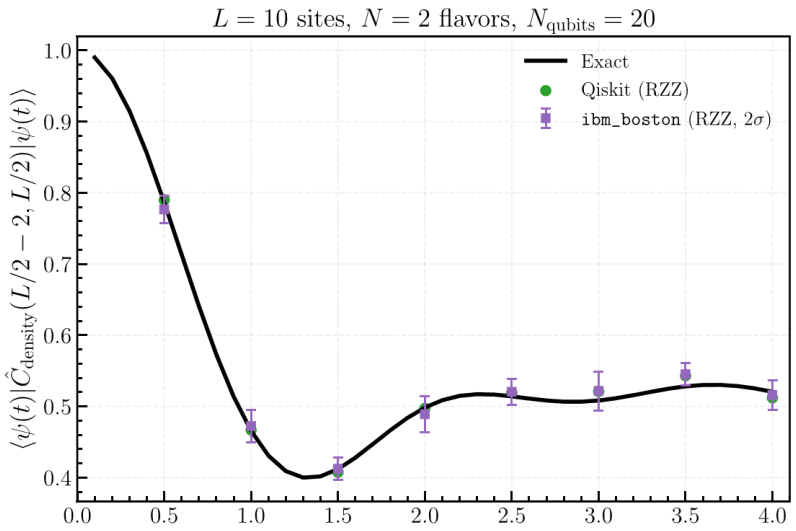
Toward Complex Hamiltonians
(Simulation Results)

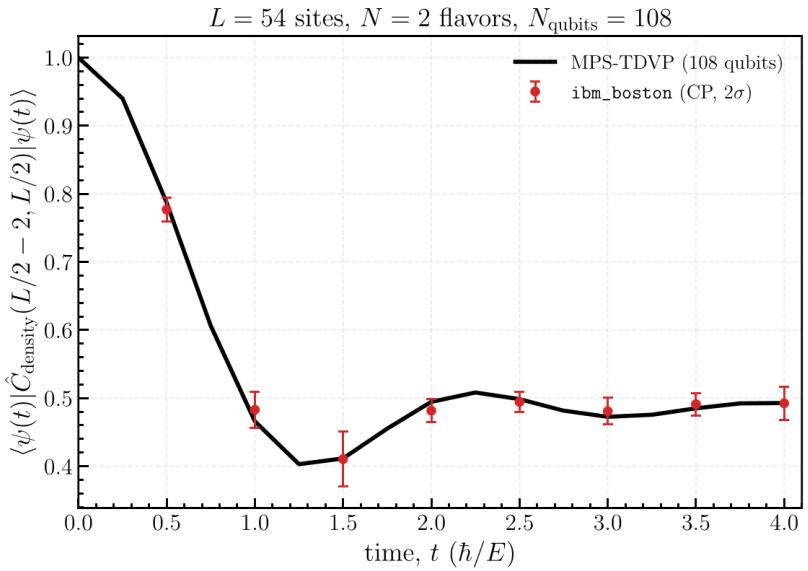
Time Evolution of 2-flavor GN model

10 sites, **20** qubits

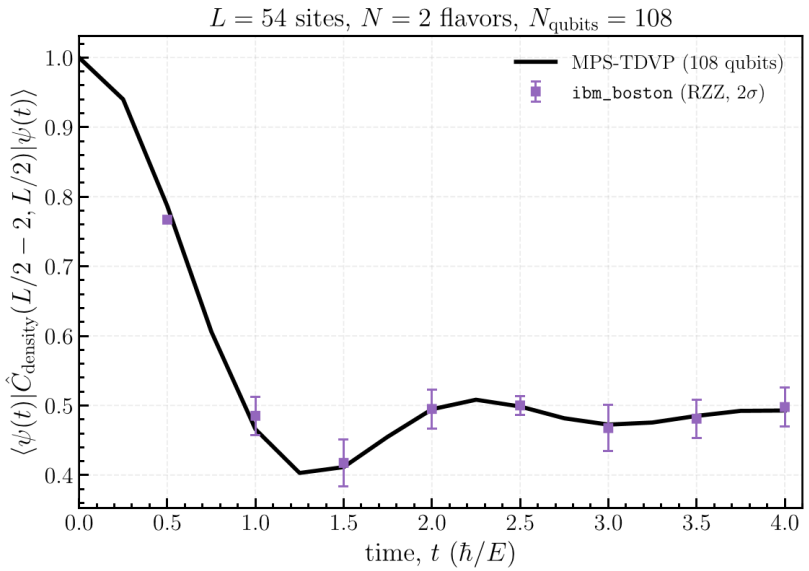
54 sites, **108** qubits







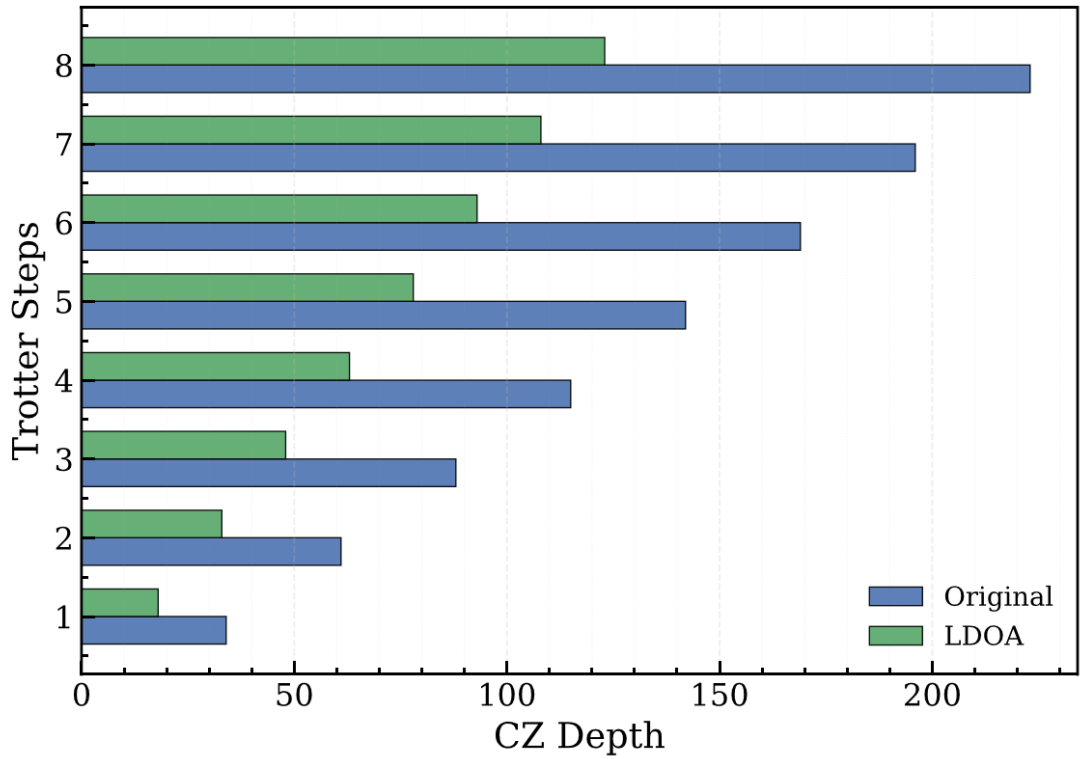
(a) CP Approxiamtion



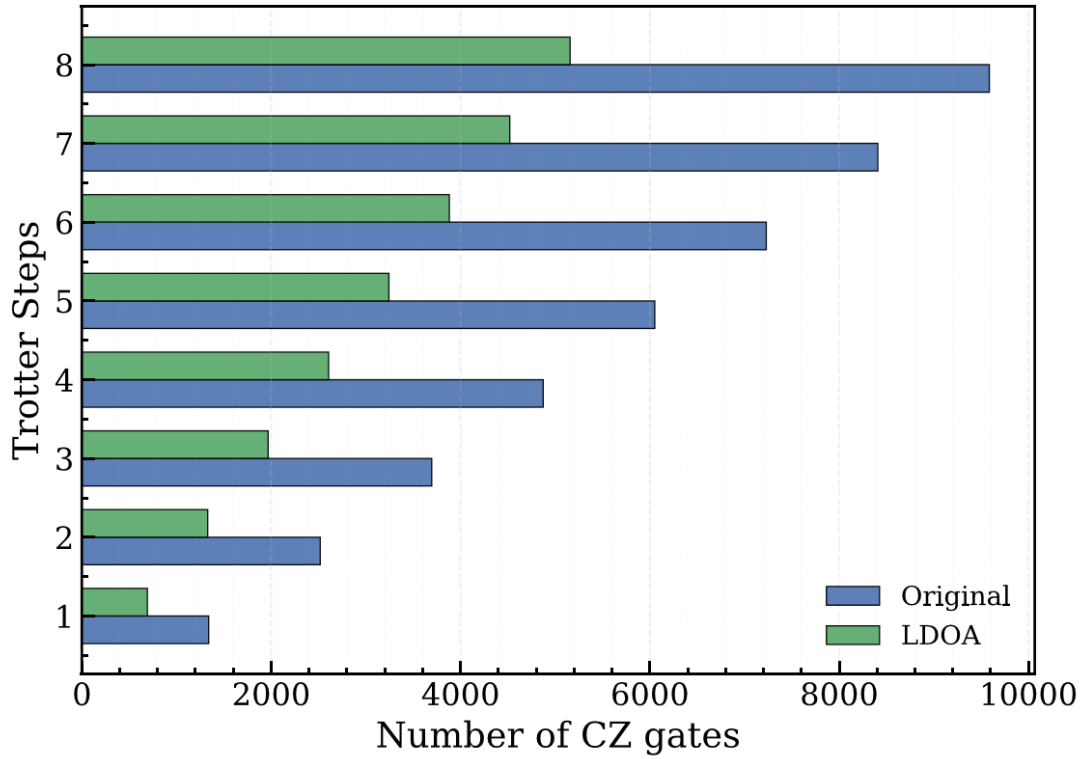
(b) RZZ Approximation

Brookhaven National Laboratory

Time Evolution of 2-flavor GN model



(a) CZ Depth of 108-qubit system



(b) Number of CZ gates of 108-qubit system

Perspective & Conclusion

Hardware-aware Design for Neutral Atom

- **Finite-range native interactions**: Neutral-atom systems do not provide true global all-to-all connectivity. Gate performance strongly depends on atom distance, blockade radius, and interaction geometry, making connectivity-aware algorithm design essential.
- **Dynamic but costly qubit movement**: Atom transport enables reconfigurable connectivity, but movement introduces latency, heating, decoherence, and atom-loss risks. Hardware-aware algorithms can minimize transport overhead and movement frequency.
- **Strong geometry dependence**: Performance depends heavily on the spatial arrangement of atoms (1D, 2D, arbitrary graphs, lattices). Algorithm mapping and qubit placement must be co-designed with hardware geometry.
- **Transport-aware compilation requirements**: Qubit routing in neutral-atom systems is fundamentally different from fixed-layout superconducting architectures. Compilation must jointly optimize gate scheduling, atom rearrangement, and interaction locality.
- **Tradeoff between connectivity and fidelity**: Increasing interaction range or transport complexity can reduce gate fidelity. Hardware-aware design is necessary to balance circuit depth, routing cost, and execution accuracy.
- **Scalability challenges beyond connectivity**: Large-scale neutral-atom systems introduce scheduling, calibration, control-resource, and synchronization constraints that require architecture-aware algorithmic optimization.
- **Utility-scale quantum simulation demands**: Many promising applications for neutral atoms, including quantum simulation and many-body dynamics, require co-design across hardware, compilation, noise mitigation, and algorithm structure to achieve practical quantum utility.

QC Regimes & Constraints

- NTQC
  - Hardware and Noise-aware design is critical (not optional).
  - hardware-aware compilation
  - circuit optimization (squeeze accuracy)
- Early FTQC:
  - Logical qubits with non-negligible error rates (~10^{-6}, 10^{-8}): logical qubits are still noisy.
  - Decoding + syndrome noise (and cost)
  - Scheduling + architecture constraints remain
  - Trade-off between logical gate and connectivity; Surface code (lattice surgery) vs Color code (3D connectivity)
  - mid-circuit measurement and feedforward latency: key system-level bottleneck in early FTQC
- Mature (full) FTQC:
  - still far from predictable realization
  - idealized abstraction
  - algorithmic noise tolerance dominates: analogous to numerical analysis in classical computing, where stability and conditioning govern robustness under finite precision errors

Myth and Reality of Early FTQC

Myth	Reality
FTQC removes the need for hardware-aware algorithm design.	Logical qubits still operate under stochastic, hardware-dependent error processes.
Error correction makes hardware-aware design irrelevant.	Decoding, scheduling, and architecture constraints remain hardware-sensitive.
Algorithms become fully abstract in FTQC.	Algorithm performance still depends on system-level noise structure and implementation details.
Only physical-level noise matters in NTQC.	In FTQC, noise is transformed, not eliminated, and persists at the logical level.
Classical machine epsilon comparison implies FTQC is effectively exact.	Error semantics differ: quantum errors are stochastic and circuit-dependent, not deterministic numerical rounding.

Fault tolerance (of early FTQC) changes the error layer, NOT the relevance of hardware and noise-aware algorithm design.

Thank You!

kyu@bnl.gov